

Drupal GOTCHA'S

Hardening your Drupal Site

David Hazel
www.hazelconsulting.com
dave@hazelconsulting.com
253-686-0296

My Background

- MS CSS - UW Tacoma 2007
- Web developer since 1996
- Entrepreneur since 1994
- Java, Python, C#.net, MSSQL,
- These Days
 - PHP, MySQL
 - Drupal

Hardening

- Only allow what the user needs

Input Filtering

- admin:admin for User #1 bad idea
- Safe use of Input Formats
 - don't allow `<script>`, ``, etc for untrusted users
- Configure Input Formats Drupal 5
- Configure Input Formats Drupal 6

Disable PHP Input for untrusted users

- WHO???

EVERYONE!!!

WHY??

- Google Hacking

Need the Functionality

- turn any custom blocks into modules

Hardening – Access Control

- Role based security
- Fine grained access control
- Rule based access
- **Access Control - Drupal 5**

Writing Secure Code

- `check_plain()` | `t()`
- `filter_xss()` | `filter_xss_admin()`
- `check_markup()`
- `drupal_urlencode()`
- `check_Url()`
- `mime_header_encode()`
- `db_query()`

check_plain() | t()

- Target -> Plain Text
- check_plain()
 - strips all tags
- t()
 - run all text through t() so it can be translated
 - translations aren't checked

-

filter_xss()

- Target -> HTML
- `$allow_tags = array('a', 'em', 'strong', 'p');`
- `filter_xss($string, $allowed_tags)`
- returns a “clean string” with only the white listed tags
- only allows white listed protocols in
 - `filter_allowed_protocols` in `settings.php`

filter_xss_admin()

- Target -> HTML
- users accessing admin pages are more trustworthy
- filter_xss_admin() uses a more liberal list of allowed tags
- all but:
 - `<script>`
 - `<style>`

check_markup()

- Target -> HTML
- check_markup()
 - runs text through filters
-

drupal_urlencode()

- Target -> HTML
- drupal_urlencode()
 - pass escaped characters as part of url
-

db_query()

- Target -> SQL
- db_query()
 - escapes embedded single quotes
- i.e. db_query("insert into {table_1} (nid, uid", %nid, %uid);

db_rewrite_sql()

- Target -> SQL
- db_rewrite_sql()
 - makes sure user has permissions to access content
 - checks permissions by doing a join with the node_access table

Ajax Security

- Users break stuff by accident, Malicious users break stuff on purpose
- Server side Ajax components might be called directly, i.e not from the jQuery code you wrote, so test all access

Using Eval()

- DON'T!!!
- if you must use Eval()
- DON'T!!!

Resources

- These Slides
<http://www.hazelconsulting.com/dcv08>
- <http://drupal.org/security>
- <http://drupal.org/security-team>
- <http://drupal.org/writing-secure-code>