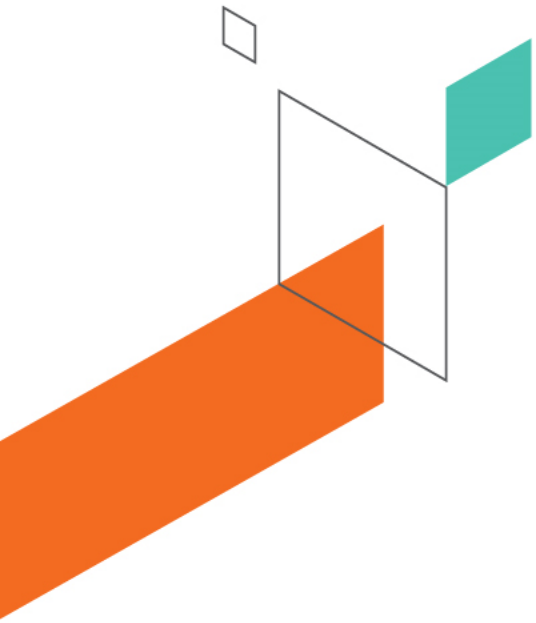


First Steps to Building Secure Magento Extensions

https://tale.sh/MLIN17

Talesh Seeparsan

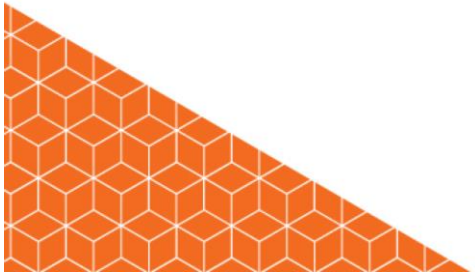
CTO
Bit79

There is no such thing as an unhackable site


You just need to be able to run faster than your friends

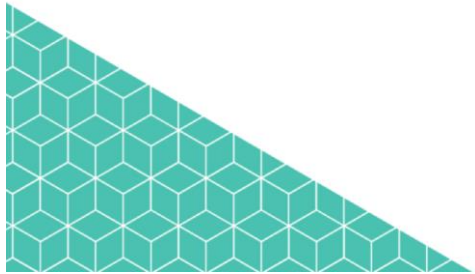
PART 1

- Lower level tools and strategies
- Useful for building a single extension
- Use during SDLC



PART 2

- Architecture level
- Useful for planning an entire site build
- Useful for securing live sites



Theme for our strategies:

- Don’t negatively affect team productivity.
- Let the computers do the work for us.

Part 1:
Securing at a code level

Tool #1 : Use PHPCS

- Scans our code and flags dangerous parts
- ECG Ruleset understands Magento 2
- Comes with built in security scans

```
$ composer require \
    magento-ecg/coding-standard
```

```
$ phpcs --config-set installed_paths /
  ./vendor/magento-ecg/coding-standard
```

```
$ phpcs --standard=EcgM2 /path/to/code
```

magento-ecg / coding-standard

Watch 44 | Unstar 216 | Fork 71

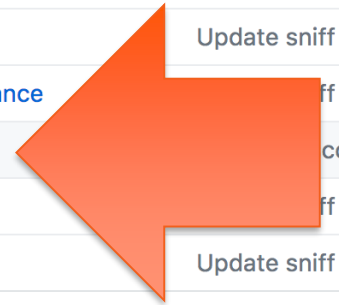
Code | Issues 8 | Pull requests 1 | Projects 0 | Wiki | Pulse | Graphs

Branch: master | coding-standard / Ecg / Sniffs / | Create new file | Upload files | Find file | History

zlik Update DiscouragedFunctions.php ... Latest commit a4b8a50 on Apr 26, 2016

..

Classes	Update sniff classes according to namespace usage	a year ago
PHP	Update sniff classes according to namespace usage	a year ago
Performance	ff classes according to namespace usage	a year ago
Security	couragedFunctions.php	a year ago
Sql	ff classes according to namespace usage	a year ago
Strings	Update sniff classes according to namespace usage	a year ago



magento-ecg / coding-standard

Watch 44 | Unstar 216 | Fork 71

Code | Issues 8 | Pull requests 1 | Projects 0 | Wiki | Pulse | Graphs

Branch: master — coding-standard / Ecg / Sniffs / Security / — Create new file | Upload files | Find file | History

zlik Update DiscouragedFunctions.php ... — Latest commit a4b8a50 on Apr 26, 2016

..

AclSniff.php	Update sniff classes according to namespace usage	a year ago
DiscouragedFunctions.php	Update DiscouragedFunctions.php	a year ago
ForbiddenFunctionSniff.php	Update ForbiddenFunctionSniff.php	a year ago
IncludeFileSniff.php	Update sniff classes according to namespace usage	a year ago
LanguageConstructSniff.php	Update sniff classes according to namespace usage	a year ago
SuperglobalSniff.php	updated superglobal sniff M1 and added to M2	a year ago

PHPCS Best Practices

- Run as a **`git/svn`** hook automatically

PHPCS Best Practices

- Fix any issues raised immediately

Tool #2 : OWASP ZAP

- Scans inputs instead of code
- Used by Magento HQ
- Industry standard

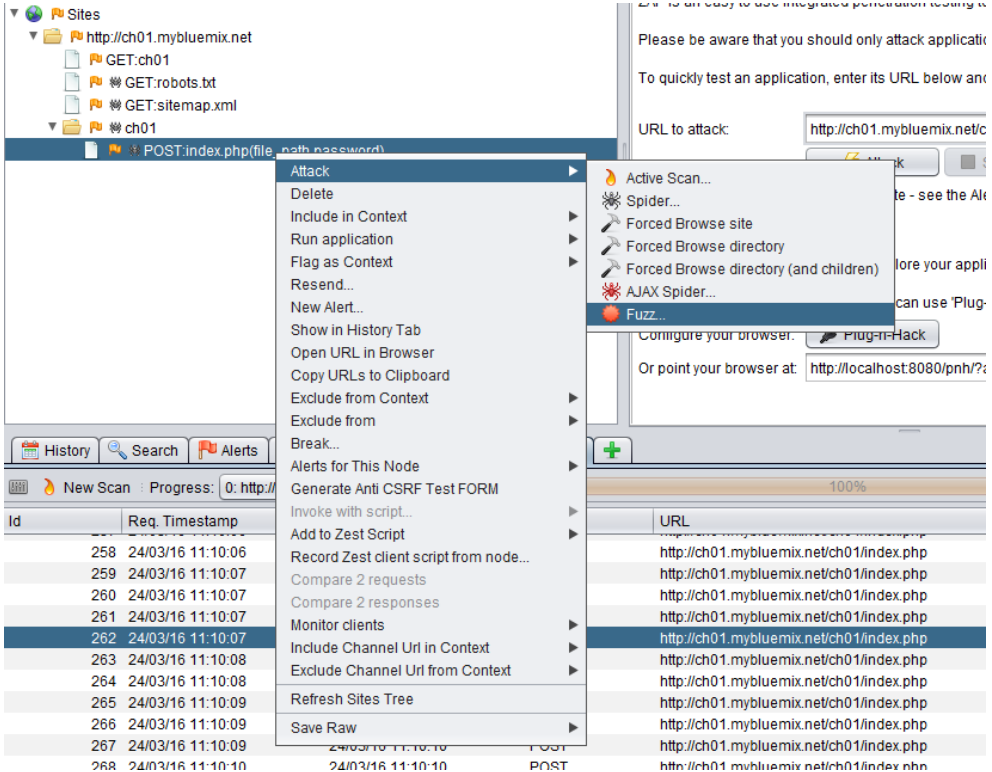
XSS attack string

```
<script>alert(document.cookie);</script>
```

XSS attack string

```
&lt;IMG
SRC=&#0000106&#0000097&#0000118&#0000097&#0000115&#0000099&#0000114&#0000105&#0000112&#0000116&#0000058&#0000097&#0000108&#0000101&#0000114&#0000116&#0000040&#0000039&#0000088&#0000083&#0000083&#0000039&#0000041&gt;
```

OWASP GUI



OWASP ZAP Demo

- http://tale.sh/owasp-zap-demo

OWASP ZAP Best Practices

- Let it run overnight/over the weekend, working while you sleep.

OWASP ZAP Best Practices

- Create tickets in Asana/Jira for each problem it finds.

Builtin Magento 2 security features

- And how/when to use them

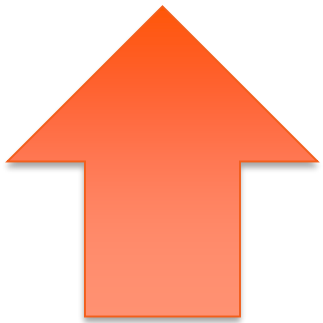
Use the Magento 2 ORM

- ~~Handcoded SQL queries~~
- Robust framework that facilitates Server side input validation

Defends against Injection and XSS

The Magento 2 Escaper

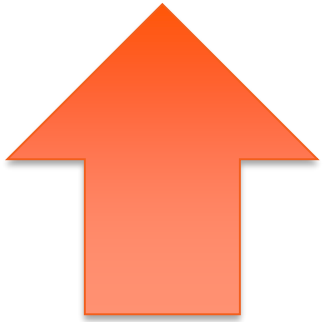
- Implementation: /lib/internal/Magento/Framework/Escaper.php
  - Usage: `<?php echo $this->escapeHtml(__($this->variable); ?>`



Defends against XSS

CSRF Defense : Anti Forgery Tokens

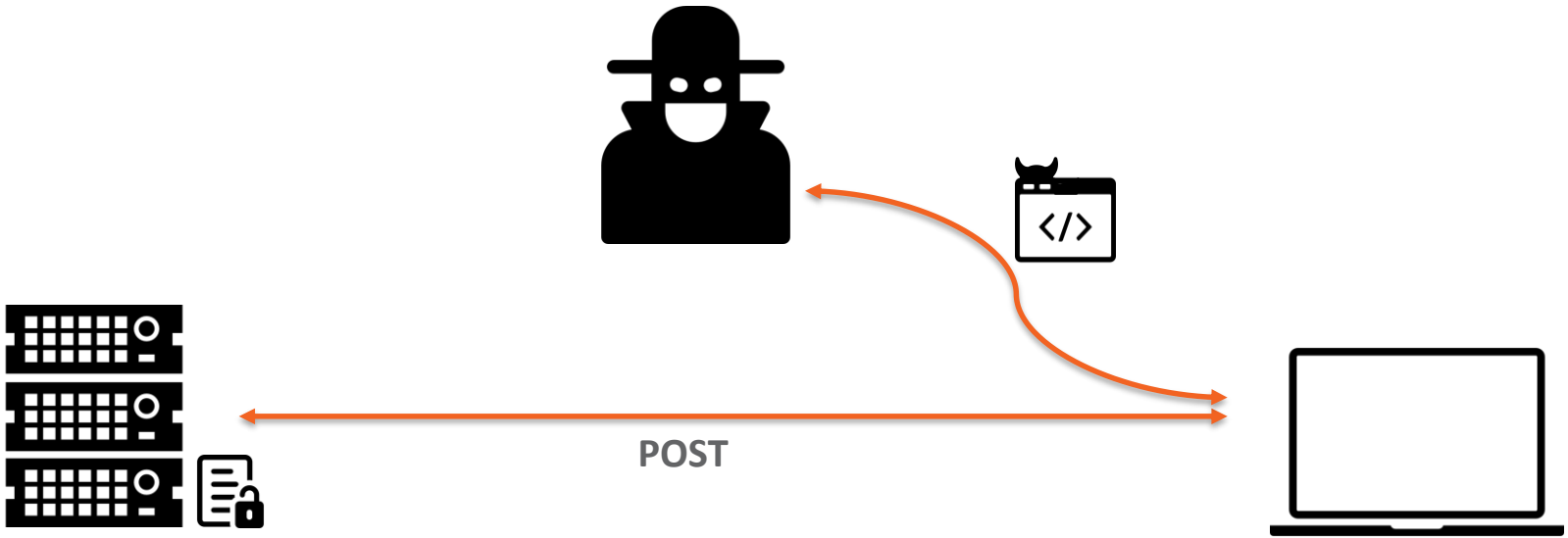
```
<?php echo $this->getBlockHtml('formkey')?>
```



Defends against CSRF

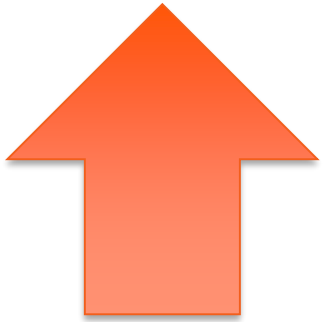
CSRF explanation

- Trick an authenticated user to POST information on your site



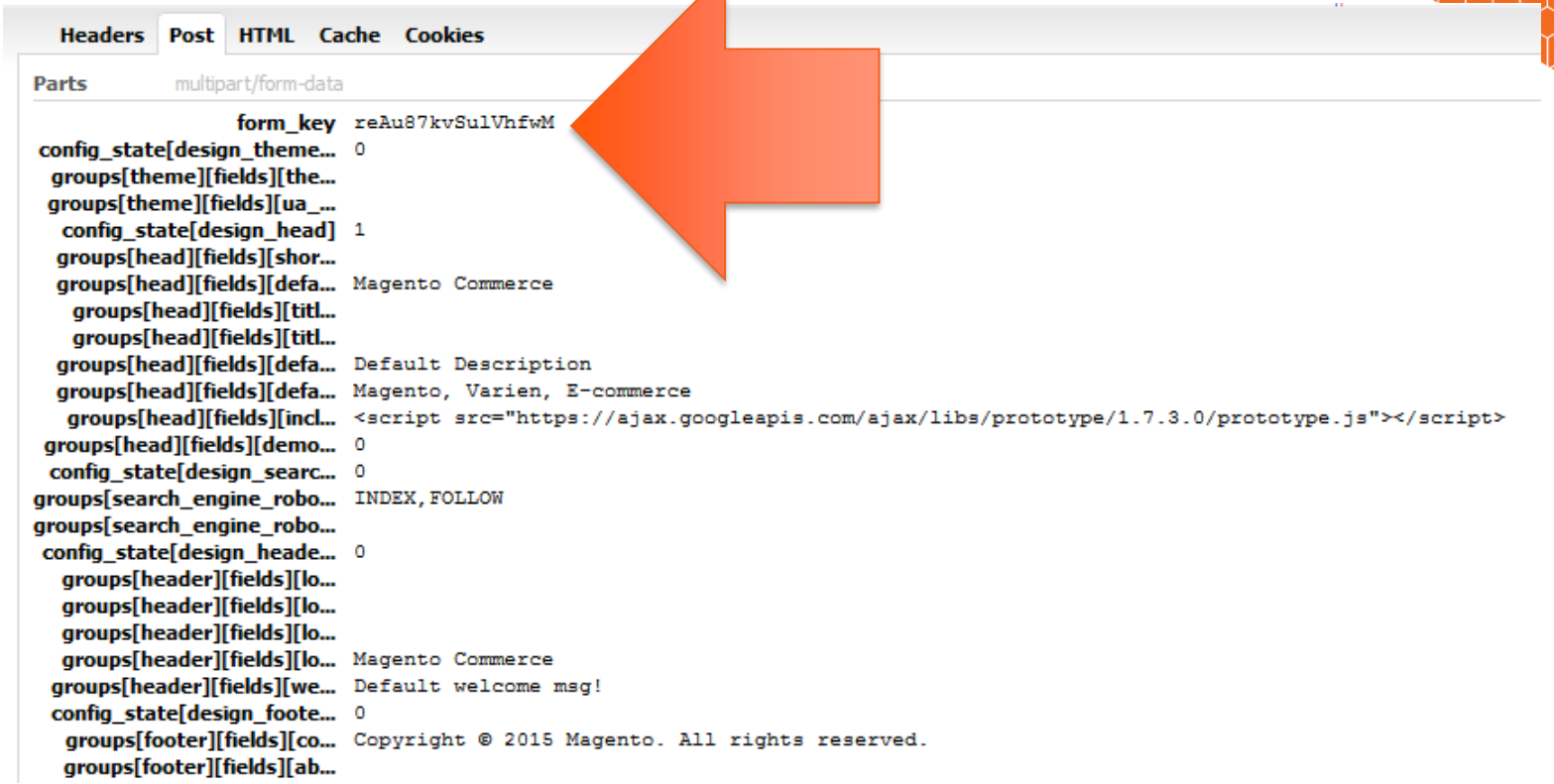
CSRF Defense : Anti Forgery Tokens

```
<?php echo $this->getBlockHtml('formkey')?>
```



Defends against CSRF

CSRF Defense : Anti Forgery Tokens



Defends against CSRF

Pay attention to cookie permissions

- "**HttpOnly**" flag is set on some important cookies eg:
  - "admin" cookie
  - "PHPSESSID" cookie
  - "X-Magento-Vary" cookie

- "**Secure**" flag is set on some important cookies eg:
  - "admin" cookie
  - "X-Magento-Vary" cookie

Defends against Broken Authentication

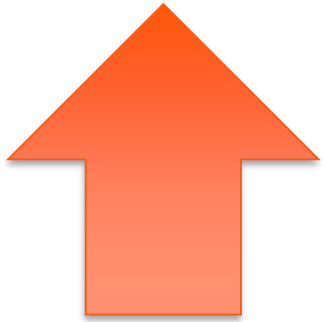
Rely on the CustomerSession Object

```
public function __construct(
        Context $context,
        CustomerSession $customerSession
    ) {
      parent::__construct($context, $customerSession);
    }
```

Defends against risks: **Insecure Direct object references**
Missing function Access control

Don’t roll your own Crypto!

```
<field id="password" translate="label" type="obscure" showInStore="0">
     <label>Password</label>
     <backend_model>Magento\Config\Model\Config\Backend\Encrypted</backend_model>
</field>
```



Defends against risks: Security Misconfigurations
Sensitive data exposure
Missing function level access control

Part 2:
Securing at an architecture level

Patches

- Subscribe to https://magento.com/security
- Patch quickly, plan your time for patches
- Easiest way to get hacked

Production is sacrosanct

- No unecessary files there
- No DB backups
- No git/svn data
- No test files
- No file backups
- File permissions must be impeccable
- No unnecessary tools like Magmi

Magento Malware scanner

- https://github.com/gwillem/magento-malware-scanner

- `wget git.io/mwscan.txt`
- `grep -Erlf mwscan.txt /path/to/magento`

Magento Security Council

Promotes & facilitates secure Magento stores globally.

https://magesec.org

External Site scanners

- https://www.magereport.com/
- https://magescan.com/
- Magento Security Scan from Magento Inc. (currently in Beta) securityinfo@magento.com (https://tale.sh/mss-beta)

Keep your Admin URL random

- Use the randomly generated one in Magento 2
- Generate your own in Magento 1
- Don’t use /admin /console /backoffice or anything similar
- Consider limiting access via IP Whitelist or even VPN

2FA for your admin URL

- https://github.com/magento-hackathon/Magento-Two-factor-Authentication
- https://github.com/nexcess/magento-sentry-two-factor-authentication

Check your composer for known vulnerabilities

- Upload your `composer.lock` file on https://security.sensiolabs.org/

- `php checker security:check /path/to/composer.lock`

Stronger password hashing

- `https://bitbucket.org/creaminternet/module-securepasswords`

PROCESS > TOOLS

PEOPLE > PROCESS > TOOLS

http://github.com/talesh/response

धन्यवाद
Thank you

https://tale.sh/MLIN17

@_Talesh

Magento Live

India | 2017