Why Windows XP will be the DENIAL OF SERVICE Exploitation Tool of Choice for Internet Hackers Everywhere

by Steve Gibson, Gibson Research Corporation

Page last modified: Mar 15, 2005 at 12:05

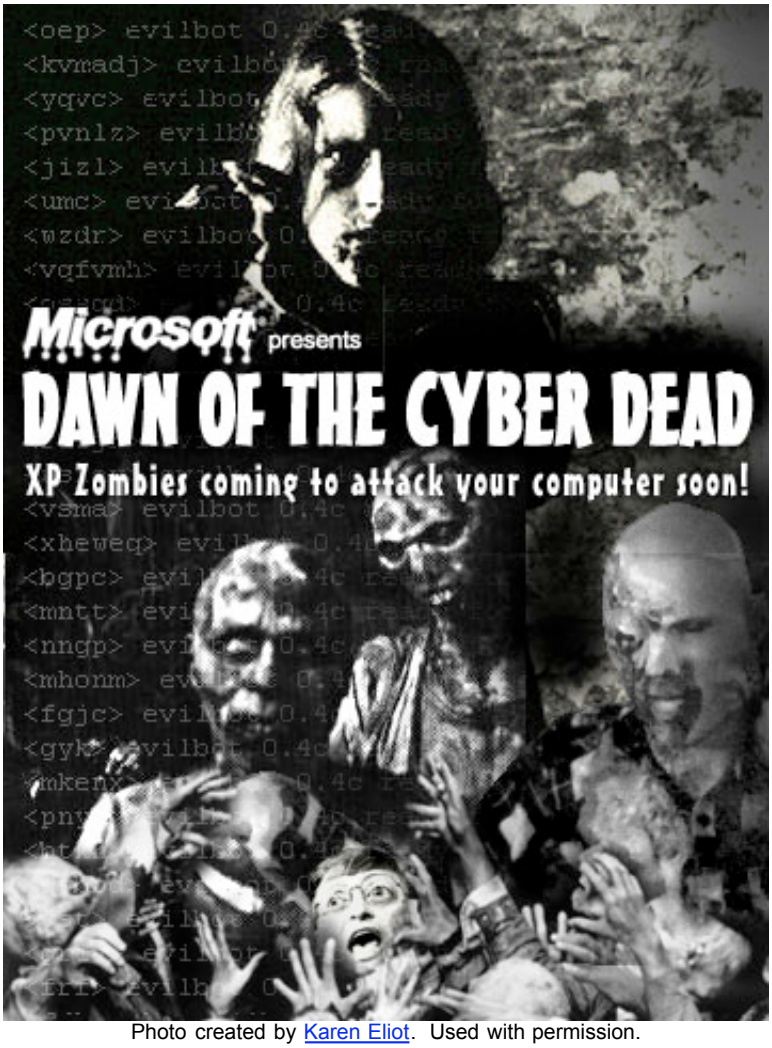


Photo created by Karen Eliot. Used with permission.

Page Updates & News:

- **06/15/2001 — Network Egress Filtering**
  I added a new section to examine and address the comments that "Network Egress Filtering" by ISP's is the "real solution" to the problem of Denial of Service

attacks.

- **06/20/2001 — GRC.COM Attack Log**
  On Wednesday, June 20th, 2001, we were attacked by 195 Windows 2000 servers running insecure versions of Microsoft's IIS web server. IIS was the apparent point of hacker entry into the system. We describe the attack and list all attacking IP addresses and machine names (where available).
- **06/28/2001 — My Meeting with Microsoft about WinXP**
  On Thursday, June 28th, 2001, I accepted Microsoft's invitation for an eight-way telephone conference with their top Windows XP executives and developers. I believe that they intended to show me why I was wrong about this whole full raw sockets issue. But instead they showed me that they really don't understand the fundamentals of security.
- **07/05/2001 — A Brief Summary of This Page's Arguments**
  This page can be rather daunting due to its length and the interlocking complexity of its various arguments. I have prepared this "brief summary page" for those who just want to understand the situation without all of the detailed background and supporting evidence.
- **07/19/2001 — Microsoft Laughs Off Windows XP Security**
  The Register's Thomas Greene interviewed Microsoft's Security Program Manager Scott Culp during the 2001 Blackhat and DEFCON conferences. According to The Register, Scott spent most of the time laughing about these issues. As you will see from my analysis, he also introduced a lot of spin.

Another LONG page . . .

I know that this is another of my looooooong pages. I worry that it won't be nearly as fascinating as my account of Wicked and the DDoS attacks. However, this is a complex and important issue that can not be quickly summarized.

If you are someone who eats dessert first, I urge you to at least read the story of "'Junior' and his XP Gang". (click the link) then I hope you will come back up here and start from the beginning.

"Hacker" vs "Cracker"

When I want to describe the actions of a "malicious hacker" the term "cracker" falls far short of the mark for me. It just doesn't seem very malicious. So I have stayed with the term "malicious hacker" because it is precisely descriptive.

I know that the majority of people who proudly call themselves "hackers" honor the same ethical principles I do **. . .** so NO disrespect is EVER meant.

> *"Steve,*
>
> *I've reviewed your note with the Windows network development architects, as well as our Corporate IT Security and Security Response groups. Your instincts are correct --*

they thoroughly understood the nature of the issue ***. . ."***

— A Microsoft Executive

With all due respect to Microsoft, I believe that either the right people within the organization are not yet fully aware of this issue, or that they have not really "thoroughly understood the nature of the issue."

Microsoft has a lot of really smart people — from Bill Gates and Steve Ballmer right on down the line. But they are human, and they sometimes make human mistakes. Sometimes it's worse than that, and as a company they're stubborn in the face of some really bad decisions. Like script-enabling their eMail clients so the virus du jour, like Melissa, can impersonate the user and happily eMail itself across the Internet to everyone in our address books.

What a DUMB thing.

My concern today is that we have another **SERIOUSLY DUMB IDEA** in the works from Microsoft in Windows XP.

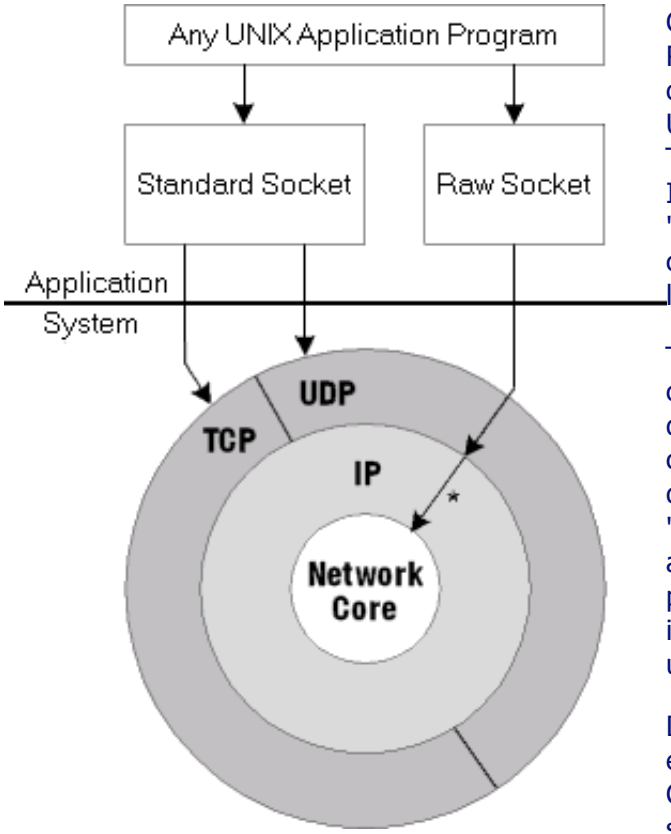
I regret my silence when scripting was being added to eMail. It was the dumbest thing I had ever seen, but I didn't care since I use Eudora. So I didn't work to make the world take notice. Now eMail viruses are born daily to travel the Internet at light speed. And it could have — should have — been prevented.

From a recent SANS Security article (See the Security Editor's Note)

> **13 & 14 June 2001— Malicious E-Mail Could Cause Problems for Japanese Wireless Internet Customers**
>
> A Japanese wireless phone carrier has warned subscribers of its I-Mode wireless Internet service that malicious e-mail messages could cause their phones to dial an emergency number, make lots of calls, or freeze the phone screen. The company advises its customers not to open e-mail from unknown sources and offers suggestions for thwarting the potential problems.
>
> The ComputerWorld Story The CNET Story
>
> [Editor's Note: Script-enabled mail and web clients are a disaster, and apparently G3 cell phone manufacturers have fallen into the same trap as the designers of MS Outlook and the people who invented Javascript for web browsers. At least with Netscape the user can disable Java and Javascript for web and mail. One suspects that G3 (third generation) cell phone users will not be so lucky.]

This time, with the disaster of Windows XP support for "RAW SOCKETS" looming, **there is still time to get Microsoft to yank it out.** But as the correspondence below demonstrates, I have not yet managed to reach the right people or convince them that they must.

What are "Sockets"?
And why are some of them "Raw"?



Circa 1981 — The Computer Systems Research Group (CSRG), at the University of California at Berkeley, first mated the Unix operating system to the Internet. This was done by implementing the Internet protocols and creating a so-called "TCP/IP Stack" for Unix. This is shown as concentric regions in the diagram to the left.

To simplify the task of creating Internet-communicating applications, the CSRG designed a simplified abstraction of the complex underlying protocols. They dubbed this abstraction "Sockets" or "Berkeley Sockets". Under this system, application programs request easily-programmed Internet "sockets" and are insulated from the details of the underlying network protocols.

Data is exchanged across the Internet by either establishing a bi-directional "TCP Connection" between two machines, or by sending a uni-directional "UDP Datagram" message from one machine to another. Both of these data transferring operations employ **standard sockets.**

Smooth and orderly traffic flow across the Internet requires machines to inform each other of various non-data events such as closed ports, network congestion, unreachable IP addresses, etc. The ICMP (Internet Control Message Protocol) was created to fill this need.

The operating system's built-in TCP/IP stack automatically and transparently generates and receives most of these "Internet plumbing" ICMP messages on behalf of the machine. To facilitate the creation of Internet plumbing applications, such as "ping" and "traceroute", which also employ ICMP messages, the Berkeley designers allowed programmers to manually generate and receive their own ICMP, and other, message traffic. As shown in the diagram above, the Berkeley Sockets system provides this power through the use of a so-called "Raw Socket". A Raw Socket short-circuits the TCP/IP stack to open a "backdoor" directly into the underlying network data transport.

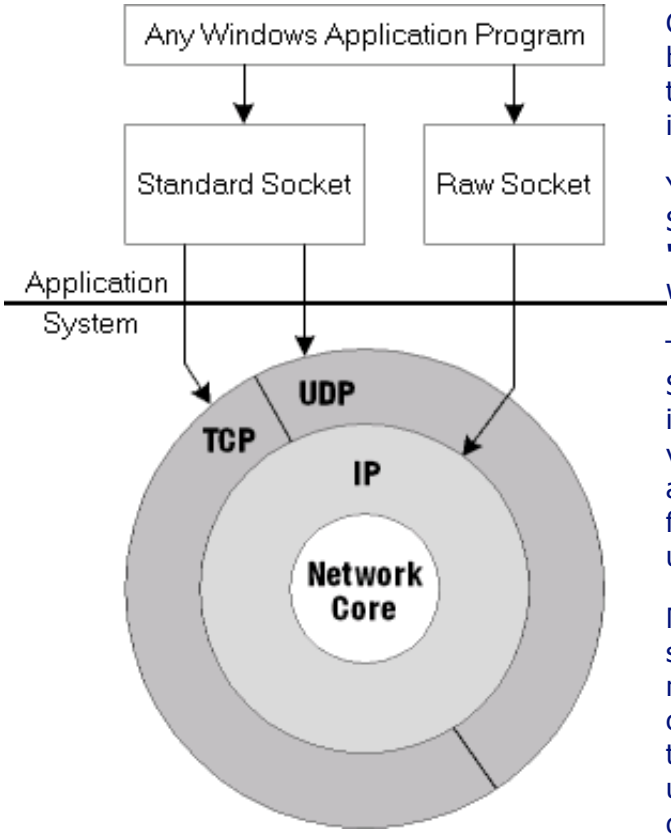
This provides full and direct "packet level" Internet access to any Unix sockets programmer.

Beyond their use for supporting simple "ping" and "traceroute" commands, the original Berkeley designers intended **Raw Sockets to be used for Internet protocol research purposes only.** Because they fully appreciated the inherent danger of abuse of Raw

Sockets, they deliberately denied Raw Socket access to any applications not running with maximum Unix "root" privileges. User-level applications were thus prevented from accessing and potentially abusing the Raw Sockets capability. (See asterisk '*****' in diagram above.)

Full Raw Sockets were created as a potent research tool. They were NEVER INTENDED to be shipped in a mass-market consumer operating system.

The Traditional (safe) Microsoft Stack



Compare the schematic diagram we've been looking at above, to Microsoft's traditional Windows Sockets (WinSock) implementation to the left.

You will notice that the Windows' Raw Socket's connection **does not "penetrate"** the encompassing IP wrapper layer.

This means that while Windows' Raw Sockets can be readily used for their intended and safe purpose of generating valid ICMP ping and traceroute packets, application programs are effectively cut off from direct "lower-level" access to the underlying physical Internet.

Note: I am FULLY aware that full raw socket-style access can be created by modifying any standard Windows operating systems through the addition of third-party device drivers. I have been a user of such tools for years. However, as I demonstrate below, aftermarket operating system modifications have proven to be irrelevant to the purposes of malicious hackers.

Therefore, as I stated in my DDoS Strange Tale Report, and as I will demonstrate and prove conclusively below **. . .**

Windows' traditional lack of full Berkeley Unix Raw Socket support has been a silent blessing that has undoubtedly contributed hugely to the stability of the global Internet of the past.

It is the Internet's future that concerns me greatly **. . .**

What IS the threat from Full Raw Sockets?

I constructed the diagrams above in the form of insulating layers surrounding the system's network core to help demonstrate that the operating system's IP and TCP/UDP protocol layers serve to protect the Internet from direct access by malicious application software running inside the system.

Any system whose fundamental architecture prevents applications from gaining "Raw" access to the Internet will be MUCH harder to exploit.

As I will show with concrete examples below, these layers of "Internet insulation", which were traditionally provided by Microsoft's "half-baked" Raw Socket implementation, HAVE BEEN A VERY GOOD THING for the Internet so far. But now Microsoft has fatally pierced this insulation by providing full Unix-style Raw Sockets in a high-volume, impossible to secure, consumer operating system.

As proven by the success of the Windows NT server platform (lacking full Raw Sockets support) and the successes of the Internet-connected Windows 95/98/ME platforms (also lacking full Raw Sockets), **full Raw Socket support is absolutely unnecessary for the use of ANY benign Internet applications.** Extensions to the Internet protocols, which represent a valid use for Raw Sockets, would be performed within the operating system's network core. "Sockets" are an application-level interface, not a system level resource, and applications have no valid need for full Raw Sockets. **None.**

In other words, what Microsoft has done with Windows 2000 and Windows XP, is to add a number of powerful and completely unnecessary networking features because, they say, "some people complained about Windows lack of full Raw Socket support". However, it will ONLY be Internet-hostile malicious code that will need to use the advanced "direct access" provided by Windows' new full Raw Socket support.

With Microsoft's traditionally-limited Raw Sockets support, Windows applications were UNABLE to "forge" or "spoof" the machine's actual IP address to hide the source of any malicious traffic they might generate. This source address "spoofing" prevents effective backtracking through the Internet. Windows applications were also unable to generate deliberately malicious "SYN flooding" style attacks, which are essentially unfilterable, and are used to effectively attack any sort of Internet TCP-connection server. (Note that even if the Internet is someday able to block spoofed source IP's, the creation of fraudulent TCP connections, which is not possible using "standard sockets" but is trivial with full Raw Sockets, will remain a problem.)

Until the advent of Windows 2000 & XP, the most common and familiar, complex, potent, and untraceable Denial of Service and Distributed Denial of Service attacks **have only been generated** from Unix-family operating systems. Due to the sheer volume of Windows XP machines soon to be loose in the world, Unix systems will quickly be supplanted as the premiere launching pad for new torrents of Denial of Service floods. This will have an unfortunate corollary effect for XP users:

The huge number of Windows XP machines will motivate hackers to find new ways into those machines — AND THEY WILL. Then users of Windows XP machines will become the most sought-after target for penetration.

In other words, the use of the high-power, mass-market and unsecurable Windows XP operating system, promises to paint a big target on every user of that system.

In the hands of a clueful hacker, fully-supported Raw Sockets is the enabling factor for the creation of a series of "Ultimate Weapons" against which the fundamentally trusting architecture of the global Internet currently has no effective defense.

Windows XP is the malicious hacker's dream come true.

My Initial Dialog with Microsoft . . .

A look at my initial attempts to prevent this **. . .**

Hi Greg,

It's been a while since we've talked.

I'm writing to you first for some navigational direction. Windows 2000 and the forthcoming new MS platforms offer something never before seen in any Microsoft platform: A complete implementation of the Windows sockets RAW SOCKETS specification.

While, as a networking developer, I *love* the idea of having this much power, there is a serious DARK SIDE to this which troubles me greatly: For the first time ever, software running on Windows platforms -- including, presumably, the Home-Targeted Windows XP -- will be able to trivially generate IP packets carrying spoofed Source IP addresses.

Before now, the many tens of thousands of Trojans and Zombies being installed into insecure Windows boxes across the Internet on high-bandwidth connections have been COMPLETELY UNABLE to spoof their source IP's. This has been a blessing, since, until now, only UNIX derived boxes have had complete RAW_SOCK support.

But with Windows 2000, and WinXP, etc. ... Windows applications will be able to forge their "return address" -- which spells catastrophe for the integrity of the Internet.

I would appreciate having you forward this note to whomever should receive it. I need to understand Microsoft's formal position on this before I go off and make a big bunch of noise and draw the world's attention to this impending threat to the operation and security of the global Internet.

Thanks for your time and attention to this matter.

Oh! ... and while I've been intending to mention this danger for some time, I stumbled upon a chunk of hacker source code a few days ago that frightened me a lot:

(Taken directly from hacker source code)

>----------

```
>
> 6. Some words about DDoS from Windows OS.
>    The new feature IP_HDRINCL that comes with win2k can make
>    windows to a powerful DDoS server because it enables IP-
>    spoofing!
>
>    THE IP_HDRINCL
>    setsockopt(ssock, IPPROTO_IP, IP_HDRINCL, (char *)&bOpt,
>    sizeof(bOpt));
>
>    That means win2k-servers can become a base for DDoS that
>    is equal to Unix servers.
>
>-------------------------------------------------------------
```

I sincerely hope that you and Microsoft will sufficiently appreciate the significance of this problem, and the danger it represents.

Please keep me in the cc-chain and let me know what, if anything, transpires.

All the best!

Steve.

Greg was terrific about following up. I received a few progress reports as my note went through "channels". Then a couple of days later I received the following complete reply:

Steve,

I've reviewed your note with the Windows network development architects, as well as our Corporate IT Security and Security Response groups. Your instincts are correct -- they thoroughly understood the nature of the issue ... I guess the response is sort of "good news, bad news" story, depending on your point-of-view. In a nutshell, it will be very much harder to get hostile code running on a Windows XP system than on a Windows 9x, or even a Windows 2000 system ... but a determined hacker still can ... If you'd like to speak to someone about the issue in more detail, I'd be happy to arrange.

Here's the summary:

- Your concern is the ease of spoofing the IP address under which a Windows 2000 or Windows XP system operates on the Internet. You believe that our providing a raw sockets option will make it much easier for malicious parties to develop zombie code that is capable of operating with a spoofed source address.

- You're particularly concerned with Windows XP which, as a consumer system, will be very widely represented on the Internet and operated by naive home users who won't have the time or expertise to take active steps to configure their systems securely.

- Windows 9x and NT systems have for some time offered the capability to send raw packets from the NDIS layer. This capability is just as exploitable for IP spoofing as the IP_HDRINCL API is.

- The real issue here is more the ability to get control of the zombie system than the ease of writing the zombie code that exploits it. If I can get code running on a system, it's pretty much guaranteed that I can write code to exploit it. The issue is only one of how much code I have to load onto the zombie and how hard it is to get it right (neither a trivial issue, but neither a showstopper for the hacker).

- It is much harder to get hostile code running on a properly configured Windows 2000 system than on an Windows 9x system. While proper security configuration takes some work and expertise, we do provide tools and guidelines to help with this task. Because Windows 2000 primarily appeals to technical home users, we believe this is a reasonable balance.

- It will be very much harder to get hostile code running on a Windows XP system than on a Windows 9x, or even a Windows 2000 system. The integrated Internet Connection Firewall will be enabled by default for users who go through the wizard that connects an XP system to the Internet. A number of system default settings have been tightened to make it harder to get code that makes it through the firewall to run (for example, Outlook XP, Outlook 2000 SR2, and Outlook Express V6 all process HTML e-mail messages in the IE Restricted Sites zone).

In summary, our security folks believe that it will be significantly harder to get an army of zombies running on XP systems than has been the case today with Windows 9x. Unfortunately, even if we prohibited sock_raw, determined hackers can go around that restriction ...

Finally, as to the actual "why are providing a raw sockets option" ? , I'm told it's less about "need", and more a response to customer demand for Winsock standard compliance.

As I said, please let me know if you'd like to speak with someone about this. And if you think we are absolutely nuts, then I really would like to know about that, too. I can only assume that you will expose your point of view to your readership on your web site, and if you think we are being insanely irresponsible, then I'd rather hear it from you first, than read it on your site...

Thanks, and have a great weekend!

Greg

I will respond, in detail, to these points Microsoft has raised. But first I would like to raise and respond to some of the other questions raised early in this controversy **. . .**

A Standard by any other Name

Windows 2000 (NT5) was the first Microsoft Windows platform to bring the full "Berkeley Sockets" specification — including Raw Sockets — to WinSock. Until this time, there had been essentially NO APPARENT CHANGE in Microsoft's TCP/IP stack.

In fact, while discussing the OS Fingerprinting capabilities of his well-known "nmap" Internet scanner, nmap's author Fyodor, has this to say about nmap's detection of Windows versions:

> [...] Even with all the tests above, nmap is unable to distinguish between the TCP stacks of Win95, WinNT, or Win98. This is rather surprising, especially since Win98 came out about 4 years after Win95. You would think they would have bothered to improve the stack in some way (like supporting more TCP options) and so we would be able to detect the change and distinguish the operating systems. Unfortunately, this is not the case. The NT stack is apparently the same crappy stack they put into '95. And they didn't bother to upgrade it for '98.
>
> But do not give up hope, for there is a solution. You can simply start with early Windows DOS attacks (Ping of Death, Winnuke, etc) and move up a little further to attacks such as Teardrop and Land. After each attack, ping them to see whether they have crashed. When you finally crash them, you will likely have narrowed what they are running down to one service pack or hotfix.
>
> I have not added this functionality to nmap, although I must admit it is very tempting :).

If you read through Fyodor's fingerprinting description, you will see that nmap is a superlatively sensitive detection tool that can generally sense even the tiniest changes in the implementation of a system's TCP/IP stack. (Fyodor is certainly a "hacker" in the truest and most positive sense of the term.) Yet Microsoft's stack apparently never changed **. . .** until it suddenly changed completely.

Several people have been quoted in the press defending Microsoft's stance by simply stating that "following standards is a good thing". In this context it is important to recognize that "Unix Sockets" is simply a "specification"; it has never been ANY sort of recognized "standard". For this reason, the proper way to view the past situation, is that the traditional Windows Sockets system implemented only as much of the Raw Sockets portion of the full Berkeley specification as was useful and required for Windows application software.

In other words, Microsoft's original "WinSock" was exactly right for a consumer operating system.

I agree that following good and safe SPECIFICATIONS can be a good thing. But it seems to me that blindly following ANY recipe, whether it's a specification or a standard, and lacking an understanding and independent evaluation of its role in the intended application, is tantamount to replacing your own judgement with someone else's. For a well-informed person, abdicating responsibility is not always a good thing.

Perhaps these people have never actually programmed the Windows Sockets system (as I have extensively). There is very little about Microsoft's Sockets — with their wild extensions (which I love by the way) — that follows the Berkeley specification. So the truth is, that either with or without full Raw Socket support, Windows Sockets never has been, and never can be "standard".

Windows 2000 -vs- Windows XP

Other people have noted that Windows 2000 is already out in the world with full Raw Socket support. They seem to believe that my lobbying so firmly against the subsequent release of WinXP — with its similar Raw Sockets — is the equivalent of closing the barn

doors after all of the horses have escaped.

These people miss the essential aspect of "scale". If Microsoft were going to sell only a few thousand copies of Windows XP, I would not be wasting either your time or mine with this entire issue. But whereas Windows 98 and Windows ME have been largely uninteresting upgrades, Microsoft has loaded so many new goodies into WinXP that it will make for a compelling Christmas season.

Microsoft has executed this perfectly: In the near future, Windows XP is going to become THE generic consumer personal computer operating system.

Therefore, my concern is with the **DEFAULT** feature-set of the system and with the probable size of that feature-set's installed base. Sure, I wish that Windows 2000 were **also "Raw Sockets Neutered"** so that malicious hackers could not assume that all Windows 2000 machines were exploitable.

I have no problem with the idea of an after-market add-on download pack from Microsoft, or of making the "deluxe stack" available in a Windows XP resource kit or MSDN subscription.

It is the idea that EVERY CONSUMER MACHINE will have such dangerous capabilities that are NOT NEEDED AT ALL for Internet connectivity, that strikes me as being SO unnecessarily dangerous and . . . ultimately . . . dumb dumb dumb!

Network "Egress" Filtering & ISP Responsibility

Many thoughtful and well-informed people have suggested that the **real responsibility** for stopping these attacks lies not with the behavior of the user and/or their Internet-connected machine (e.g. Windows XP), but with the Internet's ISP's. These people point to well-known and long-established RFC's (Internet standards documents) (RFC 2267) (RFC 2827) and other Internet "Best Practice" documents which recommend that packets carrying spoofed source IP addresses should not be allowed to "egress" (leave) the ISP's locally-controlled network. Such clearly invalid packets should simply be discarded as they attempt to "escape" out onto the global Internet.

The beauty of "network egress filtering" is that each ISP becomes responsible for curtailing the IP spoofing of their own users. As I explain on my (still unfinished) DoS pages, once a forged packet "gets loose" from the ISP, and out onto the Internet, the task of tracking it back to its source is essentially impossible. The only opportunity to "block and drop" a spoofed packet is while it's still within the ISP's local network where it is EASILY identifiable as invalid and forged. Once that packet "egresses" onto the main Internet backbone, it's too late.

Adding Egress Filtering

For many ISP's, implementing egress filtering is as simple as adding a SINGLE LINE to the configurations of their various routers. For example, Cisco routers have included this option for years, merely requiring the addition of this single line:

```
ip verify unicast reverse-path
```

> In most cases, that's all there is to it. However, despite the fact that this has been known and discussed for more than three years (the issue date on RFC2267) the vast majority of ISP's have simply not bothered with this simple security measure.

I believe that proponents of ISP network egress filtering are COMPLETELY correct. I have stated this at the conclusion of my previous page describing the Wicked DDoS Attacks. My announced plans for "Spoofarino", a free, user-oriented utility for encouraging ISP accountability for the lack of egress filtering, has already been discussed by the computer press. Today, the practice of **network egress filtering** is more the exception than the rule, but we can hope that it will be widely adopted as these issues attain increasing visibility in the future.

However, this potential for an improvement in the Internet's infrastructure notwithstanding, it is important to recognize that **. . .**

Egress filtering does NOT solve the whole problem.

While egress filtering will be a good thing once it exists, it fails to solve the problems of Denial of Service attacks in two ways:

- **Local Domain Spoofing:**
  Egress filtering operates by verifying that a packet's "return address" lies within the local network domain. Egress filtering DOES NOT, and can not, verify that the packet ACTUALLY CAME FROM the designated machine within that domain. Since the local network domain being managed by a router often includes thousands of valid IP addresses, any machine may still generate forged packets which appear to be sourced from any other addresses within its immediate neighborhood.

  Therefore, the site under attack will still have difficulty filtering the attack and/or identifying the true attacking machine(s). Rather than identifying and perhaps blocking the individual IP addresses of malicious machines, the inbound routers of a site under attack would need to temporarily ban entire "malicious networks" from access. The effect of this would be that sites under attack would "go dead" to whole regions of the Internet which contain "locally spoofing machines". This is hardly an optimum solution, and even so, it requires a degree of router-blocking sophistication which is uncommon.

- **Non-Spoofing Attacks:**
  The widespread availability of trivial source IP address spoofing is **only one** of the problems created by Windows XP's support for full Raw Sockets. Unlike any previous, unmodified, consumer Windows operating system, Windows XP supports the generation of **SYN-packet floods**.

  The Windows-hosted distributed attacks against grc.com employed 474 machines flooding our Internet connection with ICMP and UDP traffic. We were fortunately able to filter those attacks, and remain on the Web, **only because** those attack-hosting Windows machines were unable to generate SYN-floods.

  Attacks hosted on the future Windows XP consumer operating system will have no such limitations. Non-spoofed attacks, which will never be blockable by egress filtering, will be far more damaging when hosted by Windows XP than previous consumer versions of Windows.

As this analysis demonstrates, network egress filtering is undoubtedly a good thing for

the long term future of the Internet. But it does not, and can not, provide a cure-all solution to the problem of the Internet protocol abuse promoted by the existence of Windows XP's full Raw Socket support.

Microsoft Reacts

After the May 31st release of my widely read DDoS attack report, in which I was strongly critical of Microsoft's publicly stated and confirmed intentions to equip the consumer-targeted Windows XP with full Raw Sockets capabilities, Microsoft produced and publicized a formal response on their TechNet web site. Since then their article has been removed, and I can't publish it in its entirety due to copyright limitations.

So I will summarize what I read as Microsoft's stated position, point-by-point, and reply to each in turn:

Microsoft's Position:

This is not really anything new, since previous versions of Windows had support for Raw Sockets.

HUH?!!
This is a very disappointing position for Microsoft to be taking. Within the present context, they MUST KNOW that this is simply not the truth. The entire debate centers upon the distinction between partial and full Raw Socket support. So I can only presume that Microsoft is hoping to achieve some quick public relations damage-control, even at the **ultimately extreme cost** of sacrificing the truth.

On Friday, June 8th, the TechNet page referenced above states:

> *"**...** as raw sockets implementations are already present in Linux, VMS, Unix, Mac OS X, and even in previous versions of Windows."*

And in their first reply to me, shown above:

> *"Windows 9x and NT systems have for some time offered the capability to send raw packets from the NDIS layer. This capability is just as exploitable for IP spoofing as the IP_HDRINCL API is.*

The following three examples provide concrete evidence of Microsoft's apparent confusion over the issue of full Raw Socket support in previous versions of Windows:

- **Proof #1:**
  My original note to Microsoft quoted from the "readme" file of a Windows DDoS attack tool named Skydance v3.03:

  "The new feature IP_HDRINCL that comes with win2k can make windows into a powerful DDoS server because it enables IP-spoofing!" ...and...

  "That means win2k-servers can become a base for DDoS that is equal to Unix servers."

  Also appearing on that readme page under the section "Client Usage", is the following:

  "The Client will try to use a spoofed source address. You should test your spoofing-

ability first to ensure that you can not be revealed. The test will fail on WinNT and Win9x/Me systems. It should not fail under Win2000."

How much more plain can that be? Windows 9x/ME and WinNT DO NOT HAVE the Raw Socket capability to spoof the machine's actual IP address. This was only added into Windows 2000 and is now being carried down into the consumer market by Windows XP.

Here, located on the "megasecurity" hacker site (provided with their knowledge and permission), is the entire "readme" page for this typical Windows DDoS attack tool. Note that it is not currently useable on any consumer-grade Windows systems **. . .** but Microsoft's XP will soon be changing that, to the delight of malicious hackers everywhere:

http://www.megasecurity.org/trojans/skydance/Skydance3.03.html

I got a personal chuckle out of the last paragraph on that page. It talks about the trouble with the new personal firewalls and suggests that by renaming the DDoS Trojan to the name of a common Internet application (like "ping.exe") — which presumably has firewall permissions — you may be able to get past outbound blocking firewalls by "impersonating" a permitted application.

Those of you who have been following my work at grc.com will note that this was **exactly** the personal security problem I anticipated when I created and promoted my free LeakTest utility. Its goal was to bring market pressures to bear, thereby inducing firewall vendors to prevent this trivial exploit. At the time of LeakTest's release, all but one firewall was vulnerable to this. But today, every reputable firewall has been updated as a direct consequence of LeakTest's influence.

As you might imagine, the personal firewall vendors were as furious with me then, as Microsoft appears to be now. But those vendors are happy today, and their users are much safer.

(So as not to confuse people, I should mention that BlackICE Defender still fails the LeakTest, and would therefore presumably allow this Trojan to operate. However, Network ICE has stated that, despite the declarations on their web site, BlackICE is not a firewall. So it is exempt from the class of products I refer to as "reputable firewalls".)

While you're at the megasecurity site, take just a moment to browse through their catalog of the Trojans which will soon be competing for space on Windows XP hard drives:

http://www.megasecurity.org/trojans/

Impressive.

- **Proof #2:**
  Located on Internet.com's "CodeGuru" site is another of the many typical articles and code samples floating around the Internet which highlights the unique power of Windows 2000 and — soon — Windows XP:

http://www.codeguru.com/network/tcpip_lib31.html

The ZIP file, which may be freely downloaded from that page, contains complete source code for this set of "C++" library functions to leverage the "WinSock" system for the purpose of experimenting with the Internet.

Of particular interest is the included "Attack" program which, according to that program's readme.txt file, currently only runs under "w2k". Why? Because Windows 2000 is the only Windows operating system that currently supports full Raw Sockets. But as we all certainly know by now, this will soon change.

Under the "How does it work?" section, the author explains:

"When I send only the SYN, and spoof the address of a non working address (no host over there to reset the connection), the remote system will never get the SYN+ACK response and it will wait until that connection will time out (around 20 seconds), assume I'll send 60,000 of this sockets, the amount of resource I'll tie up will do some damage."

Under the "Requirements" section, the author explains:

"Currently working only under w2k."

Right.

This author is describing a classic SYN flooding attack using a spoofed Source IP. NO PRIOR VERSION OF WINDOWS allows its applications to arbitrarily generate Internet packets. As this example demonstrates, deliberately invalid — and malicious — SYN packets can NOT be generated unless the application is running on Windows 2000 **. . .** or, soon, Windows XP.

This sample highlights another interesting aspect of Microsoft's poor judgement in this matter:

The threat of attacks is NOT ONLY from surreptitiously installed remote-control Zombie/Bot Trojans, but also from PC hobbyists who will soon be able to gleefully launch untraceable spoofed IP SYN-flooding attacks from the comfort of their own bedrooms. Presumably after finishing their homework.

The typical teenage hacker has not had access to Windows 2000. He or she has been limited to playing video games on Windows 95/98/ME. But this Christmas will change all that: When "Junior" asks Mom and Dad if he can get an upgrade to the new really cool Microsoft Windows XP for Christmas, Mom and Dad will smile and nod. "What a GREAT idea!" they think to themselves.

Yeah. Great.

- **Proof #3:**
  One of the most well known and sophisticated remote-control attack Zombies is named: **Trinoo**. This remote-control Bot was originally written to run on compromised Unix- and Linux-style platforms which, as we all know by now, have traditionally been unique in having the ability to generate spoofed source IP

flooding attacks.

Trinoo was such a successful attack tool over on the *NIX platforms, that it was "ported" to the Windows environment under the name: **WinTrinoo**.

But, of course, WinTrinoo's malicious capabilities are somewhat limited under Windows. Unlike its Unix cousin, **WinTrinoo can neither spoof source IPs, nor generate SYN flooding attacks.** You KNOW that WinTrinoo's authors know how to spoof source IPs and generate SYN floods. They did it for Trinoo. So why doesn't WinTrinoo have the same power under Windows? You know why: Because Windows has traditionally lacked support for the full Raw Sockets specification.

Next year, after **"The XP Christmas of Death"** has passed, tens of millions of home PC's will be happily running Windows XP. How many minutes do you think it will take for **"WinTrinoo2"** to arrive on the scene and for it to take full advantage of XP's Unix-style full Raw Socket support?

A note to the Internet's Hackers: It would be TERRIFIC if you were to name your second-generation WinTrinoo version: **WinTrinoo-XP !!**

So now, in light of what you've just seen, reconsider the intent behind Microsoft's summarized position, as documented above:

"This is not really anything new, since previous versions of Windows had support for Raw Sockets."

What are they thinking up there in Redmond?

I hope it is clear to you, in light of this little bit of evidence (there's an endless amount more), that the currently planned release of Windows XP into the consumer market, represents a crucial mistake. And given that Microsoft is fully aware of this, a shocking example of corporate hubris.

NEXT

This quote is taken directly from Microsoft's TechNet page referenced above:

From the TechNet Page:

"The presence of operating system-level functions to manipulate data packets is not a critical factor in the number of DDOS attacks. If it were, the explosion in DDOS attacks should have already occurred, as raw sockets implementations are already present in Linux, VMS, Unix, Mac OS X, and even in previous versions of Windows."

We have just examined the obfuscation that was apparently intended at the end of that quote. (Regarding the applicability of full Raw Sockets to previous versions of Windows.) I hope you're no longer fooled by that. Let's look at the rest of it.

"If it were, the explosion of DDOS attacks should have already occurred..."

Perhaps Microsoft hasn't been reading about the rapid rise (explosion) in the number of DDoS attacks which is already occurring. One must wonder how they could be unaware of this since they have, themselves, been frequent targets of those attacks. Furthermore, they must know, as I demonstrated above, that the widespread availability of Linux and Unix, with their "system-level functions to manipulate data packets" are **clearly responsible** and are a "critical factor" in the number of DDoS attacks.

It is precisely because of the rapid growth in the number of hobbyist-owned Unix and Linux boxes — often configured insecurely then compromised with Trojans — that we are now seeing a rapid growth in the number of DDoS attacks.

Microsoft is about to massively escalate this problem!

Although it is not completely clear what message Microsoft intended to convey with that quote, what they APPEAR to be saying here is something along the lines of:

"Everyone else has full Raw Sockets, . . . so why shouldn't we?"

Assuming that this is what that quote was trying to say, it raises a good question which is worth exploration:

As we have seen, it is indeed unfortunate that "everyone else" has full Raw Socket support. The Internet has already been suffering the consequences. That problem is certainly going to grow with time and needs to be dealt with as well.

The fact that Microsoft was not the first to make this crucial mistake on the Internet in no way reduces their now-fully-informed responsibility to prevent their negligent compounding of the problem.

The installed-base of consumer Windows operating systems dwarfs that of all other platforms combined. In shipping their Windows XP system, squarely targeted at the home and small office user, tens of millions of existing Windows platforms which have never had full Raw Socket support, will be upgraded overnight into powerful Internet attack platforms. And all new computers sold after Windows XP's release will have that built-in.

If you think the Internet is in trouble now, just wait.

NEXT

From the TechNet Page:

"Nor is the absence of such functions a significant impediment to such attacks. Most modern operating systems allow new functions — including networking functions — to be added via installable drivers. An attacker who had the ability to install

> **zombie software on another user's machine could just as easily install a network driver to provide any functions it needed, including functions to disguise the source address of the attack."**

That is absolutely true . . . and absolutely irrelevant.

Every one of the three concrete examples we looked at first demonstrated that — in actuality — the lack of the DEFAULT AVAILABILITY of full Raw Socket support in traditional versions of Windows, **completely prevented** that malicious tool from gaining access to IP spoofing and TCP flooding capability. We KNOW that they all wanted it. Trinoo has it under Unix and Linux, but WinTrinoo doesn't under Windows. The other program examples apologized that they were only useable under Windows 2000 because of W2K's support for full Raw Sockets.

There is a huge PRACTICAL gulf between what COULD be accomplished in theory, and what IS ACTUALLY ACCOMPLISHED in practice.

Operating system kernel-level "packet drivers" are freely available on the Internet. Microsoft even provides a (buggy) sample of such a driver in their own "Platform SDK" (A sample kit for Windows developers.) But these have existed for years and have never been employed by any of the popular malicious tools. The reason for this is that it has never been nearly as simple as Microsoft makes it sound. Those solutions tend to be operating system version dependent and difficult to reliably install — especially remotely. As a result, and despite what might be possible, all of the evidence demonstrates that malicious tools exploit the interfaces provided by the native operating system.

The addition of full Raw Socket support to the DEFAULT Windows XP consumer product guarantees that the next-generation Windows-hosted tools of mass malicious exploitation will be far more powerful than any previously designed for today's Windows operating system.

It's just that simple. **How could Microsoft NOT see that?**

NEXT

From the TechNet Page:

> **"The real issue is whether the attacker could run hostile code on another user's computer. Like viruses, Trojan horses and other hostile code, a zombie program can only run if an attacker can install it and run it."**

Wrong. The **REAL ISSUE** is that Windows XP puts an operating system in the hands of the consumer which allows ANY SIMPLE APPLICATION PROGRAM — whether installed by a malicious hacker or used by the system's owner — to trivially generate sophisticated source IP spoofing Denial of Service (DoS) TCP SYN floods. None of Microsoft's previous, unmodified, consumer-targeted Windows operating systems allowed this. This is a huge change for the worse.

Remember "Junior" whom we met awhile back when he asked his parents for a Windows XP upgrade for Christmas? Let's take a look at what "Junior" is up to after he returns to school from Christmas vacation **. . .**

The Story of "Junior" and his XP Gang

"Junior" is basically a good student and a good kid who likes computers and trades mp3 music files and Windows programs with his friends at school.

But not long after Christmas, one of his friends finds a cool new program written for Windows XP. If you try to run this program (which is called "NukeEmNow.exe") on the old versions of Windows, which they all had before Christmas, it just says: "You need WinXP to use this safely." and it won't do anything else.

This little Windows program allows anyone to launch a completely untraceable "personal" Denial of Service attack. It launches a protracted SYN flood with spoofed source IPs against any web site the user wishes. When you run it under Windows XP, a window pops up asking its user to enter the URL of the website to be driven from the Net.

A couple of the school's more knowledgeable young computer geeks explain that Windows XP is really cool because, unlike the earlier versions of Windows, XP lets you "Spoof" your computer's IP address to make you completely anonymous and invisible when attacking others on the Internet. The geeks caution that the program should only be run from a diskette, never stored on the hard drive, so that no evidence of its use is ever left behind. Since Windows XP has all the fancy full Raw Socket support built right in, the "NukeEmNow.exe" program, which was written in Visual Basic, fits easily on any spare diskette.

Of course, this program didn't exist and wasn't written before XP, because it wasn't possible under any traditional versions of Windows. But now many such programs are popping up all over the Internet. All it took was the new release of Windows.

Now, after school every day, "Junior" and his close friends — all with fresh Windows XP Christmas upgrades and cable modems — meet behind the Gym to decide whom they want to punish with their own little neighborhood-wide Distributed Denial of Service attack.

Feeling a bit like super-cool spies, glancing around stealthily, they synchronize their watches and each head to their respective homes, ready to click the "Launch Attack" button at the appointed time.

It is a virtual certainty that applications such as the hypothetical "NukeEmNow.exe" will be written for Windows XP, and that those applications will be used by malicious individuals of ALL ages. (I didn't mean to single-out teenagers.)

It is worth noting that since TCP "SYN" packets are extremely small (60 bytes) compared with data-carrying packets (1500 bytes), many more SYN packets can be sent per second than data packets. This gives a SYN flooding machine more "packet potency"

than one which is attempting to transfer valid data. The consequence of this, is that a **single SYN-flooding machine** can completely knock out any other machine connected at the same or lesser speed. Coupled with Windows XP, and a breed of cyber-war toys like the still-hypothetical "NukeEmNow.exe" described above, we can expect "one-on-one" cyber-battles between individuals. If someone doesn't like what someone else says or does, they are too easily blown off the Internet.

Let me say it again: This is all COMPLETELY UNNECESSARY since no Windows applications have ANY need for full Raw Socket support. No VALID use exists outside of an Internet research setting. Raw Sockets were only included by the original Berkeley designers for Internet protocol research. In a consumer computer system, they will only be exploited for malicious purposes.

So, as we see, the "real issue" (to quote Microsoft) is NOT whether the attacker can run hostile code on another user's computer. I submit that the "real issue" is whether a personal computer can be **much too easily programmed** to generate untraceable and maliciously damaging Internet traffic. Until now, for Windows, that answer was no, and as a direct consequence it was never done.

Windows XP flaunts its ability to trivially generate malicious traffic. You already know what will happen.

"Microsoft Security"

Here is a simple fact:

It is absolutely impossible to create a secure, consumer, personal computer.

Security is black and white. Either you are secure and protected, or you're not. Strange as it might seem at first, I don't blame Microsoft for their demonstrated inability to build a perfectly secure personal computer. After all, it's not possible. But I do hold Microsoft responsible for continually marketing and selling something they can never produce. And they MUST be held responsible for the consequences of believing their own marketing and press.

Ask any real security expert, like Bruce Schneier of Counterpane Internet Security. They will tell you flat out that it's an impossible task to secure a personal, consumer, computer. Why does Microsoft continually insist otherwise? Because it is what people desperately want to hear, and desperately want to believe. **Well . . . it's not possible.**

Microsoft's software has NEVER been secure, and NEVER will be. With each generation of feature-rama upgrade, it becomes more and more complex, and less and less understandable. There can not be anyone left at Microsoft whose mind can still grasp the technical details of the entire system. They had to give that up with MS-DOS. Microsoft's lack of security foresight is single-handedly responsible for creating the eMail virus. Their consumer operating systems — as well as their high-end server platforms — are notoriously insecure.

Just two weeks ago (May 23rd, 2001) . . .

Windows MediaPlayer Security Fixes

Microsoft has released security fixes for its audio and video player. ***The problem could allow hackers to run any code they want on your machine.*** The security advisory has details for the technically inclined. Media Player 6.4 users should install a patch which has been posted on its security website. Users of Media Player 7 should install the latest Windows Media Player 7.1, which is available at Microsoft's media website.

7.1: http://www.microsoft.com/windows/windowsmedia/en/default.asp
Bulletin: http://www.microsoft.com/technet/security/bulletin/MS01-029.asp

Whoops.

So the question is **. . .** Have **YOU** installed the appropriate patch described above? Oh, you didn't know about it? Gee. Microsoft now says that any hacker could run any code they want on your computer. This sort of thing is a DAILY OCCURRENCE with Microsoft and Windows. Sure, it's a daily occurrence because Microsoft and Windows are the biggest targets, but this also means that they have the biggest responsibility.

But, patching Windows doesn't always work either **. . .**

From the SANS Institute NewsBites service

13 & 14 June 2001— Exchange 2000 Patch Woes

The first patch Microsoft issued for an Exchange 2000 security flaw contained an error that caused servers to hang. The second, which contained outdated files, did the same thing. The company released a third version of the patch last week. One security consultant described the patch's effect as essentially launching a denial-of- service attack on one's own server.

The ZDNet Story The ComputerWorld Story

And then there's what the hackers know **. . .**

From the SANS Institute NewsBites service

13 June 2001— Cracker Group Defaces More Sites Because They Can

A cracker group notorious for its defacing scads of Chinese web sites earlier this year has recently defaced a dozen sites worldwide; all the sites have in common is the word "security" in their domain names. In an email to CNET, the group claims that they target Windows NT and 2000 servers because they are so easy to infiltrate.

How, then, can anyone accept Microsoft's defense for adding incredibly exploitable and utterly unnecessary Internet technology into their base consumer-level system as: "Don't worry, THIS one will REALLY be secure." ??

You MUST KNOW that not long after its release, the world will begin finding huge security holes in Windows XP. Oh sure, Microsoft will issue patches. Then the users will

be blamed for not installing them in a timely fashion. What's WRONG with those damn users anyway?

And even if, against all logic and our wealth of experience, you're on the fence with this question **. . .** WHAT IF MICROSOFT IS WRONG? There is just too much riding on the issue of the security of this completely unproven new system. We lose NOTHING if I am wrong about this and if Windows XP has its full support for Raw Sockets removed. But **. . .**

What if they are wrong?

Speaking of which, this just in . . .
June 11, 2001

Office XP Cracked Already

I was just sent the following email. Name withheld on request:

" I find it amazing that MS put so much effort into forcing registration and locking down Office XP to prevent piracy and yet just yesterday I was given a copy of Office XP pro with Frontpage and it was "Cracked". Now while I don't use pirated software nor do I have the need to do so. I did have to install it to see if what I was told about the CD was true. And I'll be darned if it wasn't. I put the CD in and the MS Installer kicked in asked me for a long CD Key and BOOM it went about its business. I filled in my installation choices I chose upgrade 10 minutes later I was running all of the applications. In the MS Office tools menu I saw the option to activate my software I clicked it and I got a Message state my products were already activated. I tried this online and offline, win 98 and win2k everything went flawlessly. Now I have removed the pirated software from my machine **. . . "**

As I said, it is impossible to create a secure consumer personal computer.

The Third-Party Joke

And EVEN IF Windows XP shocked the world by turning out to be secure, that will last ONLY until the myriad of Windows applications start loading themselves into the system.

How many people complain that the annoying "Comet Cursor" keeps getting installed into their computer whenever they visit certain web sites? That's Comet Cursor's CODE being downloaded and run without the user's knowledge or permission. Microsoft's default web browser settings — which the typical user uses without a second thought — allows all manner of similar remote web-based exploits. What happens when a Windows XP user innocently surfs to a site that was set up to take over those machines?

When I reverse-engineered the Aureate/Radiate advertising Spyware, to create the OptOut spyware detector and removal tool, I mentioned that it would be trivial for any malicious hacker to commandeer the Aureate Trojan and cause it to do their bidding.

One line added to an innocuous and unprotected file, will cause any of the more than 30 million Aureate Trojans to "phone home" to a different server. From there it's trivial to have the Trojan accept a file to download and then run it. And if all that's not worrisome enough, the Aureate Trojan undetectably runs within the Internet Explorer browser process; this lets it slip past the system's firewall by trading on the browser's Internet access permissions. So much for Windows XP security.

As any of you who run a personal firewall with "noisy logging" know, routine scans for the PC Anywhere remote control utility are STILL occurring on the Internet. Why? Because people installed PC Anywhere in their machines to give them remote access across the Internet. The only problem was, a great many of these people never bothered with a password. Thus, anyone could scan the Internet to find a machine running the "PC Anywhere" Trojan and "own it". So much for the "security" of that machine.

As you can see from these examples, the goal of an "absolutely" secure personal computer for the masses is impossible to achieve. It's true that exercising extreme care and caution can result in "a more secure PC". But that can only be achieved in degree, never absolutely. For example, even Windows 95 could be quite secure if the user were careful about its configuration and diligent about its use. But the average consumer can not be expected to appreciate the subtle and complex nuances of Internet security — especially when being stalked, tricked, and seduced by malicious hackers. Typical consumer computer users will tend to do insecure things. There's no practical means to prevent that, since that's what they want to do. More than anything else, personal computer users want freedom.

Windows NT and 2000 are supposedly "secure" operating systems, yet malicious Russian hackers have been breaking into those machines left and right, then stealing consumer credit card data. Has anyone ever supported the contention that WinNT and Win2000 are immune to viral infection? Has anyone ever contended that? I've never heard any such thing because we all know it's ridiculous.

We all know that no Windows system is inherently safe or secure. The truth is, that can NEVER change due to the customer-base these systems have been built to satisfy.

Yes, I'm Finally Finished

It is apparent that, for unfathomable and never articulated reasons of their own, Microsoft is DETERMINED to ship an inherently unsecurable, consumer-targeted operating system, containing the openly accessible Internet research interfaces known as Full Raw Sockets.

I think that's really dumb.

Denial of Service Pages

Denial of Service Home Page
The Tale of Our Investigation
Denial of Service Attack Log
The Windows XP Internet Threat
The Microsoft Security Oxymoron
Microsoft Laughs Off XP Security

Brief XP DoS Threat Summary

Last Edit: Mar 15, 2005 at 12:05 (865.36 days ago) Viewed 80 times per day