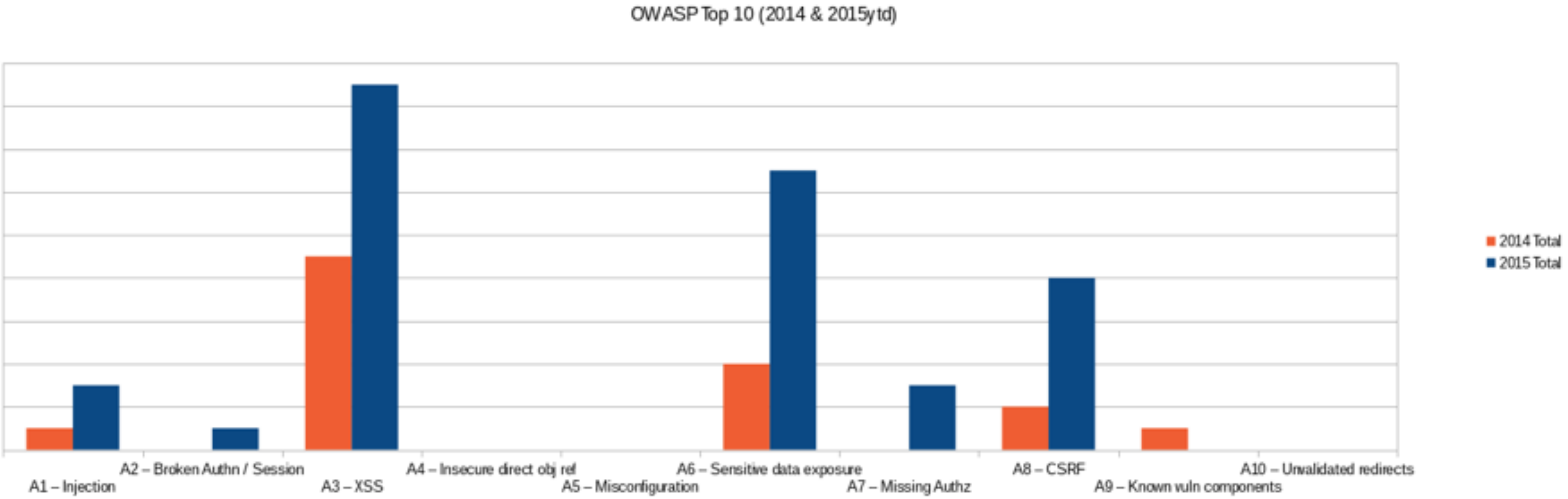
Secure Coding for MediaWiki Developers


OWASP Top 10 (2014 & 2015ytd)
A1 – Injection
A2 – Broken Authn / Session
A3 – XSS
A4 – Insecure direct obj ref
A5 – Misconfiguration
A6 – Sensitive data exposure
A7 – Missing Authz
A8 – CSRF
A9 – Known vuln components
A10 – Unvalidated redirects
2014 Total
2015 Total

Agenda

XSS (Cross-Site Scripting)

CSRF (Cross-Site Request Forgery)

SQL Injection

Private Data Exposure

XSS

(Cross-Site Scripting)

Attacker injects client-side code (JavaScript, HTML, CSS, etc.) in a web page viewed by other users.

Results in attacker hijacking the user’s browser.

XSS

Reflected XSS (1st Order)

```
<input type=“text” value=“<? echo $_GET[‘search_term’]; ?>” />

<!-- http://example.com/foo.php?search_term=“><script>alert(‘XSS’)</script> -->
```

XSS

Stored XSS (2nd Order)

```
<?php
 $articles = $dbr->query(“SELECT id, title FROM ‘articles’”);
 foreach ($articles as $article) {
   echo “<a href=‘read.php?id={$article[‘id’]}’>{$article[‘title’]}</a>”;
 }
?>

/*
 * id = ‘><script>alert(”XSS”);</script>
 * OR
 * title = <script>alert(“XSS”);</script>
 */
```

XSS

DOM XSS (3rd Order)

```
<script>
 document.write(“a href=’#“ + location.hash + “‘>Next Section</a>”);
</script>

<!--
 http://example.com/foo.html#'onClick='alert(“XSS”)
-->
```

XSS

Best Practices

Validate Input, Escape Output

Trust No Input (including cookies, database, DOM content)

Use HTML/XML classes, know which functions escape and which don't

Templating can be affective, the security is demonstrable

Escape as close to the output as possible

In javascript use: createElement(), setAttribute(), appendChild();

Avoid html(), innerhtml(), document.href

Avoid $(“untrusted data”) in jQuery

Remember html parser converts entities

Always keep in mind the DOM context where you are writing out user-controlled data

See “Manual Detection”

XSS

Prevention

Use MediaWiki’s built-in output functions.

```
// Example 1
$attribs = array(
 ‘name’ => ‘wpSourceType’,
 ‘type’ => ‘radio’,
 ‘id’   => $id,
 ‘value’ => $this->mParams[‘upload-type’],
);
if ( !empty( $this->mParams[‘checked’] ) ) {
 $attribs[‘checked’] = ‘checked’;
}
$label .= Html::element( ‘input’, $attribs );
```

XSS

Prevention

Use MediaWiki’s built-in output functions.

```
// Example 2
$out .= Xml::openElement( ‘div’, array( ‘class’ => ‘search-types’ ) );
$out .= Xml::openElement( ‘ul’ );
…
$out .= Xml::closeElement( ‘ul’ );
$out .= Xml::closeElement( ‘div’);
```

XSS

Prevention In HTML Contexts

Body – prevent tag creation

Attribute names – prevent javascript handlers (onMouseover, onClick, etc.)

Quoted attribute values – prevent breaking out of quotes

URL attributes – prevent **`javascript:`** or **`data:`** targets

CSS – normalize, and prevent scripting

JavaScript – don't write out data here

XSS

Manual Detection

Review code and start at output and trace variables back to source, looking for absence of entity escaping.

Test input fields using the following string:

```
``';!--"<XSS>=&lt;{()}}
```

XSS

Prevention

Use appropriate HTTP Headers:

```
Content-type
X-Frame-options: deny
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Security-Policy
```

Implementation of Content Security Policy in core is in progress.

CSRF

(Cross-Site Request Forgery)

Attacker forces unauthorized web requests from a user’s browser.

Abuses web browser submission of cookies with all requests to a domain, even when possibly initiated from another website.

CSRF

(Cross-Site Request Forgery)

```
<!--
Example 1 - Snippet of http://example.com/foo.html
-->
<html>
…
<body>
…
<img src=“http://en.wikipedia.org/wiki/index.php?title=some_thing&action=delete” />
…
</body>
</html>
```

CSRF

(Cross-Site Request Forgery)

```
<!--
Example 2 - Snippet of http://example.com/bar.html
-->
…
<form name=“wikiEdit” method=“POST” target=“hiddenFrame”
  action=“http://en.wikipedia.org/wiki/index.php?title=some_thing&action=submit”
>
<input type=“hidden” name=“wpTextbox1” value=“whatever the attacker wants” />
…
</form>
<iframe name=“hiddenFrame” style=“display: none”></iframe>
<script>
  document.wikiedit.submit();
</script>
…
```

CSRF

Prevention

Add a random token to HTML forms and check the token on form submission.

The HTMLForm class handles this automatically.

CSRF

Prevention

Use Edit Tokens when performing actions that change pages or otherwise execute commands.

```
// When parsing a form
$token = $request->getVal( ‘wpEditToken’ );
$this->mTokenOk = $this->getUser()->matchEditToken( $token );
```

The API modules return true for **`needsToken();`**

SQL Injection

Occurs when attacker is able to pass and execute SQL, rather than just data in a lookup context.

```
$qry = “SELECT user_id, user_password FROM
    user WHERE username=‘$userName’”;
```

SQL Injection

```
$qry = “SELECT user_id, user_password
FROM user WHERE username=‘$userName’”;

// $username = “foo’ OR 1=1”

SELECT user_id, user_password FROM user
WHERE username=‘foo’ OR 1=1
```

SQL Injection

Prevention

Use MediaWiki’s built-in database access functions.

```
$result = $dbw->select(
   ‘user’,                                   // table
   array(‘user_id’, ‘user_password’),        // columns
   array(‘user_name’ => $username),          // where
   clause
   __METHOD__
);
```

Private Data Exposure

Store only the minimum necessary data.

Some data on the wiki needs to be protected for privacy and legal reasons.

For legal compliance, any contributed data must be able to be deleted or suppressed, including:

Usernames, revisions, edit messages, titles, images…

Private Data Exposure

Don’t reimplement the revision deletion/suppression system.

Private Data Exposure

If revision deletion/suppression is reimplemented, then it must be reimplemented **completely:**

- Core
  - archive.ar_deleted
  - filearchive.fa_deleted
  - ipblocks.ipb_deleted
  - logging.log_deleted
  - revision.rev_deleted

- Common WMF Extensions
  - CentralAuth
    - globaluser.gu_hidden
  - Abuse Filter
    - ip/ua of logs (always)
  - abuse_filter.af_hidden
    - abuse_filter_log.afl_deleted
- CheckUser
  - cu_log, cu_changes

It's complex. Don't do it.

Privacy Policy

- Information you provide us or information we collect from you that could be used to personally identify you. To be clear, while we do not necessarily collect all of the following types of information, we consider at least the following to be "personal information" if it is otherwise nonpublic and can be used to identify you:

  - (a) your real name, address, phone number, email address, password, identification number on government-issued ID, IP address, user-agent information, credit card number;

  - (b) when associated with one of the items in subsection (a), any sensitive data such as date of birth, gender, sexual orientation, racial or ethnic origins, marital or familial status, medical conditions or disabilities, political affiliation, and religion; and

  - (c) any of the items in subsections (a) or (b) when associated with your user account.

Need Help?

Open a Phabricator ticket and tag it “Security-Reviews” or
Message dapatrick or csteipp on IRC or
E-mail dpatrick@wikimedia.org or csteipp@wikimedia.org

Ask For Help

Authentication / authorization / session handling

Executing external programs via the shell

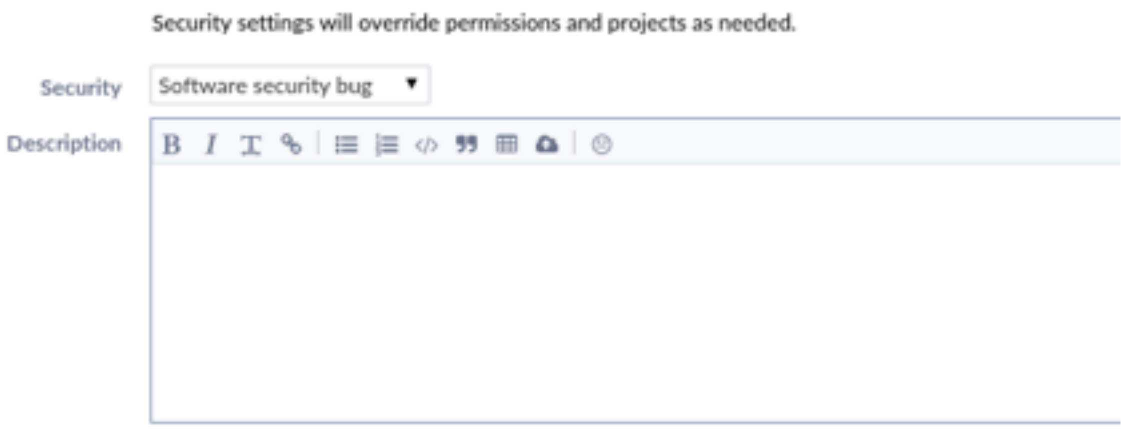
Serving new content types

Encryption and hashing

Disabling page output

Reporting an Issue?

Open a Phabricator ticket and select “Software security issue” from the “Security” dropdown or
E-mail security@wikimedia.org



Security settings will override permissions and projects as needed.

Security: Software security bug ▼

Description

Thanks. Questions?

Further Resources

General

https://www.mediawiki.org/wiki/Manual:Coding_conventions

https://www.mediawiki.org/wiki/Security_checklist_for_developers

CSRF

https://www.mediawiki.org/wiki/Cross-site_request_forgery

https://www.mediawiki.org/wiki/Manual:Edit_token

Revision Deletion/Suppression and Privacy

https://www.mediawiki.org/wiki/Help:RevisionDelete

https://wikimediafoundation.org/wiki/Privacy_policy

SQL Injection

https://www.mediawiki.org/wiki/SQL_injection

https://www.mediawiki.org/wiki/Manual:Database_access

XSS

https://www.mediawiki.org/wiki/Cross-site_scripting