CT-ER: On the Use of Security Metrics to Identify and Rank the Risk of Vulnerability- and Attack-prone Components

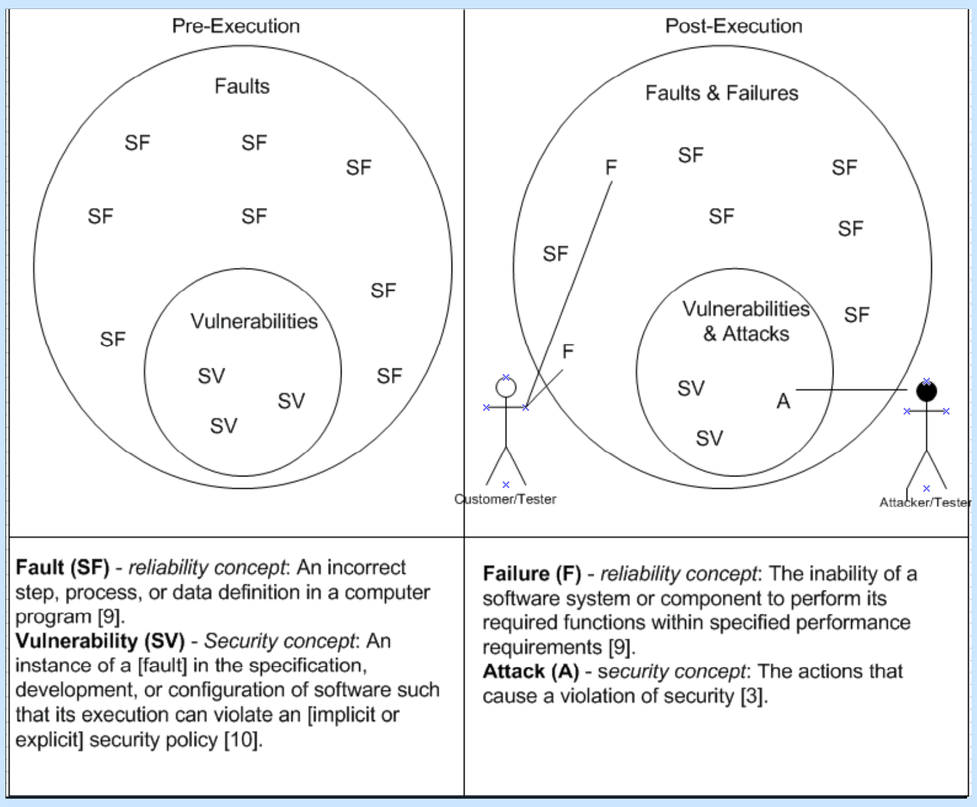
Laurie Williams, http://collaboration.csc.ncsu.edu/laurie/

NSF Grant 0716176

Problem

Late and uninformed security efforts are costly and less likely to be effective than efforts that start early in the software life cycle. Predictive models can provide early warnings of where vulnerabilities and attacks will most likely reside in a software system.

Extensive research has shown that software metrics can predict fault- and failure-prone components early in the software life cycle. This research parallels the reliability work in the security realm by using metrics to predict vulnerability- and attack-prone components.



Fault (SF) - *reliability concept*: An incorrect step, process, or data definition in a computer program [9].
Vulnerability (SV) - *Security concept*: An instance of a [fault] in the specification, development, or configuration of software such that its execution can violate an [implicit or explicit] security policy [10].

Failure (F) - *reliability concept*: The inability of a software system or component to perform its required functions within specified performance requirements [9].
Attack (A) - s*ecurity concept*: The actions that cause a violation of security [3].

Industrial application

Guide security experts to components that are most likely to have security problems.

Approach and Impact

New approach	Research Impact
• Vulnerability- and attack-prone component prediction	• Informed risk management
• Internal and external metrics	• Test case prioritization
• Statistical models	• Early security efforts

Technical description

We performed a case study on a large commercial telecommunications software system that had been deployed to the field for two years. The candidate metrics, code churn, count of source lines of code, and static analysis tool warnings, were chosen as candidates to predict vulnerabilities identified by late-cycle pre-release system testing and potential vulnerabilities reported by customers. The metrics can be obtained early in the software life cycle. We used logistic regression and classification and regression trees as the statistical techniques in our models. The predictive models can distinguish between attack-prone and non attack-prone components.

We are currently working on various code metrics to identify the characteristics of code that differentiates non-fault-, fault- and vulnerability-prone area in source code. This characterization helps to mine vulnerabilities from bug reports or fault prediction results.

Results

- Attack-prone components most likely to have high code churn and large count of static analysis tool warnings.
- Exploitable vulnerabilities isolated to 40% of the system components.

False positive rate 8% False negative rate 0%