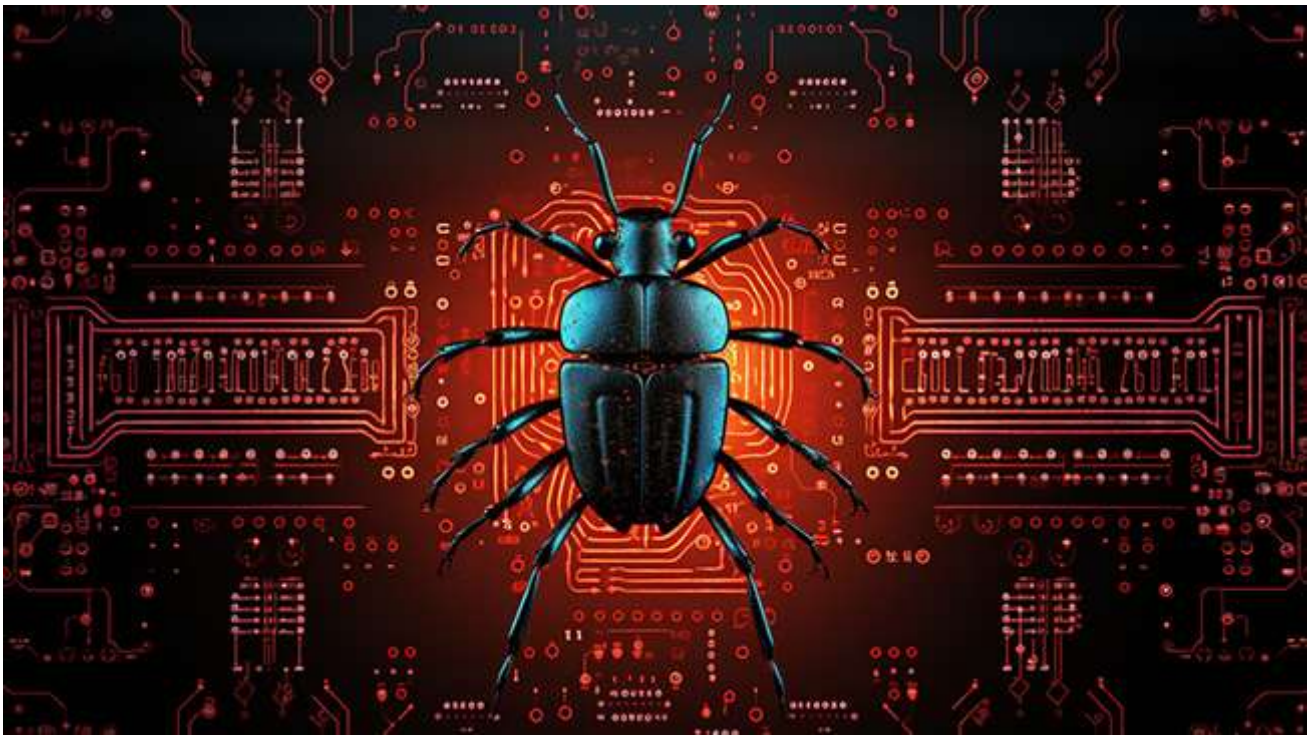
Help Net Security
January 10, 2024

Researchers develop technique to prevent software bugs

A team of computer scientists led by the University of Massachusetts Amherst recently announced a new method for automatically generating whole proofs that can be used to prevent software bugs and verify that the underlying code is correct.



This new method, called Baldur, leverages the artificial intelligence power of LLMs, and, when combined with the tool Thor, yields efficacy of nearly 66%.

“Software bugs have a profound impact on society today. They range from annoying but inconsequential errors that pop up when we’re trying to do work or crash our movie-viewing apps when we’re trying to relax to vulnerabilities that make medical devices susceptible to attacks or that allow hackers to steal personal information, causing billions of dollars in damages in economic growth,” Yuriy Brun, professor in the Manning College of Information and Computer Sciences at UMass Amherst and the paper’s senior author told Help Net Security.

“Reducing bugs in software, or even producing bug-free software, has been a holy grail of systems building for decades, but, unfortunately, the state-of-the-practice in our society is that we expect all software to have bugs. Building bug-free software is just an incredibly difficult challenge.”

Understanding buggy software’s wide reach

The effects of buggy software can range anywhere from the annoying—glitchy formatting or sudden crashes—to potentially catastrophic security breaches or the precision software used for space exploration or controlling healthcare devices.

Of course, there have been methods for checking software for as long as it has existed. One popular method is the simplest: you have a human being go through the code, line by line, manually verifying that there are no errors. Or you can run the code and check it against what you expect it to do. If, for instance, you expect your word-processing software to break the line every time you press the “return” key, but it instead outputs a question mark, then you know something in the code is wrong.

The problem with both methods is that they are prone to human error, and checking against every possible glitch is extraordinarily time-consuming, expensive, and infeasible for anything but trivial systems.

A much more thorough but harder method is to generate a mathematical proof showing that the code does what it is expected to do and then use a theorem prover to ensure the proof is also correct. This method is called machine checking.

However, manually writing these proofs is incredibly time-consuming and requires extensive expertise. “These proofs can be many times longer than the software code itself,” says Emily First, the paper’s lead author who completed this research as part of her doctoral dissertation at UMass Amherst.

With the rise of LLMs, of which ChatGPT is the most famous example, a possible solution is to try to generate such proofs automatically. However, “one of the biggest challenges with LLMs is that they’re not always correct,” says Brun. “Instead of crashing and letting you know something is wrong, they tend to ‘fail silently,’ producing an incorrect answer but presenting it as if it’s correct. And, often, the worst thing you can do is to fail silently.”

This is where Baldur comes in.

The power of Baldur

Baldur took several months to build. The work was done as a collaboration with Google, and built on top of a significant amount of prior research.

First, whose team performed its work at Google, used Minerva, an LLM trained on a large corpus of natural-language text, and then fine-tuned it on 118GB of mathematical scientific papers and webpages containing mathematical expressions. Next, she further fine-tuned the LLM on a language, called Isabelle/HOL, in which the mathematical proofs are written. Baldur then generated an entire proof and worked in tandem with the theorem prover to check its work. When the theorem prover caught an error, it fed the proof, as well as information about the error, back into the LLM, so that it can learn from its mistake and generate a new and hopefully error-free proof.

This process yields a remarkable increase in accuracy. The tool for automatically generating proofs is called Thor, which can generate proofs 57% of the time. When Baldur (Thor's brother, according to Norse mythology) is paired with Thor, the two can generate proofs 65.7% of the time.

Though there is still a large degree of error, Baldur is by far the most effective and efficient way yet devised to verify software correctness, and as the capabilities of AI are increasingly extended and refined, so should Baldur's effectiveness grow.

"Formal verification is one very promising method for building bug-free software. Engineers still build the software system, but with it, they build mathematical proofs that the software is correct. These proofs are quite hard to write, but there are ample tools that support this process, including specialized languages and theorem provers that then take the proof and machine-check it against the software to verify that the software is correct," commented Brun.

"Our work focuses on trying to automate the writing of these proofs. Baldur uses large language models to, given a mathematical theorem, automatically generate a proof of that theorem that a theorem prover can then verify. One benchmark, our method, combined with prior methods, generates proofs fully automatically 65.7% of the time, which is quite promising and would save engineers significant manual effort in writing these proofs," Brun concluded.

In addition to First and Brun, the team includes Markus Rabe, who was employed by Google at the time, and Talia Ringer, an assistant professor at the University of Illinois—Urbana Champaign. This work was performed at Google and supported by the Defense Advanced Research Projects Agency and the National Science Foundation.

More about

artificial intelligence code Google human error software