Nanako Chung

Brendan Dolan-Gavitt

Application Security

October 27th, 2019

Assignment 2 – Report

* Design decisions for the Web service (see below)

One of the most secure ways to build a Web service is through Python’s flask module. It automatically protects against XSS. Additionally, as the OWASP Cross Site Scripting Prevention sheet mentions, I did not put untrusted data except in allowed locations.

* How you mitigated different categories of common Web vulnerabilities (see below)

Vulnerabilities found

|  |  |  |  |
| --- | --- | --- | --- |
| **Vulnerabilities found:** | **Why they occurred:** | **Why I didn’t catch them:** | **How I patched them:** |
| String in ‘value’ attribute of input tags may contain untrusted data | The value attribute may be hijacked and could lead to potential XSS attack | Assumed that the value attribute always had safe data | Aggressive HTML entity encoding |
| String (i.e. URL) in HREF attribute may be untrusted | The HREF attribute may be hijacked to contain unsafe Strings (i.e. URLs) and could lead to potential XSS attack | Assumed that in my code, the url\_for() function would always work accordingly | URL validation/safe URL verification, avoid opening new window |
| HTML tags/body that are untrusted | Untrusted HTML can be placed anywhere in HTML code which could lead to potential XSS attack | It went over my head that not only Strings but HTML could just be inserted into code | HTML validation or sanitizer |