Towards a Pragmatic XSS Defense Framework

Joseph Bugeja
Today’s Threat Landscape

- Organizations nowadays do several things right as far as security goes
  - Firewalls and other perimeter devices are deployed
  - Servers are regularly patched
  - Network traffic is encrypted
  - Continuous monitoring via security audits and network scanning tools

- However security vulnerabilities present at the application layer (code level) are usually ignored.
Web Application Vulnerability Trend

Cross-site scripting is the “most prevalent and pernicious” Web application security vulnerability - OWASP
Cross-Site Scripting (XSS)

- XSS flaws occur whenever an application takes untrusted data and sends it to a Web browser without proper validation and encoding.
- The untrusted data, typically consisting of JavaScript content, changes the browser execution context from a passive to an active context.
- It allows attackers to execute scripts in the victim’s browser potentially impacting confidentiality, integrity and availability.
- Websites from NASA.gov, FBI.gov, CNN.com, Ebay, Yahoo, Microsoft, Google and many more all were XSSed!
XSS Attack Example

HTML code:
<pre>&lt;p&gt;No results were found for the query:&lt;br /&gt;&lt;br /&gt;&lt;span id="lblSearch">xss&lt;/span&gt;&lt;/p&gt;</pre>
XSS Attack Example

HTML code:

```html
<p>No results were found for the query: <br /><br />
<span id="lblSearch"><script>alert(document.cookie)</script></span></p>
```
XSS Attack Example

**Cookie Theft:**

```html
<script>
</script>
```
Attacker Strategies

- Availability and ease-of-use of third-party tools.
- Various enabling technologies such as JavaScript, VBScript and CSS.
- Techniques such as encoding, code obfuscation and URI shorteners are often utilized to hide away malicious XSS payloads.
- Subtle browser parsing quirks.
- Social engineering tactics.
Defense Strategies

- Client-side vs Server-side vs Hybrid.
- Development vs Operational Time.
- False Positive Rates (FPR) / False Negative Rates (FNR).
- Boilerplate reaction phase.
- Legacy/Closed-source applications.

=> Most of the evaluated tools are NOT practical!
Anti-XSS Framework Principles

- **Centralized Design**
  - Single central *choke point*

- **Hybrid Server-Based Security Model**
  - Inner Core features a +ve security model surrounded by additional optional outer cores

- **Rule-Action Based Approach**
  - Fine-grained *grammar* allowing apps to react in different ways according to context

- **Secure-By-Default**
  - Validate/Encode *all* HTTP parameters

- **Simplistic API**
  - *Easy* to use and extend
Anti-XSS Framework Architecture

- HTTP Client
- Internet/Intranet/Extranet
- Web Server
- XSSD-Proxy
- Validation-Manager
- Rule-Engine
- Knowledge-Base

Anti-XSS Engine

Use

Errors/Response

Validate

Request

Response

HTTP-Header
Anti-XSS Framework in Action

The Internet

Firewall

Anti-XSS Framework

Web Server

80

=> Validation + Encoding + ...

Databases
Anti-XSS Framework in Action

Anti-XSS Framework Logs:

[Anti-XSS] VIOLATION: Parameter txtSearch with value `<script>alert(document.cookie)</script>` is classified as malformed by rule `^[a-zA-Z0-9\s.-]+$`

[Anti-XSS] Encoded [Auto] txtSearch from `<script>alert(document.cookie)</script>` to `&lt;script&gt;alert(document.cookie)&lt;/script&gt;`
Conclusion

• XSS is very widespread and it has considerable technical and business impacts.
• Do NOT rely solely on blacklists!
• The proposed **Anti-XSS Framework** offers an effective and pragmatic solution featuring:
  - Ease-of-Deployment/Installation/Customization
  - Browser-Agnostic
  - Real-time and Immediate Protection
  - No Changes/Recompilations Required
  - Performance
  - Accuracy
  - Extensible
Thank You!

Thanks for Listening!

Joseph Bugeja
bugejajoseph@yahoo.com