Talk Outline

- What Is a PHP Security Audit?
- Setting the Bar
- Analyzing the Design
- Analyzing the Configuration
- Searching the Source
- More Information
- Questions and Answers
What Is a PHP Security Audit?

- An audit is an **examination**.
- Nothing should be off-limits.
- A PHP security audit is primarily an **examination of the source**.
- Other points of interest are the **design** and **configuration**.
Setting the Bar

- How much security do you need?
- Start with a minimum level.
- At the very least, a PHP application should **filter input** and **escape output**.
What Is Input?

- Some input is obvious - form data ($_GET and $_POST), cookies ($_COOKIE), etc.
- Some input is hard to identify - $_SERVER
- Sometimes it depends on your perspective - $_SESSION, data from databases, etc.
- The key is to identify the origin of data. Data that originates anywhere else is input.
What Is Filtering?

- Filtering is an inspection process.
- **Prove** data to be valid.
- Consider everything else tainted.
- Ensure you can easily and reliably distinguish between **filtered** and **tainted** data.
- I use a strict naming convention.
<?php

$clean = array();

switch($_POST['color'])
{
    case 'red':
    case 'green':
    case 'blue':
        $clean['color'] = $_POST['color'];
        break;
}

?>
<?php

$clean = array();

if (ctype_alnum($_POST['username'])) {
    $clean['username'] = $_POST['username'];
}

?>
What Is Output?

Some output is obvious - HTML, JavaScript, etc.

The client isn't the only remote destination - databases, session data stores, feeds, etc.

The key is to identify the destination of data. Data destined for anywhere else is output.
What Is Escaping?

- Escaping preserves data as it enters another context.
- Some characters need to be represented in a special way:
  - *O’Reilly* (SQL)
  - *AT&T* (HTML)
- In most cases, there is a function you can use.
- If you must write your own, be exhaustive.
<?php

$html = array();

$html['username'] = htmlentities($clean['username'], ENT_QUOTES, 'UTF-8');

echo "<p>Welcome back, {$html['username']}.</p>";

?>
<?php

$mysql = array();

$mysql['username'] = mysql_real_escape_string($clean['username']);

$sql = "SELECT *
            FROM profile
            WHERE username = '{$_POST['username']}'";

$result = mysql_query($sql);
?>
Analyzing the Design

- Have the design explained first.
- Avoid unnecessary complexity.
- Encourage distinction between tainted and filtered data.
Analyzing the Configuration

- Mostly dictated by `php.ini`.
- Also consider `httpd.conf`, `.htaccess`, `ini_set()`.
Analyzing the Configuration

- Things to avoid:
  - register_globals
  - allow_url_fopen
  - magic_quotes_gpc
  - display_errors
Searching the Source

- Identify input and trace it forward.
- Identify output and trace it backward.
- Ensure input is filtered and output is escaped.
Identifying Input

- HTML Forms:
  - form
  - input
  - $_GET
  - $_POST
  - $_REQUEST
Identifying Input

- Databases:
  - mysql_query()
  - SELECT

- HTTP Headers:
  - $_COOKIE
  - $_SERVER
Identifying Output

- Client:
  - echo
  - print
  - `<?=`
Identifying Output

- **Databases:**
  - `mysql_query()`

- **Commands:**
  - `exec()`
  - `passthru()`
  - `system()`
<?php
$action = $_POST['action'];
$query_string = "action=$action";
$link = "index.php?$query_string";
?>

<a href="<?php echo $link; ?>">Click Here</a>
<?php
$username = $_COOKIE['username'];
$greeting = "Welcome back, $username.";
$html = "<p>$greeting</p>";
echo $html;
?>
Gotchas

- Trust of HTTP Headers:
  - Referer

- Trust of \$_SERVER:
  - \$_SERVER['PHP_SELF']

- Trust of Client-Side Restrictions:
  - maxlength
More Information

- PHP Security Consortium
  http://phpsec.org/

- My Business Web Site
  http://brainbulb.com/

- My Personal Web Site and Blog
  http://shiflett.org/
Thanks for Listening!

Chris Shiflett
chris@brainbulb.com