

OWASP Top 10

im Kontext von Magento

Ich

- ✦ Fabian Blechschmidt
(@Fabian_ikono)
- ✦ blechschmidt@fabian-blechschmidt.de
- ✦ PHP seit 2004
- ✦ Freelancer seit 2008
- ✦ Magento seit 2011
- ✦ Certified Magento Developer
- ✦ spielt gerne, aktuell mit
 - ✦ OWASP TOP10



Überblick

- Injection
- Cross-Site-Scripting
- Session-Management
- Unsichere direkte Objektreferenzen
- Cross-Site-Request-Forgery (CSRF)
- Kryptografisch unsichere Speicherung
- Unzureichende Absicherung der Transportschicht

Überblick

- Injection
- Cross-Site-Scripting
- Session-Management
- Unsichere direkte Objektreferenzen
- Cross-Site-Request-Forgery (CSRF)
- Kryptografisch unsichere Speicherung
- Unzureichende Absicherung der Transportschicht

Injection

Injection

- Ungeprüfte Parameter in sicheren Kontext einfügen

Injection

- Ungeprüfte Parameter in sicheren Kontext einfügen
- Verschiedene Formen

Injection

- Ungeprüfte Parameter in sicheren Kontext einfügen
- Verschiedene Formen
 - **SQL-Injection**

Injection

- Ungeprüfte Parameter in sicheren Kontext einfügen
- Verschiedene Formen
 - **SQL-Injection**
 - LDAP

Injection

- Ungeprüfte Parameter in sicheren Kontext einfügen
- Verschiedene Formen
 - **SQL-Injection**
 - LDAP
 - **mail()**

Injection

- Ungeprüfte Parameter in sicheren Kontext einfügen
- Verschiedene Formen
 - **SQL-Injection**
 - LDAP
 - **mail()**
 - Systemaufrufe (exec(), `rm -rf /`, passthru(), system())

SQL-Injection

- Magento kümmert sich um alles
 - ...wenn ihr Magento lasst!

SQL-Injection: Collections & Models

SQL-Injection: Collections & Models

- \$collection

SQL-Injection: Collections & Models

- \$collection
 - ->addFieldToFilter()

SQL-Injection: Collections & Models

- \$collection
 - ->addFieldToFilter()
 - ->addAttributeToSelect()

SQL-Injection: Collections & Models

- \$collection
 - ->addFieldToFilter()
 - ->addAttributeToSelect()
 - ->addAttributeToSort()

SQL-Injection: Collections & Models

- \$collection
 - ->addFieldToFilter()
 - ->addAttributeToSelect()
 - ->addAttributeToSort()
 - ->addAttributeToFilter()

SQL-Injection: Collections & Models

- \$collection
 - ->addFieldToFilter()
 - ->addAttributeToSelect()
 - ->addAttributeToSort()
 - ->addAttributeToFilter()

SQL-Injection: Collections & Models

- \$collection
 - ->addFieldToFilter()
 - ->addAttributeToSelect()
 - ->addAttributeToSort()
 - ->addAttributeToFilter()

SQL-Injection: Collections & Models

- \$collection
 - ->addFieldToFilter()
 - ->addAttributeToSelect()
 - ->addAttributeToSort()
 - ->addAttributeToFilter()
- \$model

SQL-Injection: Collections & Models

- `$collection`
 - `->addFieldToFilter()`
 - `->addAttributeToSelect()`
 - `->addAttributeToSort()`
 - `->addAttributeToFilter()`
- `$model`
 - `->load($id, $field=null)`

SQL-Injection: Collections & Models

- `$collection`
 - `->addFieldToFilter()`
 - `->addAttributeToSelect()`
 - `->addAttributeToSort()`
 - `->addAttributeToFilter()`
- `$model`
 - `->load($id, $field=null)`
 - z.B. `->load(„123“, „sku“)`
(geht nicht überall!)

SQL-Injection: Collections & Models

- \$collection
 - ->addFieldToFilter()
 - ->addAttributeToSelect()
 - ->addAttributeToSort()
 - ->addAttributeToFilter()
- \$model
 - ->load(\$id, \$field=null)
 - z.B. ->load(„123“, „sku“)
 - (geht nicht überall!)
 - Mage_Catalog_Model_Product

SQL-Injection: Collections & Models

- `$collection`
 - `->addFieldToFilter()`
 - `->addAttributeToSelect()`
 - `->addAttributeToSort()`
 - `->addAttributeToFilter()`
- `$model`
 - `->load($id, $field=null)`
 - z.B. `->load(„123“, „sku“)`

(geht nicht überall!)
 - `Mage_Catalog_Model_Product`
 - `loadByAttribute($attr, $val, $additionalAttributes = '*')`

mail() Injection

- mail (\$to, \$subject, \$message, \$additional_headers, additional_parameters)

mail() Injection

- mail (\$to, \$subject, \$message,\$additional_headers, additional_parameters)
- SMTP Header
 - From: owner@shop.com
 - To: customer@shop.com \n
 - Subject: Testmail
 - Date: Thu, 26 Oct 2006 13:10:50 +0200

mail() Injection

- mail (\$to, \$subject, \$message,\$additional_headers, additional_parameters)
- SMTP Header
 - From: owner@shop.com
 - To: customer@shop.com \n
 - BCC: mallory@hacker.net
 - Subject: Testmail
 - Date: Thu, 26 Oct 2006 13:10:50 +0200

mail() Injection über Kundenregistrierung

mail() Injection über Kundenregistrierung

Email Address *

test@fbtest.de
BCC: hacker@fbtest.de

mail() Injection über Kundenregistrierung

Email Address *

test@fbtest.de
BCC: hacker@fbtest.de

! "Email" is not a valid email address.

Personal Information

First Name *

Fabian

Email Address *

test@fbtest.deBCC: hacker@fbtest.de

mail() Injection über Kundenregistrierung

Email Address *

test@fbtest.de
BCC: hacker@fbtest.de

- geht nicht.

! "Email" is not a valid email address.

Personal Information

First Name *

Fabian

Email Address *

test@fbtest.deBCC: hacker@fbtest.de

Mails versenden mit Magento

- `Mage::getModel('core / email_template')`
- `->setTemplateSubject($mailSubject)`
- `->sendTransactional($templateId, $sender, $email, $name, $vars, $storeId);`
- auf Basis von `Zend_Mail` → kümmert sich um alle Probleme!

Überblick

- Injection
- Cross-Site-Scripting
- Session-Management
- Unsichere direkte Objektreferenzen
- Cross-Site-Request-Forgery (CSRF)
- Kryptografisch unsichere Speicherung
- Unzureichende Absicherung der Transportschicht

Cross-Site-Scripting

- Braucht ihr das wirklich?
- Wichtig:
 - ALLES muss extra escaped werden!
 - HTML-Kontext (`$this->escapeHtml()`)
 - JavaScript-Kontext

ALLES

- `$_POST`
- `$_GET`
- `$_COOKIE`
- `$_SERVER['HTTP_HOST']`
- `$_SERVER['HTTP_USER_AGENT']`
- `$this->getRequest()->getParam(*)`;

filter_var() & json_encode()

- FILTER_SANITIZE_EMAIL
- FILTER_SANITIZE_SPECIAL_CHARS
- FILTER_SANITIZE_FULL_SPECIAL_CHARS
- json_encode()

Überblick

- Injection
- Cross-Site-Scripting
- Session-Management
- Unsichere direkte Objektreferenzen
- Cross-Site-Request-Forgery (CSRF)
- Kryptografisch unsichere Speicherung
- Unzureichende Absicherung der Transportschicht

Session Management

- Session Hijacking
- Session Fixation Angriff

Session Hijacking

- Benutzer hat authentifizierte Session
 - Hacker bekommt Session-ID
- Braucht anderen Fehler
 - üblicherweise XSS

Magento gegen Session Hijacking

- Validate REMOTE_ADDR
- Validate HTTP_VIA
- Validate HTTP_X_FORWARDED_FOR
- Validate HTTP_USER_AGENT

Session Hijacking

- einzige gute Lösung: Code bugfrei halten

Session Fixation Angriff

- Hacker schiebt Benutzer Session-ID unter
- Benutzer authentifiziert sich mit dieser

Session-ID über URL unterschrieben

- z.B. <http://shop.nivea.de/?SID=12345>

Name	frontend
Value	12345
Host	shop.nivea.de
Path	/
Expires	At end of session
Secure	No
HttpOnly	No

Magento gegen Session-Fixation

- nach erfolgreichem Login
 - → neue Session-ID generieren

Ikonoshirt_SecureSession

- wenn Session in URL / POST
 - → Logout (→neue Session-ID)
 - ACHTUNG: SID zum StoreView wechsel geht nicht mehr!

Überblick

- Injection
- Cross-Site-Scripting
- Session-Management
- Unsichere direkte Objektreferenzen
- Cross-Site-Request-Forgery (CSRF)
- Kryptografisch unsichere Speicherung
- Unzureichende Absicherung der Transportschicht

Unsichere direkte Objektreferenzen

- Benutzer: Alice macht Bestellung
- Benutzer: Mallory ruft diese auf:
- http://mage1-7-0-2.dev/sales/order/view/order_id/1/

Beispiel, geht nicht!

Unsichere direkte Objektreferenzen

- Typisches Problem:
 - in der Collection wird gefiltert
 - beim **direkten Aufruf** wird nicht mehr geprüft

Überblick

- Injection
- Cross-Site-Scripting
- Session-Management
- Unsichere direkte Objektreferenzen
- Cross-Site-Request-Forgery (CSRF)
- Kryptografisch unsichere Speicherung
- Unzureichende Absicherung der Transportschicht

Cross-Site-Request-Forgery (CSRF)

- Aufruf von URL erzeugt Aktion
 - ggf. ohne das der User das will
- Beispiel:
- ``

CSRF Beispiel

- schlimmere Sache:
 - <http://mage1-7-0-2.dev/customer/remove/?id=123>
- Maßnahmen:
 - HTTP Method prüfen
 - Token mitgeben (wie im Magento Backend)

Überblick

- Injection
- Cross-Site-Scripting
- Session-Management
- Unsichere direkte Objektreferenzen
- Cross-Site-Request-Forgery (CSRF)
- Kryptografisch unsichere Speicherung
- Unzureichende Absicherung der Transportschicht

Kryptografisch unsichere Speicherung

Kryptografisch unsichere Speicherung

- Wie speichert Magento Passwörter?

Kryptografisch unsichere Speicherung

- Wie speichert Magento Passwörter?
- MD5, zwei Zeichen Salt.

Kryptografisch unsichere Speicherung

- Wie speichert Magento Passwörter?
- MD5, zwei Zeichen Salt.
- Zeit zum knacken pro Passwort?

Kryptografisch unsichere Speicherung

- Wie speichert Magento Passwörter?
 - MD5, zwei Zeichen Salt.
- Zeit zum knacken pro Passwort?
 - Keine Ahnung, aber
1,3 Milliarden+ Versuche / Sekunde möglich

Wie Passwörter speichern?

- key derivation function, z.B.
 - PBKDF2-HMAC-SHA256, $c = 86000$
 - bcrypt, $\text{cost} = 11$
 - scrypt, $N = 2^{14}, r = 8, p = 1$
- langer Salt, z.B.
 - `md5(username)`
 - `md5(time())`

Ikonoshirt_PBKDF2

- Observer auf erfolgreichem Login
 - wenn Passwort md5-Hash, ersetzen durch PBKDF2
- Admin, API und Customer-Login

Überblick

- Injection
- Cross-Site-Scripting
- Session-Management
- Unsichere direkte Objektreferenzen
- Cross-Site-Request-Forgery (CSRF)
- Kryptografisch unsichere Speicherung
- Unzureichende Absicherung der Transportschicht

Unzureichende Absicherung der Transportschicht

Unzureichende Absicherung der Transportschicht

- SSL!

Unzureichende Absicherung der Transportschicht

- SSL!
- ÜBERALL!

Unzureichende Absicherung der Transportschicht

- SSL!
- ÜBERALL!
 - weil die Session-ID schützenswert ist

Secure Sockets Layer

- ordentliches Zertifikat für richtige Domain
- Next Step: alle Cookies secure-flag: 1
 - damit sie nur via HTTPS übertragen werden

Ikonoshirt_ StrictTransportSecurity

- Implementiert HSTS
 - (HTTP Secure Transport Security)
- Kommunikation nur via SSL, kein http!
- Kein HTTPS -> keine Verbindung



[Learn more about the Magento Hackathons](#) and [register](#) to be part of a Magento Hackathon.

[Register for the next Hackathon »](#)

Berlin (11. - 13.1.2013)

Price: 120 €

The Magento Hackathon will take place in a nice youth hostel in the south of Berlin, Germany's capital city. An international meeting for new and nousey, as well as for experienced and committed Magento developers.

[View details »](#)

Links

- <https://github.com/ikonoshirt/secureSession>
- <https://github.com/ikonoshirt/pbkdf2>
- <https://github.com/ikonoshirt/StrictTransportSecurity>