Apr 1st, 11:00 AM - 2:30 PM

Lightweight Formal Methods for Improving Software Security

Andrew Berns
University of Northern Iowa

James Curbow
University of Northern Iowa

See next page for additional authors

Author
Andrew Berns, James Curbow, Joshua Hilliard, Sheriff Jorkeh, and Miho Sanders

Lightweight Formal Methods for Improving Software Security

University of Northern Iowa

Dr. Andrew Berns, James Curbow, Joshua Hilliard, Sheriff Jorkeh, Miho Sanders

Introduction

Software plays an increasing role in managing the responsibilities of many areas in society. As we've continued to cede more responsibility to software, the potential damage done from insecure software has grown. It is not hard to find examples of security breaches that have resulted in major financial losses and personal hardships for consumers.

While formal methods have gained popularity in safety-critical systems, they are much less common in other categories of software. Formal methods can help create better software, however, which would also be more secure. "Half of cyber vulnerabilities are software defects and the cost of avoiding and mitigating software errors approaches $100B annually." [1] Our project hopes to help bring formal methods to more types of software.

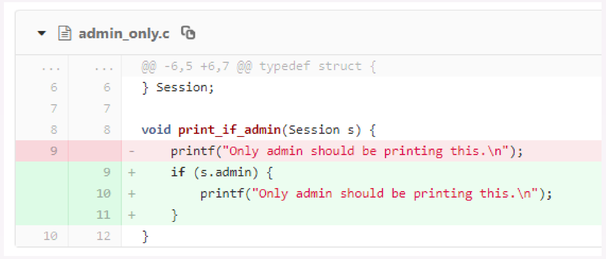
Research Goals

- Measure the effectiveness of modern static analysis tools at identifying, with programmer-defined annotations, security vulnerabilities.
- Identify a set of best practices for developers to follow when using formal specifications while programming or retrofitting existing programs.

Methods

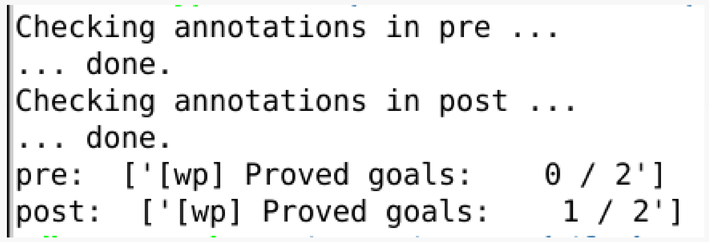
- **Step 1:** Select a vulnerability in an open-source product.

- **Step 2:** Identify the incorrect code and the corresponding fix.



- **Step 3:** Create annotations which might identify the error.

- **Step 4:** Test which annotations, if any, actually identify the error.



Lessons Learned

Retrofitting is still hard.
While the tools for static analysis with formal methods have improved in the past few years, it is still not easy to take a project which has not used formal methods and retrofit the code to work with current tools. This is often given as one of the main reasons why formal methods have not gained widespread acceptance for cybersecurity [2].

More annotations would be helpful.
Today's static analysis tools can only check a subset of possible operations. Some of the annotations that have yet to be implemented, such as whether or not a particular variable is assigned a value inside a method, have only limited support. If these annotations were checkable, it might improve the success rate for detecting vulnerabilities.

Future Work

We are continuing to add to our dataset of software vulnerability corrections and annotations for these corrections.

As the data set grows, we will be better able to identify the best practices for using formal methods to improve computer security.

References

[1] Schaffer, K., & Voas, J. (2016). What Happened to Formal Methods for Security? *Computer, 49*(8), 70-79. doi:10.1109/mc.2016.228

[2] Chong, S., Guttman, J., Datta, A., Myers, A., Pierce, B., Schaumont, P., . . . Zeldovich, N. (2016, August 1). Report on the NSF Workshop on Formal Methods for Security. Retrieved from http://dl.acm.org/citation.cfm?id=3040225