Defending smart systems on the machine learning framework level

March 5 2021

Credit: CC0 Public Domain

While smart cities and smart homes have become mainstream buzzwords, few people outside the IT and machine learning communities know about TensorFlow, PyTorch, or Theano. These are the open-source machine learning (ML) frameworks on which smart systems are built to integrate Internet of Things (IoT) devices among other things.

ML algorithms and code are often found in publically available

repositories, or data stores, that draw heavily on the aforementioned frameworks. In a December 2019 analysis of code hosting site GitHub, SMU Professor of Information Systems David Lo found over 46,000 repositories that were dependent on TensorFlow, and over 15,000 used PyTorch. Because of these frameworks' popularity, any vulnerability in them can be exposed to cause widespread damage.

"ML frameworks are used to create and run trained ML models," explains Professor Lo, who is also the Director of SMU's Research Lab for Intelligent Software Engineering. "A vulnerability in ML frameworks can potentially translate to vulnerabilities in all trained ML models that are created by it or run on top of it. As an analogy, one can imagine a vulnerability in an operating system; this vulnerability can impact all software that is run on top of the operating system."

Uncovering framework vulnerabilities

Professor Lo's latest research project titled "Uncovering Vulnerabilities in Machine Learning Frameworks via Software Composition Analysis and Directed Grammar-Based Fuzzing"seeks to address that issue. The project, which is supported by the National Research Foundation Singapore, under the National University of Singapore's National Satellite of Excellence in Trustworthy Software Systems, seeks to fill a gap in identifying vulnerabilities in ML frameworks.

Research in this field is often focused on trained ML models rather than the underlying frameworks, and existing research on ML frameworks deal more with bugs (when a system does not behave the way it should) rather than vulnerabilities (that can be exploited). But what is Software Composition Analysis and Directed Grammar-Based Fuzzing? And how does it uncover vulnerabilities?

"Software Composition Analysis (SCA) refers to a toolset that identifies

software vulnerabilities in a target project by analyzing its dependencies," writes Professor Lo. A direct 'dependency' is when a software system uses a third-party library—"a collection of source code that is developed by a third-party, e.g., a software foundation, and shared for reuse to many people"—which may, in turn, be dependent on another third party library. For such a case, the software has one direct dependency and one transitive dependency.

"SCA automatically reports vulnerabilities in the open-source libraries that an application directly or transitively dependent on, so that they can be addressed by the application's developers," explains Professor Lo. "Some of the warnings highlighted by an SCA tool can be false positives. These false positives can correspond to a piece of vulnerable code that can never be executed, and thus is non-exploitable.

"Directed Grammar-Based Fuzzing is a method to generate valid test cases (following predefined grammars) and drive test executions to vulnerable code. If such test executions can be derived, it will provide evidence that a highlighted vulnerability is executable and likely to be exploitable by attackers."

Effectively, Professor Lo's project seeks to differentiate between vulnerabilities that cannot be used by hackers to attack a software system (false positives) from those that can be. In that way, cybersecurity experts can avoid wasting time on the false positives and instead focus on fixing vulnerabilities that are executable. Additionally, it seeks to "automatically identify vulnerable dependencies in a project so that developers can continue using third-party libraries with peace of mind".

Creating success

With the active participation of industry partner Veracode, the project will seek to develop deep learning solutions to uncover vulnerability

repair activities in open source projects. In doing so, the versions of the third-party libraries in which vulnerabilities exist can be identified. Also, they plan to link third-party libraries to entries in databases such as the U.S. National Vulnerability Database (NVD). Overall, Professor Lo and his collaborators aim to better curate the list of ML framework vulnerabilities and develop a more effective SCA process.

Professor Lo elaborates: "If we know that a version N of a third-party library includes fixes to a vulnerability, we can then identify that version N-1 contains the vulnerability (that remains unfixed). We can then use this information to identify all code that makes use of version N-1, warning developers of this potential security issue."

The project, which will run for two and a half years, is one of four externally-funded projects that Professor Lo is currently working on as a Principal Investigator. What is his secret for successfully securing research grants?

"I find it very helpful to keep myself up-to-date about various grant opportunities and align my research to those opportunities," he observes. "Good collaborations with colleagues at SMU, industry, and other universities in Singapore and beyond are also important.

"Also, as the saying goes: 'The road to success is paved with failure.' I learn from the unsuccessful proposals that I wrote, which helps me in writing successful ones."

Provided by Singapore Managment University

Citation: Defending smart systems on the machine learning framework level (2021, March 5) retrieved 18 April 2024 from https://techxplore.com/news/2021-03-defending-smart-machine-framework.html

This document is subject to copyright. Apart from any fair dealing for the purpose of private study or research, no part may be reproduced without the written permission. The content is provided for information purposes only.