*Proceedings*

# 8th Layered Assurance Workshop

Layered
Assurance
Workshop

## LAW 2014

*Proceedings*

# 8th Layered Assurance Workshop

New Orleans, Louisiana, USA
8–9 December 2014

**Sponsored by**
Applied Computer Security Associates

# Empirical Evaluation of API Usability and Security[*]

Sam Weber, Robert Seacord,
Forrest Shull, David Keaton[†]
Software Engineering Institute

Brad Myers, Michael Coblenz[‡]
CMU

## ABSTRACT

The aim of our project is to gather empirical evidence on the security impacts of language and Application Program Interface (API) design. Ultimately, the cause of cybersecurity failures is flawed code written by programmers. Our philosophy is that programmers are people, and we need to study how to design APIs which are usable by programmers — APIs with which it is easy to develop secure code.

It is well-known that API design can have a large impact on security, and this barrier is difficult, if not impossible, to overcome by training alone. For example, buffer overflows were understood and documented as early as 1972, but are still one of the most common vulnerabilities. Furthermore, APIs are typically designed by a small number of experienced developers but have an extremely long life-span, and therefore the impact of poor API design can have far reaching consequences.

There has been some previous work on the usability of APIs, but so far this work has restricted itself to other software quality attributes, such as learnability. A relevant example of such work is Stylos et al [1], which studied the relative usability of different styles of constructing objects. The results were rather dismaying from a security point-of-view: programmers strongly preferred a style which would cause contructed objects to be mutable, whereas the security community generally considers mutability a source of security problems. One of our tasks will be to investigate and measure this apparent trade-off between traditional usability and security.

We should make clear that we are not targeting just APIs with security-relevant functionality, such as libraries that support authentication. Ordinary libraries — including but not limited to string, file, and XML processing and network libraries — pose more interesting problems because programmers using them are not actively considering security and are consequently more likely to be susceptible to the misconceptions, unstated assumptions, and flawed usage patterns that underlie most vulnerabilities.

API design is a broad domain to research, so we are focusing on a few select areas that research has shown to have security implications such as concurrency and design patterns like immutability. We expect that usability and security will be aligned with respect to concurrency APIs (that is, more usable APIs will also be more secure), but as mentioned above, they will be opposed with respect to mutability. Our research methodology relies on a mix of corpus review (to understand how these issues are dealt with in contemporary code bases) and studies with human subjects under more controlled conditions.

We are extracting typical design patterns by which development teams deal with concurrency API calls by analyzing code repositories that use concurrency. We are augmenting these corpus reviews with field observations and surveys using contextual inquiries of professionals, and surveying alternate concurrency standards and approaches. From this information, we will identify specific hypotheses about the usability differences between alternate styles of presenting concurrency to programmers. This will be used to design and conduct user-studies.

We will use two different populations, students and experts, with a balanced within-subjects design to control for individual programmer differences as well as ordering effects. The evaluation will include quantitative measures (such as number of errors, code length, completion time) and qualitative measures (such as rationales for design decisions) collected through a think-aloud protocol and questionnaires. This task will produce data as to the types and frequency of errors that programmers can be expected to make using alternate concurrency APIs.

In parallel, we will also conduct programmer studies of the impact of mutability on both security and usability.

The ultimate aim of our project is to produce experimentally-validated and specific guidance to API designers on how to create systems which are less prone to security vulnerabilities.

[*]This work is partially supported by NSF award 1423054
[†]email: {samweber|rcs|dmk}@cert.org, fjshull@sei.cmu.edu
[‡]email: {bam|mcoblenz}@cs.cmu.com

## 1. REFERENCES

[1] J. Stylos and S. Clarke. Usability Implications of Requiring Parameters in Objects' Constructors. *29th International Conference on Software Engineering (ICSE'07)*, pages 529–539, May 2007.