OWASP ASIDE Project Roadmap

Development Team:

Jing Xie (jxie2@uncc.edu), Bill Chu (billchu@uncc.edu), John Melton (jtmelton@gmail.com)

Goal:

ASIDE is an abbreviation for Assured Software Integrated Development Environment. It is an Eclipse™ Plugin which is a software tool primarily designed to help students write more secure code by detecting and identifying potentially vulnerable code and providing informative fixes during the construction of programs in IDEs. ASIDE may be useful by professional developers as well.

Design Rationales:

1. From Human Computer Interaction standpoint of view, one of the best practices for designing usable tools is to ease users’ cognitive burden by recognition instead of recalling. Visual cueing is one such instance. It facilitates subjects’ perception of information by providing visualized symbols such as icons, dialogs, markers and etc in the computer world.

2. Some of the vulnerabilities reside in code can be fixed by adding/modifying a few lines of code locally.

3. Program authors have the best understanding of the program. It is easiest to write secure code while their minds are actively focused on program constructing than patching it up later in the development cycle.

Envisioned Benefits:

1. Reinforce secure programming concepts, techniques such as white-list based input validation, output encoding and use of proven library such as OWASP ESAPI. Programmers will be constantly reminded whenever a potential vulnerability is detected, which shapes a mindset that makes security a first class concern throughout the development process.

2. Secure programming knowledge and standards can be shared and enforced to ensure programmers are following best practices.

3. Rules associated with validation code generated by such tools can be written and baked into static analysis tools to help with accuracy of static analysis by reducing false positives.

4. Through facilities such as logs and annotations which can record how security considerations have been addressed, programmers can provide the necessary information needed for security auditing/ assurance/course grading.

Design Descriptions:

1. Rule based specification of sources of untrusted inputs which we refer as trust boundaries. Rules which we refer to as heuristic rules are defined in XML format currently. The two major types of trust boundaries are: Method invocations (e.g. HttpServletRequest.getParameter(String string)) and Parameter inputs (e.g. the arguments of Java program entrance method main(String[] args)).
2. ASIDE recognizes untrusted inputs, based on specified rules, and prompts developers to select from a library of options for either input validation or output encoding. Such library contains validation rules and encoding mechanisms contributed to and shared by many developers in the same way as an open source project.

3. The validation specifications can include both syntactic as well as semantic validations.

**Prototype Demonstration:**

![Prototype demonstration image](image-url)

Figure 1. ASIDE detects and identifies code that imports untrusted inputs through HttpServletRequest.getParameter(String param) method.
Figure 2. ASIDE prompts potential fixes when the cursor hovers on the marked code.
Advanced Features:
Design of advanced features to address more subtle vulnerabilities such as broken access control and CSRF are still under development.

Milestones:
We will strive for releasing our Alpha version in mid 2011.