New computing architecture protects sensitive private data

July 14 2022

Credit: CC0 Public Domain

As our personal data is increasingly used in many applications from advertising to finance to healthcare, protecting sensitive information has become an essential feature for computing architectures. Applications

that process such data must trust the system software they rely on, such as operating systems and hypervisors, but such system software is complex and often has vulnerabilities that can risk data confidentiality and integrity.

Over the past two years, researchers at Columbia Engineering have been working with Arm, a leading semiconductor IP and software design company, to address these vulnerabilities. The team has now unveiled key verification technologies for the Arm Confidential Compute Architecture (Arm CCA), a new feature of the Armv9-A architecture. The paper, presented July 12 at the 16th USENIX Symposium on Operating Systems Design and Implementation (OSDI '22) in Carlsbad, CA, demonstrates the first formal verification of a prototype of Arm CCA firmware.

Arm CCA relies on firmware to manage the hardware to enforce its security guarantees, so it is essential that the firmware is correct and secure. While many previous systems rely on firmware, none of them can guarantee that the firmware has no bugs. Formal verification is a relatively new methodology now being used to guarantee correctness of software/hardware. Rather than testing, formal verification uses mathematical models to prove that the software and hardware are absolutely correct, and thus provides the highest level of guarantee of correctness.

"We've proved, for the first time, that the firmware is correct and secure, resulting in the first demonstration of a confidential computing architecture backed by formally verified firmware," said the study's lead author Xupeng Li, a Ph.D. student of Ronghui Gu, Tang Family Assistant Professor of Computer Science, and Jason Nieh, professor of computer science and co-director of the Software Systems Laboratory.

While there are many approaches to verifying the correctness of simple

programs, they are not suitable for something as complex as CCA firmware, so the researchers had to develop new verification techniques to make verifying Arm CCA firmware possible. For example, CCA firmware is designed for scalability and performance, so it allows highly concurrent operation and mixes C and assembly code together. Concurrent operation is made possible by using fine-grain synchronization methods and code with data races.

It is a design principle of Arm CCA that untrusted software must retain control of managing hardware resources, so a key challenge is proving that the system is still secure even though untrusted software can take away hardware resources as it pleases. Previous approaches have not been able to verify programs with such properties. This new verification technique is powerful enough to verify concurrent firmware with both C and assembly code.

"Bugs are really hard to find via classic software testing techniques," said Xuheng Li, another Ph.D. student of Nieh and Gu who co-authored the work. "So we showed the importance and value of our formal verification techniques with the end result being the first demonstration of a confidential computing architecture backed by verified firmware."

The team is very excited about the new verification technologies that can be used to prove the correctness of implementations of firmware underlying Arm CCA. Arm CPUs are already deployed across billions of devices around the world. As Arm CCA becomes more commonly used to protect user's private data—especially in cloud services and beyond—the verification techniques demonstrated in this paper will provide a significant improvement in data protection and security.

One of the challenges with formal methodologies applied to software is the need to adapt proofs when software is updated. The researchers are working on new technologies to help them incrementally and quickly

verify updates to Arm CCA firmware and ensure that the latest firmware available is always verified.

Gu and Nieh added that they "see the power and potential of formal verification from our work, and we're convinced that formal verification is an essential technique that will, in the near future, supplant the software testing in current use."

More information: Paper: www.usenix.org/conference/osdi22/presentation/li

Provided by Columbia University School of Engineering and Applied Science