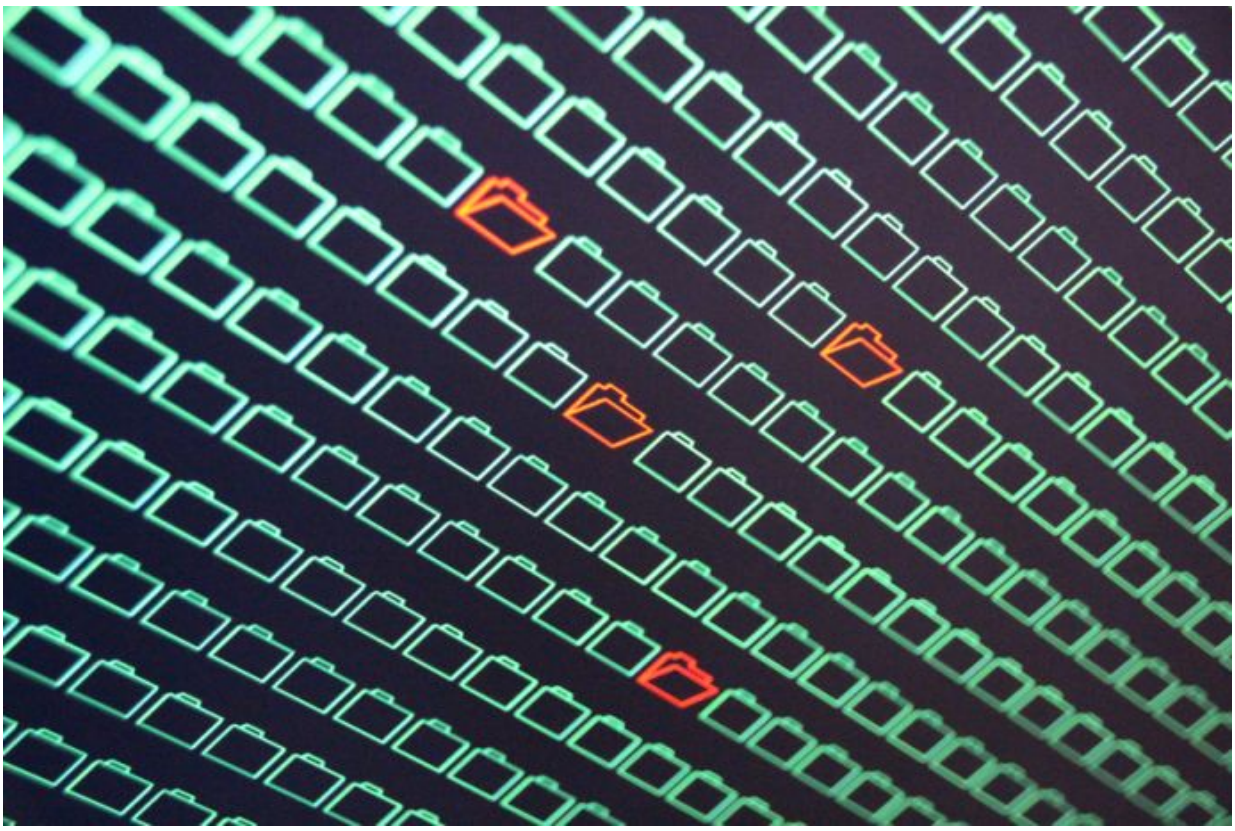
Researchers improve the process of finding vulnerabilities by intentionally adding swarms of bugs to source code

July 7 2016



Credit: NYU Tandon School of Engineering

Individuals and corporations spend millions of dollars every year on software that sniffs out potentially dangerous bugs in computer

programs. And whether the software finds 10 bugs or 100, there is no way determine how many go unnoticed, nor to measure the efficacy of bug-finding tools.

Researchers at the New York University Tandon School of Engineering, in collaboration with the MIT Lincoln Laboratory and Northeastern University, are taking an unorthodox approach to tackling this problem: Instead of finding and remediating bugs, they're adding them by the hundreds of thousands.

Brendan Dolan-Gavitt, an assistant professor of computer science and engineering at NYU Tandon, is a co-creator of LAVA, or Large-Scale Automated Vulnerability Addition, a technique of intentionally adding vulnerabilities to a program's source code to test the limits of bug-finding tools and ultimately help developers improve them. In experiments using LAVA, they showed that many popular bug finders detect merely 2 percent of vulnerabilities.

A paper detailing the research was presented at the IEEE Symposium on Security and Privacy and was published in the conference proceedings. Technical staff members of the MIT Lincoln Laboratory led the technical research: Patrick Hulin, Tim Leek, Frederick Ulrich, and Ryan Whelan. Collaborators from Northeastern University are Engin Kirda, professor of computer and information science; Wil Robertson, assistant professor of computer and information science; and doctoral student Andrea Mambretti.

Dolan-Gavitt explained that the efficacy of bug-finding programs is based on two metrics: the false positive rate and the false negative rate, both of which are notoriously difficult to calculate. It is not unusual for a program to detect a bug that later proves not to be there—a false positive—and to miss vulnerabilities that are actually present—a false negative. Without knowing the total number of real bugs, there is no way

to gauge how well these tools perform.

"The only way to evaluate a bug finder is to control the number of bugs in a program, which is exactly what we do with LAVA," said Dolan-Gavitt. The automated system inserts known quantities of novel vulnerabilities that are synthetic yet possess many of the same attributes as computer bugs in the wild. Dolan-Gavitt and his colleagues dodged the typical five-figure price tag for manual, custom-designed vulnerabilities and instead created an automated system that makes judicious edits in real programs' source code.

The result: hundreds of thousands of unstudied, highly realistic vulnerabilities that are inexpensive, span the execution lifetime of a program, are embedded in normal control and data flow, and manifest only for a small fraction of inputs lest they shut the entire program down. The researchers had to create novel bugs, and in significant numbers, in order to have a large enough body to study the strengths and shortcomings of bug-finding software. Previously identified vulnerabilities would easily trip current bug finders, skewing the results.

The team tested existing bug-finding software and found that just 2 percent of bugs created by LAVA were detected. Dolan-Gavitt explained that automated bug identification is an extremely complex task that developers are constantly improving. The researchers will share their results to assist these efforts.

Additionally, the team is planning to launch an open competition this summer to allow developers and other researchers to request a LAVA-bugged version of a piece of software, attempt to find the bugs, and receive a score based on their accuracy.

"There has never been a performance benchmark at this scale in this area, and now we have one," Dolan-Gavitt said. "Developers can

compete for bragging rights on who has the highest success rate in bug-finding, and the programs that will come out of the process could be stronger."

More information: DOI: 10.1109/SP.2016.15 , www.ieee-security.org/TC/SP201 … /papers/0824a110.pdf

Provided by NYU Tandon School of Engineering