Boosting faith in the authenticity of open source software

December 1 2023, by Steve Nadis

Credit: Alex Shipps and Kelsey Merrill via Midjourney/Massachusetts Institute of Technology

Open source software—software that is freely distributed, along with its source code, so that copies, additions, or modifications can be readily

made —is "everywhere," to quote the 2023 Open Source Security and Risk Analysis Report. Of the computer programs used by major industries, 96% include open source software, and 76% of those programs consists of open source software. But the percentage of software packages "containing security vulnerabilities remains troublingly high," the report warned.

One concern is that "the software you've gotten from what you believe to be a reliable developer has somehow been compromised," says Kelsey Merrill, a software engineer who received a master's degree earlier this year from MIT's Department of Electrical Engineering and Computer Science. "Suppose that somewhere in the supply chain, the software has been changed by an attacker who has malicious intent."

The risk of a security breach of this sort is by no means abstract. In 2020, to take a notorious example, the Texas company SolarWinds made a software update to its widely used program called Orion. Hackers broke into the system, inserting pernicious code into the software before SolarWinds shipped the latest version of Orion to more than 18,000 customers, including Microsoft, Intel, and roughly 100 other companies, as well as a dozen U.S. government agencies—including the Departments of State, Defense, Treasury, Commerce, and Homeland Security.

In this case, the product that was corrupted came from a large commercial company, but lapses may be even more likely to occur in the open source realm, Merrill says, "where people of varying backgrounds—many of whom are hobbyists without any security training—can publish software that gets used around the world."

She and three collaborators—her former advisor Karen Sollins, a Principal Scientist at the MIT Computer Science and Artificial Intelligence Laboratory; Santiago Torres-Arias, an assistant professor of

computer science at Purdue University; and Zachary Newman, a former MIT graduate student and current research scientist at Chainguard Labs—have developed a new system called Speranza, which is aimed at reassuring software consumers that the product they are getting has not been tampered with and is coming directly from a source they trust. The paper is published on the *arXiv* preprint server.

"What we have done," explains Sollins, "is to develop, prove correct, and demonstrate the viability of an approach that allows the [software] maintainers to remain anonymous." Preserving anonymity is obviously important, given that almost everyone—software developers included—value their confidentiality. This new approach, Sollins adds, "simultaneously allows [software] users to have confidence that the maintainers are, in fact, legitimate maintainers and, furthermore, that the code being downloaded is, in fact, the correct code of that maintainer."

So how can users confirm the genuineness of a software package in order to guarantee, as Merrill puts it, "that the maintainers are who they say they are?" The classical way of doing this, which was invented more than 40 years ago, is by means of a digital signature, which is analogous to a handwritten signature—albeit with far greater built-in security through the use of various cryptographic techniques.

To carry out a digital signature, two "keys" are generated at the same time—each of which is a number, composed of zeros and ones, that is 256 digits long. One key is designated "private," the other "public," but they constitute a pair that is mathematically linked.

A software developer can use their private key, along with the contents of the document or computer program, to generate a digital signature that is attached exclusively to that document or program. A software user can then use the public key—as well as the developer's signature, plus the contents of the package they downloaded—to verify the

package's authenticity.

Validation comes in the form of a yes or a no, a 1 or a zero. "Getting a 1 means that the authenticity has been assured," Merrill explains. "The document is the same as when it was signed and is hence unchanged. A 0 means something is amiss, and you may not want to rely on that document."

Although this decades-old approach is tried-and-true in a sense, it is far from perfect. One problem, Merrill notes, "is that people are bad at managing cryptographic keys, which consist of very long numbers, in a way that is secure and prevents them from getting lost." People lose their passwords all the time, Merrill says. "And if a software developer were to lose the private key and then contact a user saying, 'Hey, I have a new key,' how would you know who that really is?"

To address those concerns, Speranza is building off of "Sigstore"—a system introduced last year to enhance the security of the software supply chain. Sigstore was developed by Newman (who instigated the Speranza project) and Torres-Arias, along with John Speed Meyers of Chainguard Labs. Sigstore automates and streamlines the digital signing process. Users no longer have to manage long cryptographic keys but are instead issued ephemeral keys (an approach called "keyless signing") that expire quickly—perhaps within a matter of minutes—and therefore don't have to be stored.

A drawback with Sigstore stems from the fact that it dispensed with long-lasting public keys, so that software maintainers instead have to identify themselves—through a protocol called OpenID Connect (OIDC)—in a way that can be linked to their email addresses. That feature, alone, may inhibit the widespread adoption of Sigstore, and it served as the motivating factor behind—and the raison d'etre for—Speranza. "We take Sigstore's basic infrastructure and change it to provide privacy

guarantees," Merrill explains.

With Speranza, privacy is achieved through an original idea that she and her collaborators call "identity co-commitments." Here, in simple terms, is how the idea works: A software developer's identity, in the form of an email address, is converted into a so-called "commitment" that consists of a big pseudorandom number. (A pseudorandom number does not meet the technical definition of "random" but, practically speaking, is about as good as random.) Meanwhile, another big pseudorandom number—the accompanying commitment (or co-commitment)—is generated that is associated with a software package that this developer either created or was granted permission to modify.

In order to demonstrate to a prospective user of a particular software package as to who created this version of the package and signed it, the authorized developer would publish a proof that establishes an unequivocal link between the commitment that represents their identity and the commitment attached to the software product. The proof that is carried out is of a special type, called a zero-knowledge proof, which is a way of showing, for instance, that two things have a common bound, without divulging details as to what those things—such as the developer's email address—actually are.

"Speranza ensures that software comes from the correct source without requiring developers to reveal personal information like their email addresses," comments Marina Moore, a Ph.D. candidate at the New York University Center for Cyber Security. "It allows verifiers to see that the same developer signed a package several times without revealing who the developer is or even other packages that they work on. This provides a usability improvement over long-term signing keys, and a privacy benefit over other OIDC-based solutions like Sigstore."

Marcela Mellara, a research scientist in the Security and Privacy

Research group at Intel Labs, agrees. "This approach has the advantage of allowing software consumers to automatically verify that the package they obtain from a Speranza-enabled repository originated from an expected maintainer, and gain trust that the software they are using is authentic."

More information: Kelsey Merrill et al, Speranza: Usable, privacy-friendly software signing, *arXiv* (2023). DOI: 10.48550/arxiv.2305.06463

Provided by Massachusetts Institute of Technology