NaCl to give way to RockSalt: Computer scientists develop a tool to improve software fault isolation

July 20 2012

A team led by Harvard computer scientists, including two undergraduate students, has developed a new tool that could lead to increased security and enhanced performance for commonly used web and mobile applications.

Called RockSalt, the clever bit of code can verify that native computer programming languages comply with a particular security policy.

Presented at the ACM Conference on Programming Language Design and Implementation (PLDI) in Beijing, in June, RockSalt was created by Greg Morrisett, Allen B. Cutting Professor of Computer Science at the Harvard School of Engineering and Applied Sciences (SEAS), two of his undergraduate students Edward Gan '13 and Joseph Tassarotti '13, former postdoctoral fellow Jean-Baptiste Tristan (now at Oracle), and Gang Tan of Lehigh University.

"When a user opens an external application, such as Gmail or Angry Birds, web browsers such as Google Chrome typically run the program's code in an intermediate and safer language such as JavaScript," says Morrisett. "In many cases it would be preferable to run native machine code directly."

The use of native code, especially in an online environment, however, opens up the door to hackers who can exploit vulnerabilities and readily

gain access to other parts of a computer or device. An initial solution to this problem was offered over a decade ago by computer scientists at the University of California, Berkeley, who developed software fault isolation (SFI).

SFI forces native code to "behave" by rewriting machine code to limit itself to functions that fall within particular parameters. This "sandbox process" sets up a contained environment for running native code. A separate "checker" program can then ensure that the executable code adheres to regulations before running the program.

While considered a major breakthrough, the solution was limited to devices using RISC chips, a processor more common in research than in consumer computing. In 2006, Morrisett developed a way to implement SFI on the more popular CISC-based chips, like the Intel x86 processor. The technique was adopted widely. Google modified the routine for Google Chrome, eventually developing it into Google Native Client (or "NaCl").

When bugs and vulnerabilities were found in the checker for NaCl, Google sent out a call to arms. Morrissett once again took on the challenge, turning the problem into an opportunity for his students. The result was RockSalt, an improvement over NaCl, built using Coq, a proof development system.

"We built a simple but incredibly powerful system for proving a hypothesis—so powerful that it's likely to be overlooked. We want to prove that if the checker says 'yes,' the code will indeed respect the sandbox security policy," says Joseph Tassarotti '13, who built and tested a model of the execution of x86 instructions. "We wanted to get a guarantee that there are no bugs in the checker, so we set out to construct a rigorous, machine-checked proof that the checker is correct."

"Our proofs about the correctness of our own tool say that if you run the tool on a program, and it says it's safe to run, then according to the model, this program can only do certain things," Tassarotti adds. "Our proof, however, was only as good as this model. If the model was wrong, then the tool could potentially have an error."

In other words, he explains, think of an analogy in physics. While you might mathematically prove that according to Newton's laws, a moving object will follow a certain trajectory, the proof is only meaningful to the degree that Newton's laws accurately model the world.

"Since the x86 architecture is very complicated, it was essential to test the model by running programs on a real chip, then simulating them with the model, and seeing whether the results matched. I specified the meanings of many of these instructions and developed the testing infrastructure to check for errors in the model," Tassarotti says.

Even more impressively, RockSalt comprises a mere 80 lines of code, as compared to the 600 lines of the original Google native code checker. The new checker is also faster, and, to date, no vulnerabilities have been uncovered. The tool offers tremendous advantages to programmers and users alike, allowing programmers to code in any language, compile it to native executable code, and secure it without going through intermediate languages such as JavaScript, and even to cross back and forth between Java and native code. This allows coders to choose the benefits of multiple languages, such as using one to ensure portability while using others to enhance performance.

"The biggest benefit may be that users can have more peace of mind that a piece of software works as they want it to," says Morrisett. "For users, the impact of such a tool is slightly more tangible; it allows users to safely run, for example, games, in a web browser without the painfully slow speeds that translated code traditionally provides."

Previous efforts to develop a robust, error-free checker have resulted in some success, but RockSalt has the potential to be scaled to software widely used by the general public. The researchers expect that their tool might end up being adopted and integrated into future versions of common web browsers. Morrisett and his team also have plans to adapt the tool for use in a broader variety of processors.

Reflecting on how the class project has been transformative, Tassarotti says, "I plan to pursue a Ph.D. in computer science, and I hope to work on projects like this that can improve the correctness of software. As computers are so prevalent now in fields like avionics and medical devices, I believe that this type of research is essential to ensure safety."

Provided by Harvard University