Mozilla Security Review

General Information

Project Name:	Series WordPress Plugin
Security Reviewer:	Matt Fuller (mfuller)
Date of Review:	7/18/12
Bugzilla Bug #:	775123

Background Information:

The hacks blog would like to have the Series plugin installed on WordPress.

From the plugin page:
Organize Series is a popular (downloaded 57,055 times!) plugin for WordPress that makes creating and managing series of posts in WordPress REALLY easy.

Helpful URLs:

Plugin Site: http://organizeseries.com/

Access Information:
How is the application accessed?
Is the application internal or publicly available?
If it is internal, what mechanism prevents non-members from accessing it?
Are there links to user-only resources displayed to non-users?
Are login pages secured with SSL over HTTPS?

The plugin is accessed through a settings page with the control panel of WP. The public visible interface is seen via a few html snippets inserted onto posts in a series. The internal admin console is handled via WP's authentication.

Infrastructure and Backends:
What languages do the applications use?
What database language is used if applicable?
Are the running versions up to date?
What server is it running on?

PHP, JS, CSS, HTML

All other aspects handled by WP.

Accounts and Passwords:
If the mechanism to prevent general access is a password, how is the signup process handled?
How is account information stored?
Are passwords properly stored within databases if applicable?
Is a password policy in place?

Are accounts locked-out after a number of invalid logins?
Are passwords 8 characters or greater?
Do passwords use both numbers and letters (and possibly symbols)?
Is there a blacklist of common passwords?
Do passwords expire after X days and require a reset?
Are invalid logins logged?
Is there a lockout after X invalid attempts?
Is the error message for lockout generic (does not include if user/pass is valid)?
How are password resets handled (i.e. email, security question, etc.)?
Do emails sent after signup/reset contain a session link? (should not)
Do email verification codes expire after one use or 8 hours?
Is password reuse limited/prevented?

No accounts.

Session Management:
How long are session cookies stored?
Are session tokens 128-bit or greater?
Is session ID token creation handled by the server (or cryptographically if locally)?
Do authenticated sessions timeout after 15 minutes?
Is the Secure Flag enabled for all set cookies?
Is the HTTP-Only flag used to disable malicious script access to the session ID?
Are new session ids created on login?
On logout, are session ids invalidated?

All handled by WP, unrelated to plugin.

Third-Party Resources:
Are third-party resources used (i.e. JavaScript libraries, images, CSS, etc.)?
Can those resources be trusted / are they from reputable sources?
Is there a chance the resource could be compromised?
Is it possible to host the resources locally to mitigate risks?
Is a third-party responsible for storage of user data?
Does the application connect with services like Facebook, Twitter, etc?

No third-party resources are used besides a link to a “donate with Paypal” button loaded from PayPal.

A blog feed is loaded from http://www.organizeseries.com/category/blog/feed/ but it is only pulled via XSS and displayed within the plugin settings page.

Data Handling:
What kind of data is transferred between the user and the application?
Is this data generated by the user or generated automatically?
Can the data be trusted?
How is the data sent to the application (i.e. JSON format, plain-text, GET/POST, etc.)?
How is the data handled by the server/application?
Can the data be manipulated in transit?
What happens if the data is altered?
What is done with the data once it is received (i.e. stored in a database, displayed to users)?
Is any data storage done via cookies? If so, what kind of data is stored via this method?

Blog authors pass post information to the plugin such as title, post x of x, etc. The public facing users never send data to the plugin.

Uploaded Data:
Can the user upload data to the application?
Are extensions, file size, file type (not only based on extension), etc. checked?
Are files renamed upon upload?
Is the correct content-type used?

Only logged-in, blog authors interact with the plugin.

Data Sensitivity:
What kind of data is being stored and/or manipulated by the application?
Does this data need to be encrypted in transit? In storage?
What is the impact if this data is lost/stolen?
Is secure/sensitive data sent over SSL?

Post metadata and information about series - none critical info.

Application Administration:
Is there an administration console?
Can it be accessed publicly?
How is it secured if so?
Are correct methods used to prevent admin actions from being performed outside of the admin console (i.e. using CSRF tokens)?
Are there any configuration pages that should not be made public?

All admin features are protected by the WP login. CSRF tokens are handled by the WP admin console. All config pages for the plugin require a login to access (return blank pages unless served inside admin console).

Security Coding Flaws:
Have all user inputs been sanitized?
Is a maximum size for data (input or uploads) defined?
Do all URL variables pass through sanitization?
Is data from databases escaped properly?
Are CSRF tokens used to prevent POSTs from outside websites?
If a database is used, are protections against SQL injection in place?
Is validation done on the server side (not just client-side)?
Is output encoded in addition to sanitization of input?
Does the user ever send data to the OS level?
Are x-frame options sent to “deny” or “sameorigin”?
Is debug mode disabled?

User data is sanitized unless it’s part of the code (such as “enter the html that should be displayed to users”), however this is only modified by logged-in blog authors.

CSRF and other questions above handled by WP.

Testing:

List all tests performed on the application

1. Check for pages accessible outside admin console (by loading direct wp-content/plugin page).
2. Scan for XSS in plugin via provided parameters
3. Scan for XSS via URL param
4. Can user use %xx% keywords in comments?
5. Check for third-party resources

Results:

List the results of the above tests

1. Return blank page.
2. All input by admin is escaped
3. No URL param’s used in output
4. No - handled as expected
5. See above, no issues

Meetings and Notes: