Security defects due to implementation and interface dependencies across multiple source code files are difficult and expensive to find and fix. We are evaluating the efficacy of using architectural modular analysis tools to identify security defects and the effect of refactoring on removing security defects.

Our project’s goal is to use automated architecture analysis to identify, prevent, and mitigate security flaws in code. We are retrospectively analyzing open source software, with revision history and issue lists that include identified security flaws. With this data, we are identifying correlations and building models between the relationship of architectural flaws and security flaws. Future work could use this approach to identify areas of code to refactor for architectural and security improvements.

Statistical Analysis. We evaluated the correlations of the existence of security flaws with the existence of different architectural flaws. We also evaluated the effect of refactoring for reducing security flaws, based on the existence of security flaws before and after major refactoring.

Our analysis has found that some refactoring has significantly reduced security flaws, and that security flaws are commonly present with some architectural flaw types.

Effort (LoC)

Security Issues

Potential Impact: ~50% of total effort (LoC) to fix security issues came from fixing <10% of the security issues in Chromium.

Tools

- Source code
- Revision history
- Issue list

Architectural flaws and anti-patterns

- Patterns
- Guidance
- Security likelihood
- ROI for fixes

Analysis: correlations between architecture and security flaws

Predictive models for security fixes via architectural refactoring

Used in build pipeline as part of SDLC, and potentially automated

- Regular reports on architecture flaws introduced
- Indicators of security risk
- Suggestions for refactoring

Process for adding statistical analysis of security flaws and architectural flaws to predict yet-to-be identified security flaws, and provide confidence and a security-ROI for refactoring.