Security is not an Elective

George Stephanis
Team Lead @ Automattic
@daljo628

AUTOMATTIC

VaultPress

vaultpress.com/tryit

What are electives?

First, what are Requirements?

Take all of these classes:

- 10400 Discrete Mathematical Structures
- 21100 Fundamentals of Computer Systems
- 22000 Algorithms
- 32200 Software Engineering
- 33200 Operating Systems
- 33500 Programming Language Paradigms
- 33600 Introduction to Database Systems

What are electives?

Choose **one** of **three** *Computer Systems* classes:

31800 Internet Programming

41200 Computer Networks

43500 Concurrency in Operating Systems

Choose **one** of **four** *Computational Techniques* classes:

44000 Computational Methods in Numerical Analysis

44200 Systems Simulation

47000 Image Processing

47900 Digital Libraries

What are electives?

Not essential to know.

What do we mean by “security”?

1. Making secure choices.
2. Writing secure code.
3. Selecting secure products (plugins, themes, libraries).
4. Properly responding to security vulnerabilities.

Making secure decisions

Making secure decisions

Avoid open WiFi.

Use a VPN.

HTTPS YOUR THINGS.

Don't reuse passwords.

High password entropy.

Use a password manager.

Two factor authentication.

Update all the things.

Security Questions.

Fingerprints !== passwords.

Don't trust your users. Ever!

Making secure decisions

Writing secure code

“Timing Attacks”

PHP wants to go fast.

Optimizations.

This can leak private data.

Writing secure code

```
Source String:

zDKIiW4J70OOgznG4oISc88uvIV73yJJIGvfthiVr3mZtHXfijoMcsM1VYDl

Test String:

DNos52bBEvkfJtBRRUujnLPtDfAttOXKRQhg419fNBP19SnAFckKGWXSpRpV
```

Writing secure code

```
Source String:

zDKIiW4J70OOgznG4oISc88uvIV73yJJIGvfthiVr3mZtHXfijoMcsM1VYDl

Test String:

zDKIiW4J70OOgznG4ol5c88uvIV73yJJIGvfthiVr3mZtHXfijoMcsMlVYDI
```

Writing secure code

`Source String:`

`zDKIiW4J70OOgznG4oISc88uvIV73yJJIGvfthiVr3mZtHXfijoMcsM1VYDl`

`Test String:`

`zDKIiW4J70OOgznG4oISc88uvIV73yJJIGvfthiVr3mZtHXfijoMcsM1VYDl`

Writing secure code

```
Source String:

zDKIiW4J70OOgznG4oISc88uvIV73yJJIGvfthiVr3mZtHXfijoMcsM1VYDl

Test Strings:

DNos52bBEvkfJtBRRUujnLPtDfAttOXKRQhg419fNBP19SnAFckKGWXSpRpV

zDKIiW4J70OOgznG4ol5c88uvIV73yJJIGvfthiVr3mZtHXfijoMcsMlVYDI

zDKIiW4J70OOgznG4oISc88uvIV73yJJIGvfthiVr3mZtHXfijoMcsM1VYDl
```

Writing secure code

Easy to correct! `hash_equals()`

```
if ( $var1 === $var2 ) {  }

if ( hash_equals( $var1 === $var2 ) ) {  }
```

Writing secure code

Sanitize your inputs, Escape your outputs.

Sanitizing is about CONTE**N**T.

Escaping is about CONTE**X**T

Writing secure code

Sanitizing is about **CONTENT** -- what it is meant to be.

If `$min` can only be an integer ≥ 1,

```
$min = 1;
if ( $_GET['min'] > 1 ) {
    $min = (int) $_GET['min'];
}
```

If `$orderby` can only be \`ASC\` or \`DESC\`,

```
$orderby = 'ASC';
IF ( 'DESC' === $_GET['orderby'] ) {
    $orderby = 'DESC';
}
```

Writing secure code

Escaping is about **CONTEXT** -- how it’s meant to be used.

The same string could be escaped many ways on a single page.

```
<h1 title="<?= esc_attr($t) ?>" onclick="alert( '<?= esc_js($t); ?>' )">
    <?= esc_html($t); ?>
</h1>
```

Also `esc_textarea, esc_url, esc_url_raw, esc_sql`

* `<?=` is a shortcut for `<?php echo` but be careful! Like PHP7, it isn't always available everywhere.

Writing secure code

Can your code tell the difference between

```
<form action="process.php">
    <input type="hidden" name="action" value="delete" />
    <input type="hidden" name="id" value="42" />
    <input type="submit" />
</form>
```

and

```
<img src="//target.io/admin/process.php?action=delete&id=42" />
```

Writing secure code

Nonce: Number used Once

```
<form action="process.php">
+ → <input type="hidden" name="_nonce" value="JD7ANaoKG" />
    <input type="hidden" name="action" value="delete" />
    <input type="hidden" name="id" value="42" />
    <input type="submit" />
</form>
```

The cryptographic nonce won't be known.

```
<img src="//target.io/admin/process.php?action=delete&id=42&_nonce=…………?" />
```

ALSO: **Nonces are for intent, not for authentication!** They are for validating that the user intended to do an action. *Always also check against user caps.*

Selecting secure products

Can be anything from plugins and themes to third-party PHP or JS libraries.

- Usage: Is it obscure or ubiquitous?
- Community: Is it abandoned, or does it have responsive maintainers?
- Author: Is the creator reputable? Have you used their products in the past?
- Code: Is anything obfuscated, or can you actually read it all?
- History: If they've had security concerns in the past, how did they address?
- Upgrades: Are new releases difficult or time-consuming to install?
- Disclosure: Do they have a way to report security issues privately?

Responding to a security vulnerability

1. Make it easy for folks to reach you. (HackerOne is a good start)
2. Communicate. Early and often.
3. Publicly credit individuals or companies that disclosed.
4. Be honest, even if it's honestly declining to answer a question.

Responding to a security vulnerability

Shipping a significant security release:

- Know what you're fixing, and only fix that.
- Don't include a Proof of Concept in your commit messages.
- Be somewhat vague about the exploit in your commit messages.
- Contact security@ and plugins@wordpress.org -- they'll appreciate it.
- security@ may help you roll out point releases to multiple versions.
- Be prepared to ship multiple point releases -- one per branch.

You’re not done. There will be a test.

Unfortunately, I’m not the one giving the test.

An email from someone, or a tweet about your software will be your test.

Be aware, and be prepared.

Questions?

@daljo628

gs@automattic.com

stephanis.info

Useful Resources:

phpsecurity.readthedocs.org owasp.org

vaultpress.com/tryit