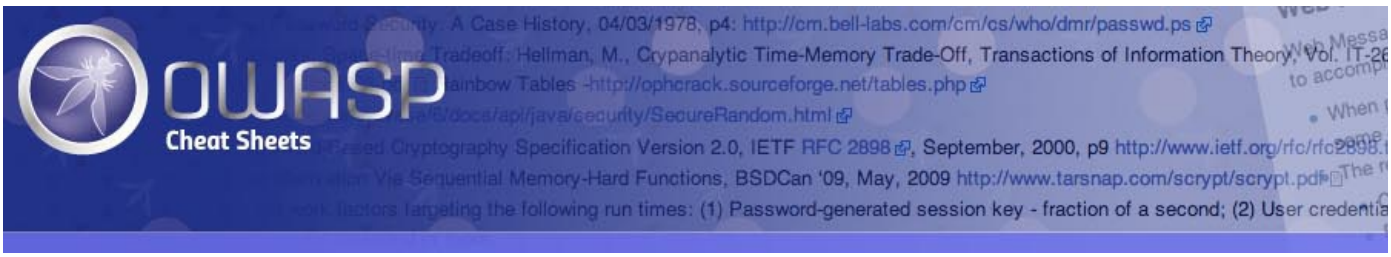
AJAX Security Cheat Sheet



Last revision (mm/dd/yy): **04/7/2018**

Introduction

This document will provide a starting point for AJAX security and will hopefully be updated and expanded reasonably often to provide more detailed information about specific frameworks and technologies.

Client Side (Javascript)

Use .innerText instead of .innerHtml

The use of .innerText will prevent most XSS problems as it will automatically encode the text.

Don't use eval

Eval is evil, never use it. Needing to use eval usually indicates a problem in your design.

Canonicalize data to consumer (read: encode before use)

When using data to build HTML, script, CSS, XML, JSON, etc. make sure you take into account how that data must be presented in a literal sense to keep it's logical meaning. Data should be properly encoded before used in this manner to prevent injection style issues, and to

make sure the logical meaning is preserved.

Check out the OWASP Encoding Project.

Don't rely on client logic for security

Least ye have forgotten the user controls the client side logic. I can use a number of browser plugging to set breakpoints, skip code, change values, etc. Never rely on client logic.

Don't rely on client business logic

Just like the security one, make sure any interesting business rules/logic is duplicated on the server side less a user bypass needed logic and do something silly, or worse, costly.

Avoid writing serialization code

This is hard and even a small mistake can cause large security issues. There are already a lot of frameworks to provide this functionality. Take a look at the JSON page for links.

Avoid building XML or JSON dynamically

Just like building HTML or SQL you will cause XML injection bugs, so stay way from this or at least use an encoding library or safe JSON or XML library to make attributes and element data safe.

- XSS (Cross Site Scripting) Prevention
- SQL Injection Prevention

Never transmit secrets to the client

Anything the client knows the user will also know, so keep all that secret stuff on the server please.

Don't perform encryption in client side code

Use TLS/SSL and encrypt on the server!

Don't perform security impacting logic on client side

This is the overall one that gets me out of trouble in case I missed something :)

Server Side

Use CSRF Protection

- Cross-Site Request Forgery (CSRF) Prevention

Protect against JSON Hijacking for Older Browsers

Review AngularJS JSON Hijacking Defense Mechanism

See "JSON Vulnerability Protection" from https://docs.angularjs.org/api/ng/service/$http

Always return JSON with an Object on the outside

Always have the outside primitive be an object for JSON strings:

Exploitable:

```
[{"object": "inside an array"}]
```

Not exploitable:

```
{"object": "not inside an array"}
```

Also not exploitable:

```
{"result": [{"object": "inside an array"}]}
```

Avoid writing serialization code. Remember ref vs. value types!

Look for an existing library that has been reviewed.

Services can be called by users directly

Even though you only expect your AJAX client side code to call those services the users can too. Make sure you validate inputs and treat them like they are under user control (because they are!).

Avoid building XML or JSON by hand, use the framework

Use the framework and be safe, do it by hand and have security issues.

Use JSON And XML Schema for Webservices

You need to use a 3rd party library to validate web services.

Authors and Primary Editors

Til Mas
Michael Eddington

Other Cheatsheets

V - T - E	Cheat Sheets
Developer / Builder	3rd Party Javascript Management · Access Control · **AJAX Security Cheat Sheet** · Authentication (ES) · Bean Validation Cheat Sheet · Choosing and Using Security Questions · Clickjacking Defense · Credential Stuffing Prevention Cheat Sheet · Cross-Site Request Forgery (CSRF) Prevention · Cryptographic Storage · C-Based Toolchain Hardening · CSS Security · Deserialization · DOM based XSS Prevention · Forgot Password · HTML5 Security · HTTP Strict Transport Security · Injection Prevention Cheat Sheet · Injection Prevention Cheat Sheet in Java · JSON Web Token (JWT) Cheat Sheet for Java · Input Validation · Insecure Direct Object Reference Prevention · JAAS · Key Management · LDAP Injection Prevention · Logging · Mass Assignment Cheat Sheet · .NET Security · OS Command Injection Defense Cheat Sheet · OWASP Top Ten · Password Storage · Pinning · Query Parameterization · REST Security · Ruby on Rails · Session Management · SAML Security · SQL Injection Prevention · Transaction Authorization · Transport Layer Protection · TLS Cipher String Configuration · Unvalidated Redirects and Forwards · User Privacy Protection · Web Service Security · XSS (Cross Site Scripting) Prevention · XML External Entity (XXE) Prevention Cheat Sheet
Assessment / Breaker	Attack Surface Analysis · REST Assessment · Web Application Security Testing · XML Security Cheat Sheet · XSS Filter Evasion
Mobile	Android Testing · IOS Developer · Mobile Jailbreaking
OpSec / Defender	Virtual Patching · Vulnerability Disclosure
Draft and Beta	Application Security Architecture · Business Logic Security · Content Security Policy · Denial of Service Cheat Sheet · Grails Secure Code Review · IOS Application Security Testing · PHP Security · Regular Expression Security Cheatsheet · Secure Coding · Secure SDLC · Threat Modeling
	All Pages In This Category

Categories: Cheatsheets | OWASP AJAX Security Project

Activities
Attacks
Code Snippets
Controls
Glossary
How To...
Java Project
.NET Project
Principles
Technologies
Threat Agents
Vulnerabilities

Tools

What links here
Related changes
Special pages
Printable version
Permanent link
Page information