Secure Coding Practices:
How modern languages excel, fail and a comparative view of mature languages to modern languages in security.

Jay Rovacsek c3146220@uon.edu.au

October 4, 2018

Contents

0.1 Preface

0.1.1 Running with Scissors

Admittedly, the title[1] for this section is very much thanks to one of the first items I read while creating this document. The analogy for development in terms of security could not be more apt for a large portion of the development community.

Why cover this topic? As a security enthusiast and now developer, I often found myself looking at a system left untouched until absolutely required, the design choices, logic and knowledge of the language it was written in left with the author. Commonly a requirement for a hotfix was/is needed in a number of this circumstances as a number of critical business services and resources may rely on the system in question.

For clarity, I should note that most of these systems only date to the VB6/VB.NET era, suggesting and age of roughly 15-20 years at oldest. I look forward to the day of encountering some fortran, cobol, pascal or c into the future.

Knowing some of the most critical systems in the world run on these languages as a base gives cause for concern, but is our concern well founded and have modern languages developed enough to allow us to be comfortable into the future?

0.1.2 Tripping Over

We can certainly critique early languages for the level of access to the machine they allow a user, without careful consideration in design and a well founded knowledge in the language used issues notorious of early languages. However, in this day and age of highly abstracted languages and frameworks have we traded old demons for new, or do we really have more safety in our computing goals?

0.2 The Most Dangerous Component of Computing

The apparent and dangerous component of computing is never the computer. Users are terrible, will break anything and everything possible and every human is a user. Seasoned veterans of technology or never used a piece of electronics, every human is amazingly fallible in comparison to the machine they attempt to drive.

To avoid heading down the tangent of human computer interaction intricacies, the coverage of human issues in modern and mature languages must be covered.

0.3 Modern Language Issues

0.4 Mature Language Issues

Bibliography

[1] R. C. Seacord, *Secure Coding in C and C++*. Pearson Education, 2005.