Essential PHP Security

By Arne Blankerts

Tutorial Day, ZendCon 2009

thePHP.cc

Who we are

Premium PHP Consulting & Training. Worldwide.

thePHP.CC
the php consulting company

Sebastian Bergmann

Arne Blankerts

Stefan Priebsch

Security in PHP Projects

thePHP.cc

The myths, the facts and the solutions

Types of security?

- Transport layer
- Infrastructure
  - Inter server communication
- Data warehouse
- Interface design
- User level
- Application level

Transport layer

- Encryption
  - TLS / SSL
  - SSH
  - Custom
- Public vs Tunnel
- Reachability
  - Open?
  - Restricted?
    - By IP-Range?
    - Login?

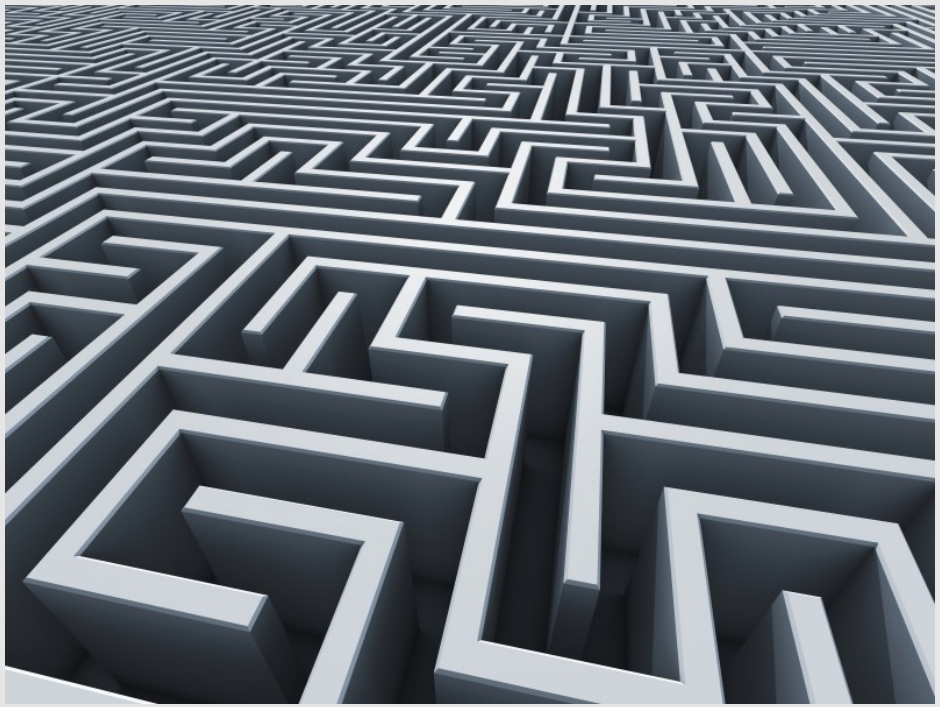
Infrastructure

- Single Server vs Multihomed
- Trust in ISP
  - Network
  - Architecture
  - Routing
- Needed services
  - Webserver
  - Database
  - 3rd party services

Data warehouse

- Connectivy
- Reliabilty
  - Storage concept
    - Encryption
- Stability
  - Can it handle peaks?
- Architecture

User interface



- Clean interface
  - readability
  - Navigation
- Error handling
- Error messages
- Follow HIGs
  - Dialog button order
  - Icon language
- I18n / i10n

User level

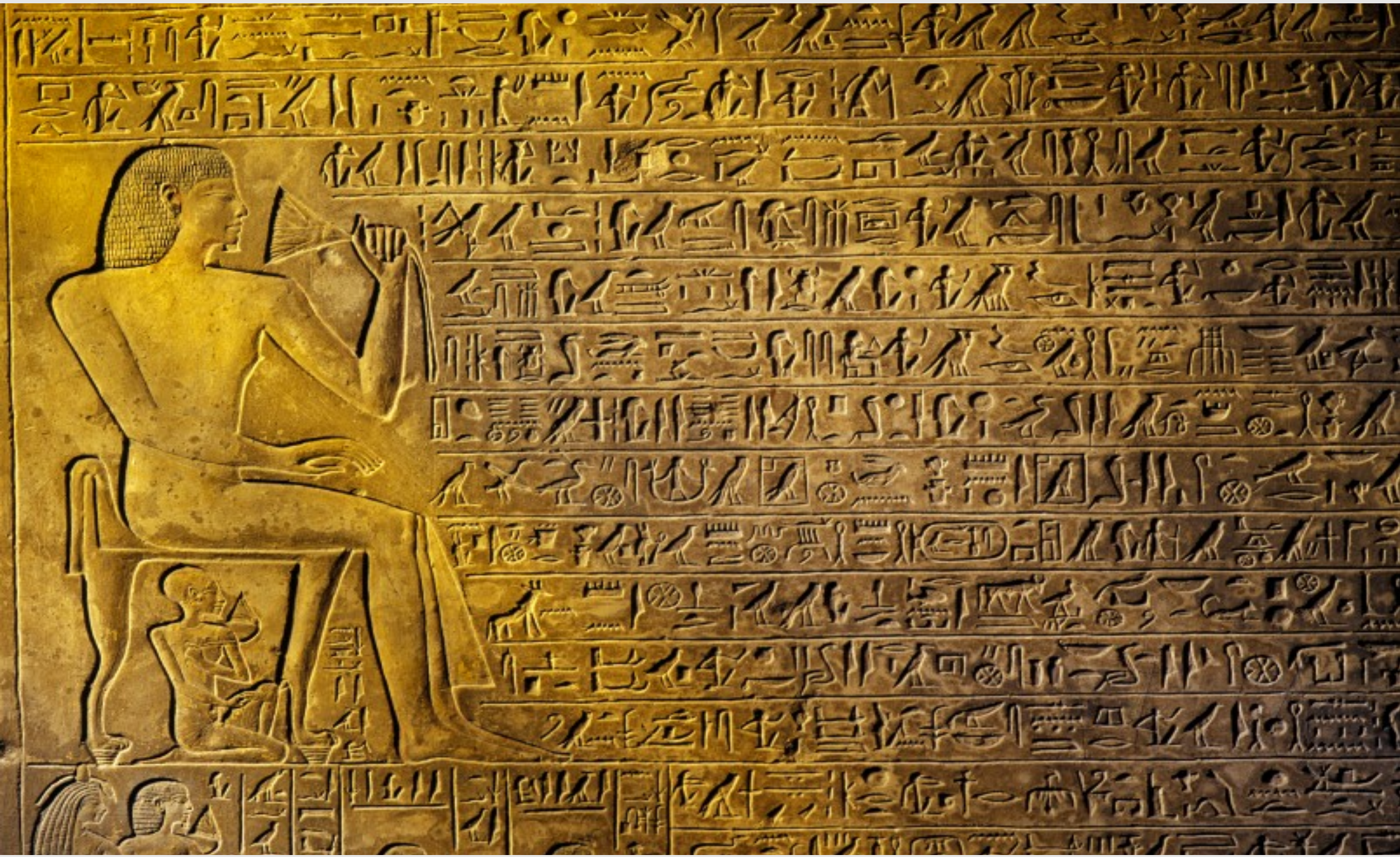
- Trustabilty
- Responsibilty
- Useability

- **Sense of security**

Application level

- Combines all
- The software stack
  - Client application
    - Browser
    - Custom App
  - Website
    - PHP
    - HTML
    - JavaScript
  - Databases
  - APIs

Security language



Security language

- Exploit
  - Software that makes use of a security problem
- Root exploit
  - An exploit that after being executed gains administrative privileges to the attacker
- Remote exploit
  - An exploit not requiring a local login or an active user account

Security language

- Keylogger
  - Software or Hardware saving every keystroke made for later (abusive) replay
    - Especially easy with wireless keyboards
      - Eavesdropping even from a distance
- Root kit
  - A Software that virtually takes over the computer hiding its existence and activity from administrators

Security language

- Injection
  - Sending malicious input to a processing software or device to exploit a vulnerability
- SQL Injection
  - SQL based injection with the intention to modify data, overwrite logins or otherswise manipulate the database

Security language

- Virus
  - Software distributing itself over a network or transportable media devices containing malicious code and runs in the background
- Viral
  - Originally a marketing form
    - Builds upon the fact people may distribute stuff they consider interesting
  - Usually contains “hidden” malicious code

Security language

- Zombie or bot
  - A computer under remote control without the original owner knowing
- Bot net
  - A big group of zombie computers usually controlled via IRC or other chat networks

Security language

- Social engineering
  - Abuse of known personal information
    - Or trying to get more information
  - Also the “abuse” of typical human behavior
    - General curiosity
    - Fears and hopes
    - Other emotions
- Phishing
  - Linked to social engineering
  - Usually redirects to faked but identical looking clones of a site

Security language

- DoS
  - Abbreviation for “Deny of Service”
  - Some claim, it's also an operating system ;)
  - An attack, trying to limit the availability of a service
    - By exploiting a crash problem
    - By abusing bandwidth

Security language

- Distributed DoS
  - A DoS using many computers at once
    - Usually performed by use of a bot network

Security language

- XSS - Cross site scripting
  - Exploit injecting HTML or scriptcode
- CSRF / XSRF - Cross site request forgery
  - Exploit requesting unauthorized operations

Questions so far?

thePHP.cc

Attack vectors

Attack vectors

- Mass mails (spam)
- Social engineering
- Script based attacks
  - Exploits against user software stack
    - Browsers
    - Plugins
  - Exploits against backend code
    - PHP
    - Webserver
    - Database

Cross site scripting

thePHP.cc

Cross site scripting

- Abbreviated to XSS
- Modifies a website with injected code
  - Not always easy to spot for an end user
  - Sometimes invisible, code only
- Allows stealing of user data
  - Passwords
  - Credit card data
  - Bank details
  - ...

Cross site scripting

- Allows for manipulation of content
  - Fake news possible
  - Self referring (chained xss) to raise “trust”

Cross site scripting

- Different types of XSS
  - Level 0: DOM
    - Temporarily, only in the browser based on JS
    - Requires a crafted link including the XSS code
  - Level 1: Non persistent
    - Temporarily, also requires crafted link
    - In the browser based on generated server output
  - Level 2: Persistent
    - Saved in the backend
    - Delivered every time on every pageload
    - No explicit user interaction needed

XSS in Action

thePHP.cc

How does it work?

- JavaScript security broken
  - Same-Domain-Policy not violated
    - Injected code runs as local
- Lousy checking of input
  - Or wrong methods to filter
- Missing escaping of output
- Wrong assumptions on user intentions

How to avoid

- Filter input
  - Check types
  - Check formats
  - Check bounds
- Escape output
  - Keep scope in mind
    - Database vs. HTML vs. JavaScript
- **Do not repair user input**

Sessions

thePHP.cc

Sessions

- HTTP is a stateless protocol
  - Every request is independent and standalone
- Sessions allow for logical links
  - Session ID usually a generated string
  - Fairly unique
  - Preferably stored in a cookie
- Server side management

Sessions

- Problems:
  - Session ID is stored remotely
    - Untrustworthy source
    - May be outdated
    - User might have changed
  - PHP does not really validate an ID
    - May contain malicious chars
    - May not point to an actual session

Session fixation

- Attack via crafted session id
  - May be provided by a link (?PHPSESSID=xx)
  - May use an XSS to set a cookie
  - May use plugins like flash
    - Breaks JavaScript security
- Takeover of Session by attacker
  - Privilege escalation

Session fixation

- Validate a session id manually
  - Do not use id of non existing session
- Always change session id after
  - User login
  - Permission change
  - Restart of session
- Only uses cookies
  - Disable session id via URL
  - Use http only cookies
- Disable trans_sid

Cross site request forgery

thePHP.cc

CSRF

- Also known as “session riding”
- Abuse of active sessions on 3rd sites
  - Embeded by iframe or image
  - Using javascript or xss
- Hugely underestimated problem
  - Typical targets are banks or bidding platforms
  - Checks hardly ever implemented

CSRF – Counter measures

- Do not use $_REQUEST
  - Deliberately use $_POST, $_GET or $_COOKIE
    - Try to avoid $_GET where possible to raise the bar
- Introduce single-use tokens
  - Add “anti-csrf” hidden field
  - Make sure tokens are not predictable
  - Do bail out on missmatch

CSRF – Wrong measures

- HTTP-Referrer checks
  - A Referrer can easily be faked
  - Is not a required HTTP header field
  - On XSS will automagically be correct
- Use $_POST only
  - While raising the bar it does not stop CSRF
- IP verification
  - Big ISPs may have proxy farms
    - Requests may come from various valid IPs

Click Jacking

- Wordplay
  - Combination of Click and Highjacking
- Uses CSS and IFRAME
- Not an exploit
- Protection:
  - NoScript Extension
  - JavaScript Iframe Test

Code injections

Code injections

- Remote include vulnerability
  - Possibly oldest PHP introduced problem
    - Partly “fixed” since PHP 5.2
      - disallowing remote includes by default
    - Sometimes reintroduced by use of eval()
      - Rumor has it, “eval” and “evil” are brothers for a reason
  - Maybe exploited by upload forms
    - Do a mime type check
    - Do a virus check
    - Do not use user input in filenames you plan using for include or require
  - Do not store uploads under document root

Attacks against databases

thePHP.cc

SQL Injection

- SQL Syntax in values used for query
  - Works in any type of sql statement
- Allows for information disclosure
- Allows unauthorized access
  - Privilege escalation
    - On database level
    - On application level
- Allows data manipulation

SQL Crash course

- `Select * from table where username='$user' and passwd='$passwd'`
- `Select * from table where ID=$id`
- `Select * from table oder by $field`

SQL Attacks

- String vs. Integer
- Escaping
  - Union select
- Comments
- Filter evasion
  - Char() / Code()
  - Hex()

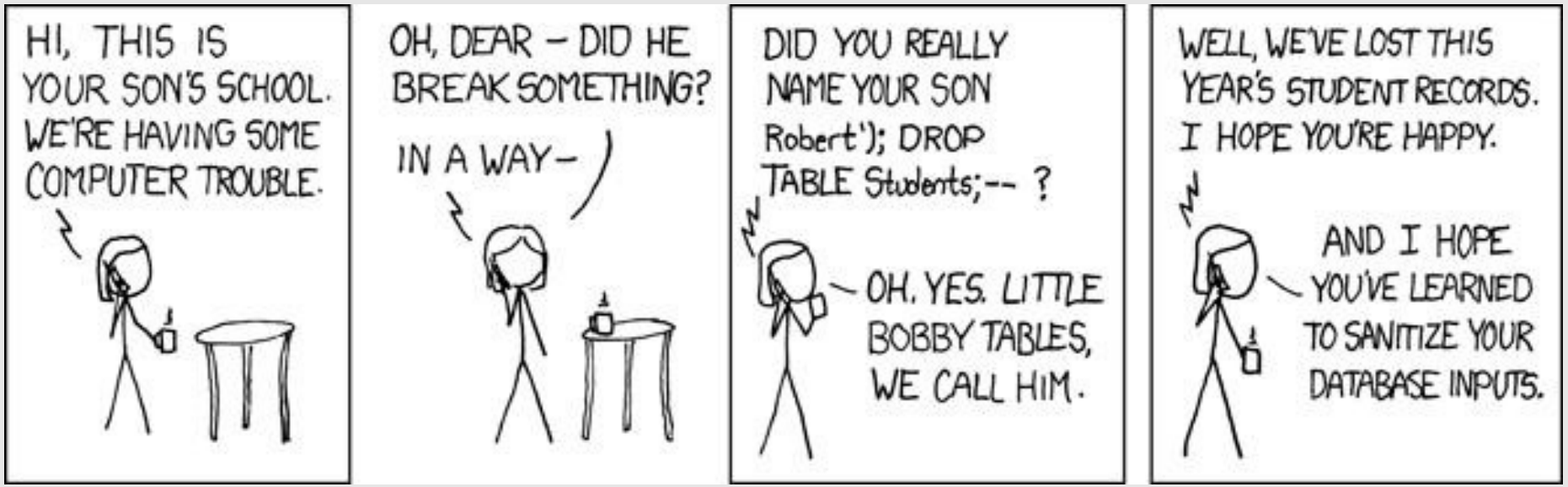
SQL Injection – a sample

- Set $user to one of the following
  - admin' --
  - admin' #
  - admin' /*
  - ' or 1=1 --
  - ' or 1=1 #
  - ' or 1=1 /*

SQL Injections


```
Select * from table where username='admin'
        --' and passwd='$passwd'
```

Data warehouse



http://xkcd.com/327/

SQL Injection – What to do?

- Use prepared Statements
  - Fixes the problem for values
  - Does not work for fieldnames
    - Filter fieldnames by whitelist
- Use escape functions
  - ext/mysql: mysql_real_escape_string()
  - ext/pdo: pdo::quote()
    - pdo + odbc: quote() not supported, does nothing
- Avoid using “select * from ...”
  - Performance gains as a side effect

Validate, validate, validate...

thePHP.cc

Validate!

- Almost all exploits work because of broken or missing validation code
  - Be politically incorrect: Every request is an attack until proven otherwise
  - The internet *is* evil
- Be paranoid
  - Just because you're paranoid it doesn't mean they're not after you
    - Scripts and bot networks scan for vulnerabilities 24 hours a day

Wanna play?

Congrats!

thePHP.cc

Thank you!

http://joind.in/talk/view/880

Please rate and comment!

Contact

- Slides will be on slideshare
  - http://slideshare.net/theseer
  - http://www.slideshare.net/group/thephpcc
- Contact options
  - Email: team@thePHP.cc / arne@thePHP.cc
- Follow us on twitter:
  - @arneblankerts
  - @thephpcc