Global Information Assurance Certification Paper

Erik Plaggenmarsch
GCIH Practical v3

Remote Exploit of a Local Vulnerability

The Linux Kernel do_mremap Function VMA Limit Local Privilege Escalation Vulnerability
CVE# CAN-2004-0077

GIAC - Certified Incident Handler
Practical Assignment Version 3
April 2004

1. Statement of Purpose 4
1.1 Introduction 4
1.2 Should we be worried about local vulnerabilities then? 4
1.3 The layout 5
2. The Platforms/Environments 6
2.1 A company in economical hard times 6
2.2 The model employee, working from home 8
2.3 The attacker and his virtual network 9
2.4 The dissatisfied employee 10
2.5 Putting it all together 11
3. The Exploit 13
3.1 Exploiting the employees PC 13
3.2 The Local Exploit 14
3.2.1 Name 14
3.2.2 Operating Systems involved 15
3.2.3 Variants 16
3.2.4 Signatures of the Attack 16
3.2.5 Introduction to the exploit: Linux Memory Management 17
3.2.6 The vulnerable function: do_mremap 19
3.2.7 The exploit 21
3.2.8 Advanced programming 23
4. Stages of the Attack 24
4.1 Reconnaissance and scanning 24
4.2 Exploiting the Employees System 25
4.3 Exploiting the internal system 27
4.5 Keeping Access 29
4.6 Covering Tracks 30
5. The Incident Handling Process 32
5.1 Preparation 32
5.2 Identification 32
5.3 Containment/Eradication/Recovery 35
5.4 Investigation 36
5.5 Lessons learned 39
5.6 Advices to improve security 41
5.6.1 Remote Policy Enforcement 41
5.6.2 Limit Split Tunneling 42
5.6.3 Strong and/or Central Authentication 43
5.6.4 Maintenance windows 43
7. References 45
Appendix A – Vulnerable Operating Systems 47

1. Statement of Purpose

1.1 Introduction

Two different vulnerabilities were found in the do_mremap function in the heart of the Linux operating system, both published in the first two months of 2004.

The two bugs are totally unrelated but are about the same internal kernel interface code:

CAN-2003-0985 - Function Boundary Condition Vulnerability
- http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0985
- http://isec.pl/vulnerabilities/isec-0013-mremap.txt

CAN-2004-0077 - VMA Limit Local Privilege Escalation Vulnerability
- http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-0077
- http://isec.pl/vulnerabilities/isec-0014-mremap-unmap.txt

Both vulnerabilities are still candidates on the Common Vulnerabilities List (CVE) and are awaiting inclusion in the list.

This paper will focus on the second vulnerability that was disclosed on the 18th of February 2004. The two vulnerabilities only differ in the way they gain full privilege on the local systems, but the incident handling process is exactly the same for both. Once local access on a system is achieved, the vulnerability can be used to gain full access or effectively bring the system to a crawl or even shutting it down.

1.2 Should we be worried about local vulnerabilities then?

But how can remote attackers from outside a company exploit this vulnerability if it requires local access on a machine on the internal network? And of course, everything will be done by administrators to prevent intruders from getting access to systems on the corporate network! So, is it not easier to find a remote vulnerability which gives an attacker automatically full access on the target machine? Perhaps it is, perhaps it is not.

In this paper I will try to show the dangers of underestimating local vulnerabilities, by describing a possible scenario on how an attacker could gain access to a machine with limited privileges after which an “upgrade” to full privileges will be done using this vulnerability, allowing him to retrieve a copy of a database with sensitive information.

In this case, it will be done, because the attacker gains control over the desktop of an employee that works from home, or even simpler. But how about grumbled employees in the office, that have direct access to corporate machines? The last possibility will not be discussed into depth in this paper, but will have many similarities with a part of this incident handling case.

1.3 The layout

To better understand the exploits used by the attacker, I will first start to explain the network layout of the attacker, the employee working from home and the company. This will be done in the following chapter

After that the exploits used in the attack will be described, mentioning but not explain the exploits used to gain control over the desktop of the employee. Instead it will focus on the local vulnerability in do_mremap and why this vulnerability can be exploited.

Next, the attack will be described from the point of view of the attacker, followed by the point of view from the Incident handlers in the company, ending with advise on how to improve security to prevent successful attacks abusing the VPN network and local vulnerabilities.

2. The Platforms/Environments

2.1 A company in economical hard times

Figure 1 below shows a typical setup for a company network that allows its employees to remotely access the corporate network. As most of the employees are allowed, and easily can, work from home they all received an ADSL connection, paid for by the company, so they can access the Internet any time a day, via the corporate Virtual Private Networking (VPN) Server, which allows tunneling traffic encrypted over the Internet between the desktop of the employee at home and the corporate network. This way, employees will be able to connect to applications on the corporate network, as if their desktop was connected directly to it.

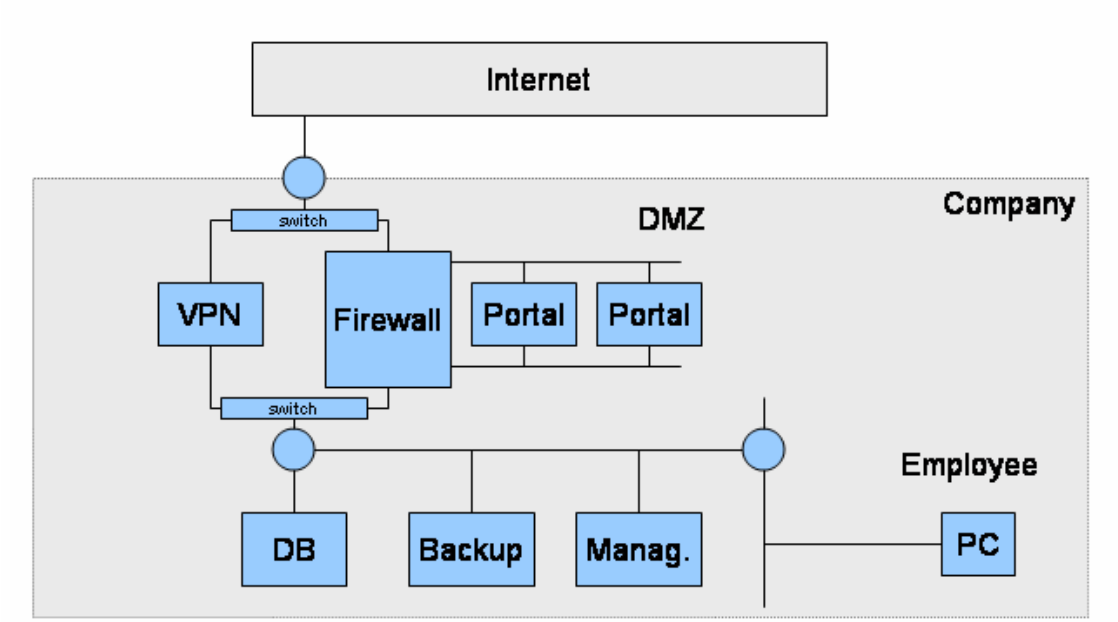

Figure 1. Company network

Compared to old fashioned dialup connections, VPN connections can significantly lower the costs of working from home while at the same time increasing productivity because the bandwidth is higher and the latency is lower. If a percentage of the employees can work from home, the company can even save on workplaces in the office! Therefore it is justified to pay for the “always

on" connection to the Internet for all employees that like to (and can) work from home!

The VPN server used is a Nortel Contivity 600 (ref. [1]). The server is configured to allow employees to connect to the Internet/LAN as well as the internal company network at the same time, by enabling 'split tunneling'. This is one by changing the routing tables on the remote desktop; all traffic is routed to the Internet by default and only the subnets of the internal network are routed to the tunnel interface. This way, employees can work on the internal network and can browse the Internet at the same time.

In these economical hard times, companies like to reduce costs any way possible. So to further reduce costs, the company created a webserver on the demilitarized zone (DMZ) of the firewall, running web applications for third parties (customers). This server has a connection to the internal corporate network and to the Internet, but will never allow customers directly on the internal network. The portal and its backend databases are critical for the company and should be up and running 24 hours a day, 7 days a week. So patching will only be done if this is an absolute necessity! As it is so critical the load is shared between to hosts, if one goes down, the other will take over automatically. The performance loss will be noticeable, but business will continue.

The portal servers are protected by the firewall and only HTTPS is allowed from the Internet to the servers. Only database connections are allowed from the portal servers to the database server on the internal network. SSH is allowed from the internal network management server to the portal servers; SSH from the Internet will not be allowed by both the hosts and the firewall. The portal server is very well hardened, so only patches for Apache, SSL and SSH are installed. All other patches can only make the system unstable, or in case of kernel updates, even force the system to go down for a reboot. So unless a remotely exploitable vulnerability is revealed, these systems are hardly ever patched.

The same patching policy is applied to the database server on the internal network, even less strict because it is on the 'safe' internal network! The database on this server stores product information and customer information, including credit card information. Just to be on the safe side, the configured encryption of sensitive data by the application storing information in the databases. It only allows incoming database connections from the portal servers, backup connections from the backup server and SSH connections from the management server. So why worry about patches if the system is stable and downtimes should be avoided at all costs?

Even worse from a security perspective, why worry about local exploits on these systems if no one is able to enter except for trusted employees?

Of course Linux (Debian 3.0, kernel level 2.4.23) is used for all servers on the DMZ as well as the internal network, because it is very stable and for free! Only the desktop systems of employees run MS Windows XP.

2.2 The model employee, working from home

The company stimulates employees to work from home if possible so the office can be kept smaller, and employees don't need to go through the traffic jams every day. To make this more attracting all employees receive an ADSL connection at home. So did our model employee (emp03). He already owns a PC running Windows 2000 which he will be using to remotely connect to the office as well as for personal use.

Our model employee is using this opportunity and created a small home network (Figure 2) as his wife and kids also like to use the ADSL line to surf the Internet while he is at work till late at night. The network consists of two PC's, the Windows 2000 PC for the office and a PC running Windows 98. The two PC's are connected to two ports on the ADSL modem/router, a Speedtouch 610 Multiport (ref. [2]).

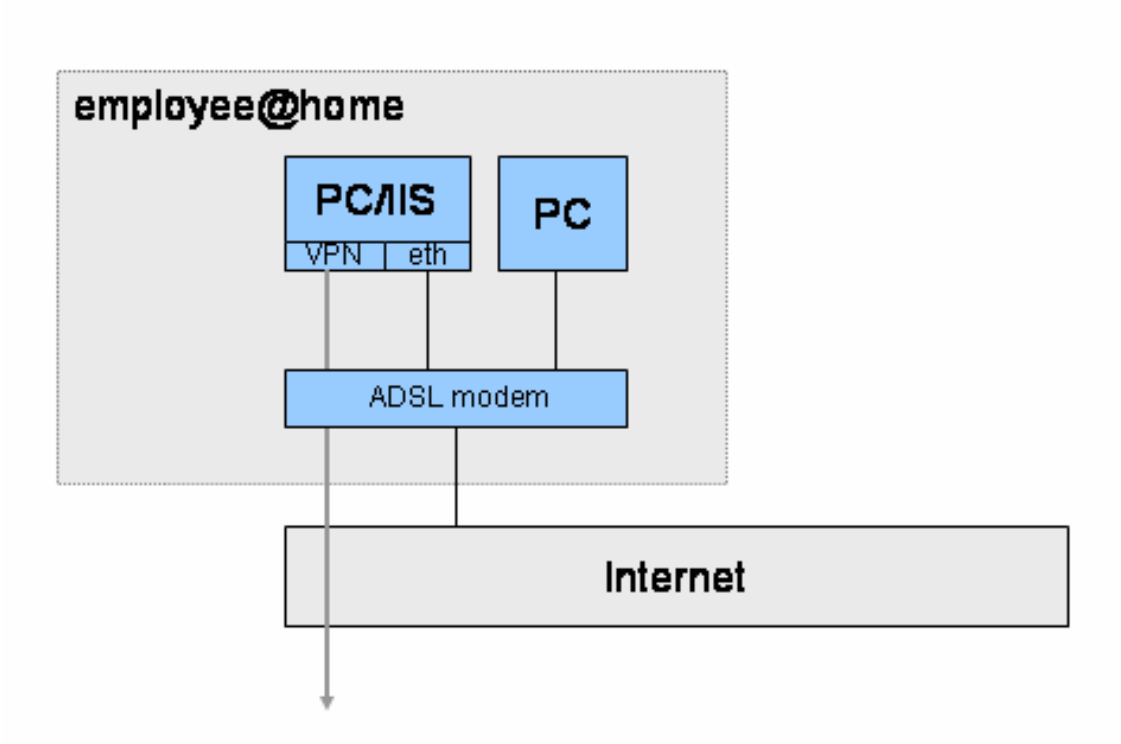

Figure 2. Employee at home

When our model employee created his network he nicely updated his machines and also ran windows update on a regular basis. But then peer-to-peer networks were discovered and downloading movies started to take place. Rebooting machines caused all connections to be lost, so windows update was turned off. A couple of weeks later he installed IIS and never bothered to update that either.

As the employee just bought a new digital camera he was in desperate need of some web space. Having a fixed IP address on his ADSL connection, he decided to install a web server on his own PC as well, keeping it running day and night. This webserver was Internet Information Server 5.0, easily installed from the Windows 2000 CD that came with the system.

2.3 The attacker and his virtual network

The attacker decides for whatever reason (money, just wasting time, proofing to be elite) to try to penetrate the company network and steal sensitive information. At home he has an ADSL connection with a fixed IP address and knowing the dangers, he shielded his home network from the Internet by putting a firewall between his ADSL router/modem and his home network. This machine is a dedicated firewall running Debian Linux with iptables [3] turned on, which gives our attacker full control over incoming and outgoing traffic. The latest patches are always directly installed of course and a scan from his IP address from the outside world would not even show that his host was online!

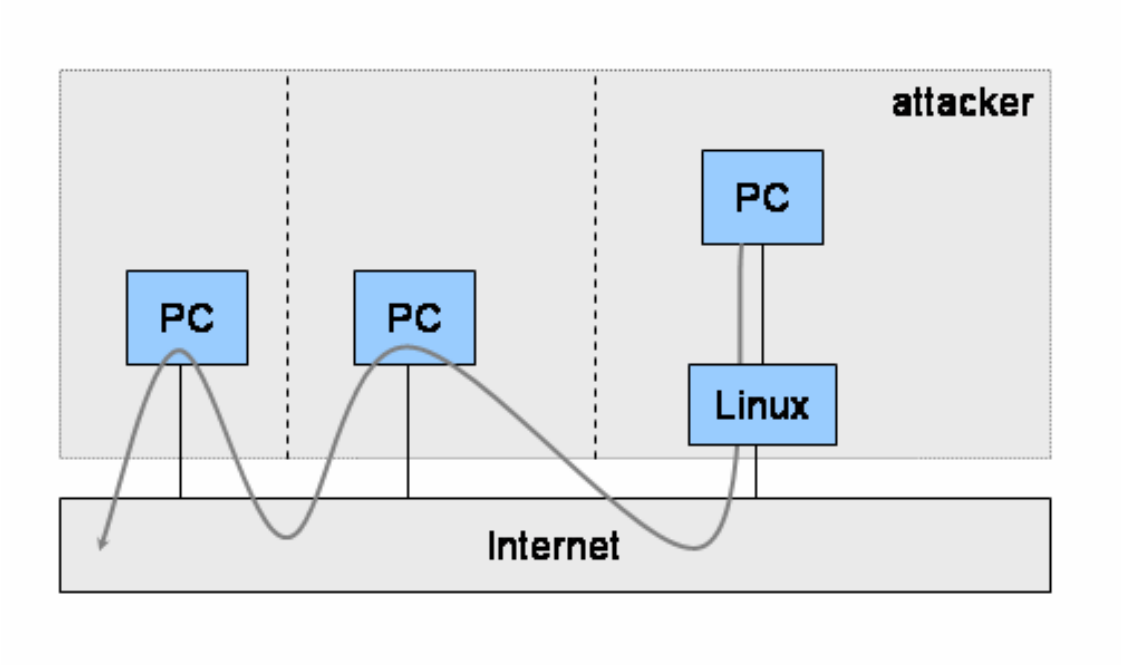

Figure 3. Attackers virtual network

To cover the tracks of his attacks, he gained control over a number of Windows machines on the Internet, in different countries, see Figure 3. Doing so was pretty easy because a lot of machines were running an unpatched IIS (just like our employee just started doing), for example in Internet café's in the Asia and South America. But also many machines are infected by worms like Sobik.F, Netsky and Code Red. Worms like these often install open TCP proxies or backdoors, allowing anyone to relay traffic via this machine to other machines on the Internet. A simple scan of some subnets on the Internet often reveals many infected machines that can be used as a relay to cover tracks of illegal activities, like attacking our company!

So instead of attacking the corporate network directly from his home PC, the attacker will use multiple machines on the Internet as stepping stones, creating a virtual network across a number of countries. Every machine he uses as a stepping stone makes detecting him a bit harder, because the Incident Handler needs to talk to Internet Providers from many different countries and cooperate with many different police forces and justice systems to find the attacker.

2.4 The dissatisfied employee

Another type of attacker could be a dissatisfied employee, already on the corporate network (Figure 4).

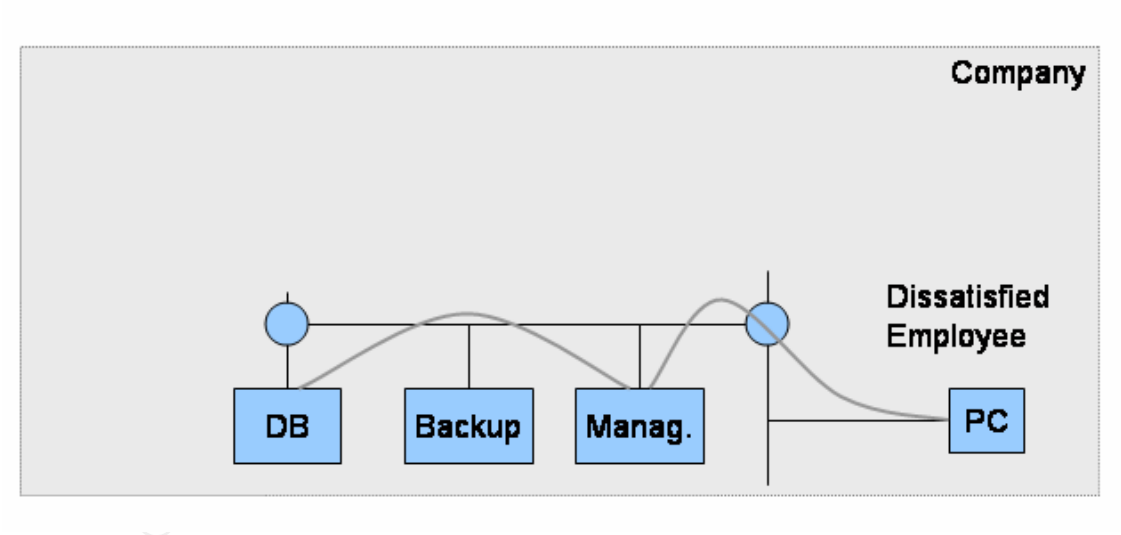

Figure 4. Employees on the corporate network

If your attacker is an employee with hard feelings towards the company, the main obstacle is already passed: the firewall. This employee might even have rights on machines that are used for management of the critical systems, or on those systems themselves. If the employee has full access to these systems (like an administrator) no exploits are needed to do a lot of damage. But also end-users

can cause a lot of trouble on unpatched systems as we will see further on in this paper. The attacker described in this paper will get the same rights on an internal system as a normal employee has. The steps taken from then on by the attacker could have been taken by an employee. As these have more knowledge and possibilities to get data of the network, they might be even harder to detect!

This paper will not discuss the damage that dissatisfied employees might do, but will instead focus on external attackers.

2.5 Putting it all together

If we add the networks of our attacker, the model employee and our company together in one picture, we get the following network diagram (Figure 5).

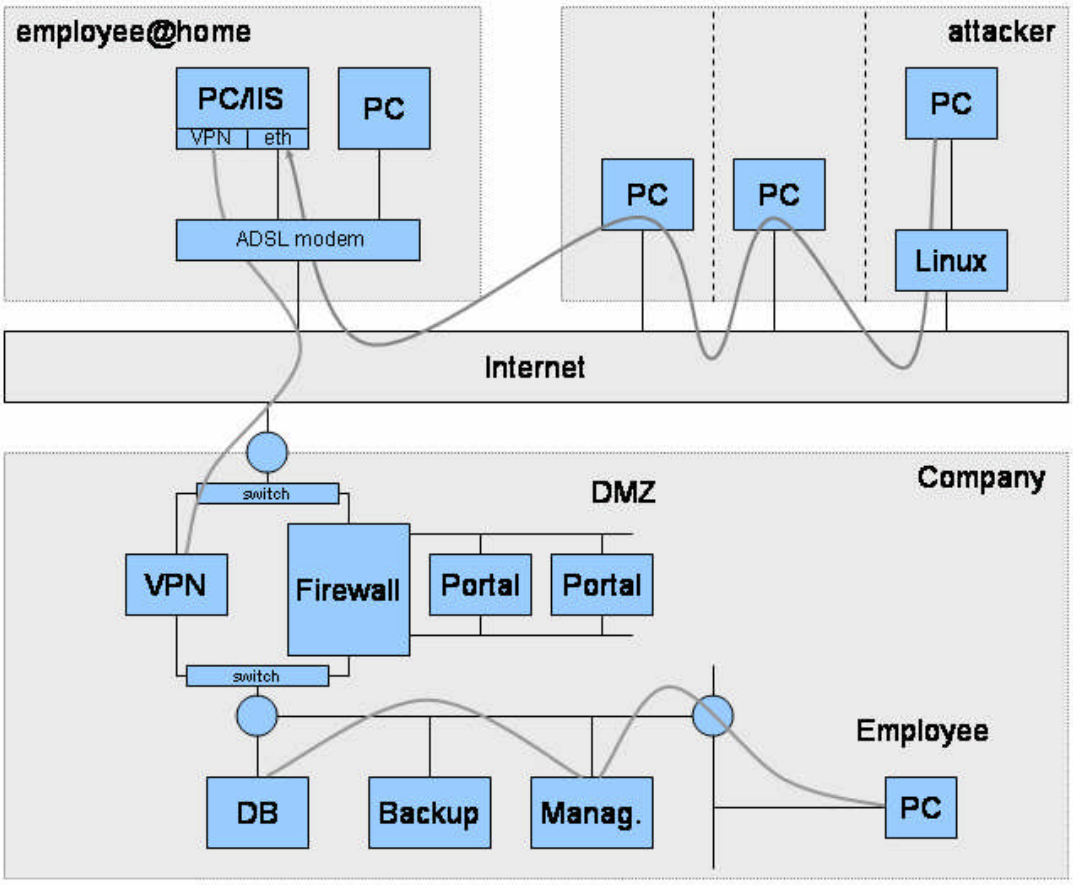

Figure 5 Total Network Diagram

Now we can see how the three different networks link together to allow for a successful, though unwanted attack on the company via the Internet. The

corporate network is very well defended by its Firewall and VPN server, making it virtually impossible to attack the network that way. But the company's internal network is extended by using the VPN server to allow employees to connect from their home. This home LAN is outside the firewall and the security of this LAN is left to the employee, and because of circumstances is very weak. This gives the attacker a way to penetrate the corporate network, by attacking the home network of the employee and either getting his passwords to setup his own VPN connection directly to the corporate network or simply routing traffic via the VPN tunnel to the company's internal network!

What exploits are used to do so, will be described in the next chapter.

3. The Exploit

The attack described in this paper relies on several exploits that allow the attacker to penetrate the corporate network, but do not give him full control of any host on the network. I will quickly describe the exploits necessary to penetrate the network, but not in detail as the main focus will be on the local exploit that allows the attacker to gain full privileges.

3.1 Exploiting the employees PC

To get access to the corporate network, the attacker does not need to hack its way through the company firewall or servers on the DMZ. In a good network configuration, these servers should be very well protected, making it nearly impossible to penetrate if kept up to date, unless a zero-day exploit[1] is used of course. But in this case the attacker will first gain access to the PC of an employee working from home, and then entering the internal network via the VPN connection.

This can be done, because the employee did not update all components on his workstation, leaving holes in the system that could be exploited. For one, he is running an IIS server to host photos to link to from a photography forum where he is a member of. This server is known to be easily exploitable, and the attacker will use the Directory Traversal Bug in IIS 5.0, described in Microsoft Security Bulletin MS00-057 (ref. [4]). This vulnerability allows a remote user to execute a command shell on the system, but with very limited privileges, not enough to fully gain control on the system. Using the (default installed) tftp client, it is very easy to download any other exploits to this machine. This is left as an exercise to the reader, although a very clear how-to presentation can be found in the references chapter as ref. [5].

As the system is running Windows 2000, a bug in the debugging subsystem can be exploited to gain full system privileges. I downloaded the exploit and tested it successfully by starting the command interpreter (cmd.exe) with full system privileges. Info on the exploit can be found on the Securiteam website (ref. [6]). This vulnerability seems to never be acknowledged by Microsoft though.

As of this point, the hacker will have full system access on the employees home PC, allowing him to install network sniffers, key loggers and backdoors at will. See for a more detailed description, the attack in the chapter 4.

[1] Zero-day exploits are exploits, based on vulnerabilities discovered by hackers, which are never publicly communicated and therefore not known to the public and security officers. No patches can exist against zero-day exploits, as we do not know what the vulnerability is. Very hard to detect what the vulnerability is as well, unless the exploit itself is found and examined carefully.

How the attacker gained access to different machines on the Internet, just as stepping stones to the machine of the employee, is outside the scope of this paper. I will only repeat that many viruses install backdoors that can be used by anyone scanning for them. A quick scan on the Internet will easily reveal a number of such hosts, which can be used to hide the tracks. Scanning for BO2k (ref. [7]) for example can be done with the following nmap command:

```
nmap –P 31337 –sS 80.1.0.0/16
```

This will scan the 80.1.0.0 – 80.1.255.255 network using a hard to detect scan method (stealth scan), looking for services listening on port 31337, well know for the BO2k backdoor (or Remote Administration Tool as the authors like to classify it).

3.2 The Local Exploit

A local vulnerability has been published on the 5th of January 2004 concerning the “do_mremap” function on Linux. This function can be used to expand (or shrink) an existing memory mapping, potentially moving it at the same time. A bounds checking issue can cause disruption of the kernel, allowing an attacker to escalate privileges on the system, or even gain root privileges (full access).

About one month after that, on the 18th of February 2004 a second vulnerability was found in the same “do_mremap” function, which was totally unrelated to the previous one! It concerned an error in the checks on the return values performed in this function, again allowing an attacker to escalate privileges and even become root (full privileges) on the system. I will describe this second vulnerability and an exploit for it in more detail. This vulnerability will be used in the attack described in this paper.

3.2.1 Name

This vulnerability is known as:

Linux Kernel do_mremap Function VMA Limit Local Privilege Escalation Vulnerability

Original publisher and discoverer: Paul Starzetz <ihaquer@isec.pl>
http://isec.pl/vulnerabilities/isec-0014-mremap-unmap.txt

CVE: CAN-2004-0077
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-0077

BUGTRAQ: 20040218 Second critical do_mremap() bug found in all Linux kernels
http://seclists.org/lists/bugtraq/2004/Mar/0003.html
http://www.securityfocus.com/archive/1/354284

Linux Kernel Mailing List:
http://marc.theaimsgroup.com/?l=bugtraq&m=107711762014175&w=2

Security Focus Vulnerabilities list
http://www.securityfocus.com/bid/9686

SecuriTeam:
http://www.securiteam.com/unixfocus/5TP030KCAY.html

The exploit used in this paper is from the original description of the discoverer Paul Starzetz, his Proof Of Concept code. The program is called “mremap_pte.c” and can be extracted from the paper of Paul Starzetz. No copy is added to this paper as the source code is copy protected and cannot be printed/published without explicit written permission from the author.

3.2.2 Operating Systems involved

This vulnerability only involves the Linux operating system, as this is the only one implementing the do_mremap function. 4.2 BSD used to have an mremap function, but this was never fully implemented and had a whole different purpose.

This vulnerability is found in many kernel versions, but is not exploitable with the exploit described below for all these versions.

Vulnerable kernel versions are:

Linux version 2.2 up to and including 2.2.25 Linux version 2.4 up to and including 2.4.24 Linux version 2.6 up to and including 2.6.2

Exploitable kernel versions via the exploit “mremap_pte.c”, written as a Proof Of Concept, by the original discover are:

Linux version 2.4.19 up to and including 2.4.24 Linux version 2.6 up to and including 2.6.2

Versions 2.2 and most pre 2.4.19 are not exploitable with this exploit, as it uses the page table cache that was introduced in 2.4 and enabled by default in 2.4.19.

A list of know operating systems that are vulnerable can be found in Appendix A, taken from: http://www.securityfocus.com/bid/9686/info/. On this list are all well known Linux versions like: RedHat, S.u.S.e., Debian, Mandrake, Slackware, etc..

3.2.3 Variants

As of this moment, various Proof Of Concepts are available, but most of them do not allow exploitation of the vulnerability. They just show whether the system is vulnerable or not and in some cases might even cause the machine to crash. As the proof of concept written by the original author will run on (probably) all of the vulnerable operating systems, most hackers do not need to write their own new exploit. The exploit will work on most systems as it only requires /bin/ping to exist and have the suid[2] bit set, and it will need a shell on the system to start with root privileges. In case the exploit is run on a system where ping is not available, any executable with the suid bit set (for example /bin/login) can be used, by giving it as an argument on the command line.

Other ways to exploit this vulnerability probably exist, but no known exploit is available at the time of writing. For example, mapping into memory of files owned by root could be abused, with /etc/password and /etc/shadow as interesting targets.

3.2.4 Signatures of the Attack

No IDS signatures are available for this attack, as it is a local exploit, run from a system and not remote via the network. The only thing that could be detected by an IDS system, would be strings in the compiled version of the exploit, if it was transferred via FTP or another unencrypted protocol to the host. But signatures like that are not available (yet).

As a security architect put it on the Neohapsys snort mailing list:

> Snort is a network IDS, the mremap and do_brk are both local vulnerabilities - so no, there are no signatures for them.

Unfortunately, running the attack on a remote host will not leave any traces in any of the system log files either! The exploit will execute the ping command and will start a shell, both not leaving traces in the system log files. As no errors are generated during the execution of the exploit, no errors will be stored either.

The only trace left might be the fact that a file is copied with scp (for example) to the system (if this is logged which is not the case on most systems), and that this file is then executed. Also traces could be found in the history files of both the employees account and the account of root. But, a smart attacker will give its exploit an obscure name that will appear to be a normal Linux command. For example, the history file might look like this:

[2] In UNIX and Linux, an executable sometimes needs full privileges in order to be run. By settings this bit in the permissions of the executable, root can allow it to run with full privileges. This is normally only done for operating system tools like 'login', 'ping', 'su' etc…

```
..
export OLDPATH=$PATH
export PATH=./:$PATH
..
..
ls /bin/ping /bin/sh
..
..
export PATH=$OLDPATH
..
```

This will not look very suspicious, even on a close look. But if the exploit is named ls (instead of 'mremap_pte') it will still run with the two optional arguments, being the suid program and the shell to execute as root. As these arguments are also the defaults if no arguments are supplied, the command could even be just 'ls', making it even harder to detect! The only thing suspicious is the fact that the path is changed to run the ls command from this directory instead of the normal /bin/ls command. Obscuring the attack even more, a simple shell script can be created and run instead of running the executable (in this case called test2.sh) directly:

```
..
vi test.sh
chmod u+x test.sh
./test.sh
rm –f ./test*
..
```

Of course, if the hacker cleans the history file after every visit, finding traces might be virtually impossible as no traces are left behind. Cleaning history files and other possible ways to detect suspicious activities will be discussed in chapter 4.

3.2.5 Introduction to the exploit: Linux Memory Management

To be able to understand the exploit, we need to understand the memory management process in Linux and how processes see and access the memory. References used to write this can be found in chapter 7, references [8]-[10].

The memory model of Linux consists of two types: real physical memory and virtual memory. The maximum amount of virtual memory that can be addressed is 4GB on an i386 machine (32 bits system), which is quite often much more than the physical memory in a machine. To be able to still allow a process to address the full 4GB of memory, the virtual memory is split in pages of 4kB each, and a so called page table keeps track of which virtual memory page is mapped onto which physical memory page. Every 4kB page in use will and should have a

matching page table entry (PTE). The page table on its turn contains multiple (page table) pages of each 1024 PTEs. See Figure 6 below.

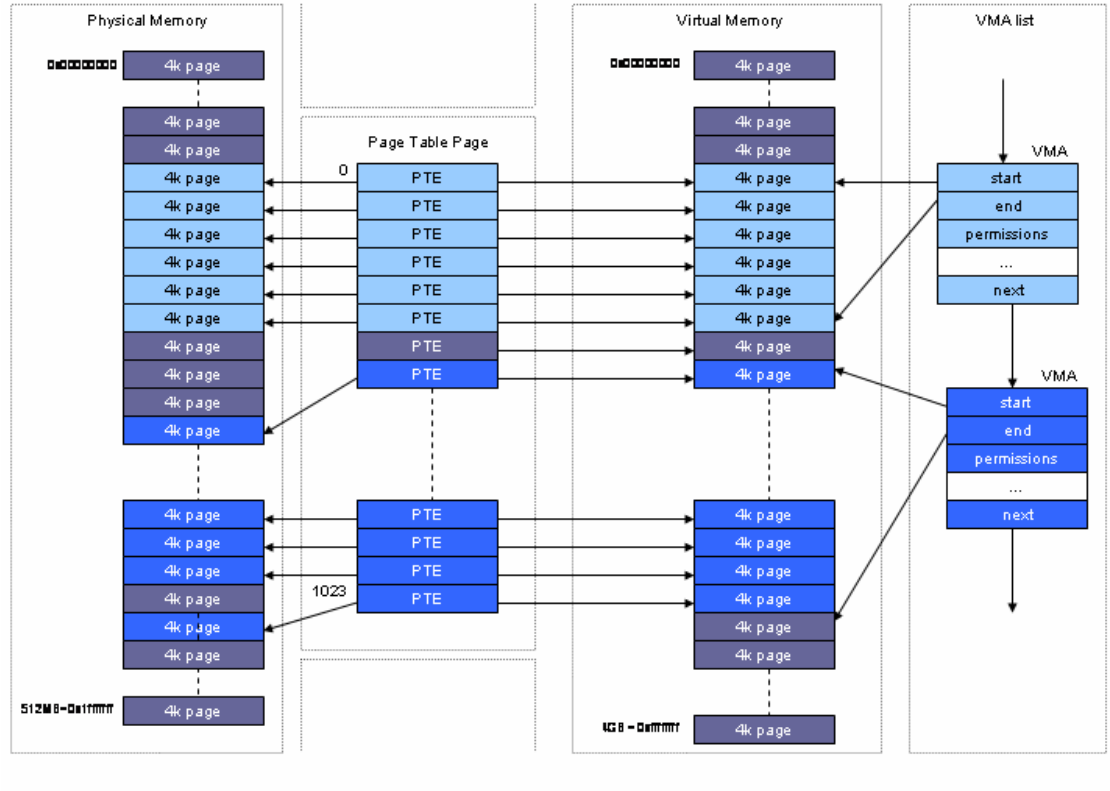

Figure 6 Simplified overview of Virtual Memory Management in Linux

The virtual memory space is again divided into logical sections by grouping pages together, which is done based on the virtual address range and permissions. These memory sections are known as Virtual Memory Areas (VMAs). For every VMA that is stored in memory, the kernel remembers the start and end address and the permissions set for this VMA. Permissions can amongst others be: read-only, writeable, executable, shared and private.

Note. If an executable is started, the code is for example loaded into memory and the VMA used for this is will have executable and read permissions, while the data for this executable will have read and write permissions.

Every process keeps track of these VMAs in a linked list in order to provide proper memory management. Besides that, the number of VMAs in use is remembered as well, the upper limit for this counter is 65535.
Linux has support for resizing of VMAs as well as moving an entire or a part of a VMA from one memory location to another memory location. So a VMA could start at

0xb0000000 - 0xb0000100 could be resized to 0xb0000000 - 0xb0000200

But a move is possible (at the same time) as well:

0xb0000000 - 0xb0000100 could be moved to 0xa0000000 - 0xa0000100

Now if we move just a part of a VMA from the middle (without resizing it) to another virtual memory address, we have to create a new VMA, because a new memory area will be used and the start and end need to be stored. Also the underlying permissions need to be copied from the original VMA to the new VMA.

But, and this is the heart of the vulnerability, what happens if the maximum number of VMAs was reached before we tried to move a part of a VMA to a new location? Splitting it will add one more VMAs to the list, but as the maximum is reached this cannot be done!

One thing more needs to be said about VMAs in Linux. When a process starts, at least the code of the executable is loaded into a memory area (VMA) as mentioned, and another VMA is allocated for the data. All memory mappings of a process can easily be seen in the special file /proc/<process id>/maps, where <process id> is the process id of the process running. For example, running cat gives the following VMA mappings:

Address	perms	offset	dev	inode	path
08048000-0804a000	r-xp	00000000	03:01	948455	/bin/cat
0804a000-0804c000	rw-p	00001000	03:01	948455	/bin/cat
0804c000-0804e000	rwxp	00000000	00:00	0	
40000000-40013000	r-xp	00000000	03:01	163584	/lib/ld-2.2.5.so
40013000-40014000	rw-p	00013000	03:01	163584	/lib/ld-2.2.5.so
40017000-4012a000	r-xp	00000000	03:01	163588	/lib/libc-2.2.5.so
4012a000-40130000	rw-p	00113000	03:01	163588	/lib/libc-2.2.5.so
40130000-40134000	rw-p	00000000	00:00	0	
bffff000-c0000000	rwxp	00000000	00:00	0	

This shows us clearly that the code for cat, the dynamic loader “ld” and the library “libc” are loaded into memory, with read-only/executable parts (code) and readable/writable parts (data).

3.2.6 The vulnerable function: do_mremap

The “do_mremap” kernel function is used to resize and move (parts of) a VMA. It will call the kernel function “do_munmap” to actually:

1. remove any existing memory mapping in the new location
2. remove the current mapping of the memory we want to move

Now things go wrong on the first do_munmap when it involves the middle part of an existing memory mapping and the maximum number of VMAs is reached! See Figure 7 below. Freeing the necessary memory in the existing VMA1 should give us two new VMAs (3 and 4). But as the limit on possible VMAs is reached, the do_munmap function will free the memory mapping, cannot create the two new VMAs so will not split VMA1, but instead it will return with an error code.

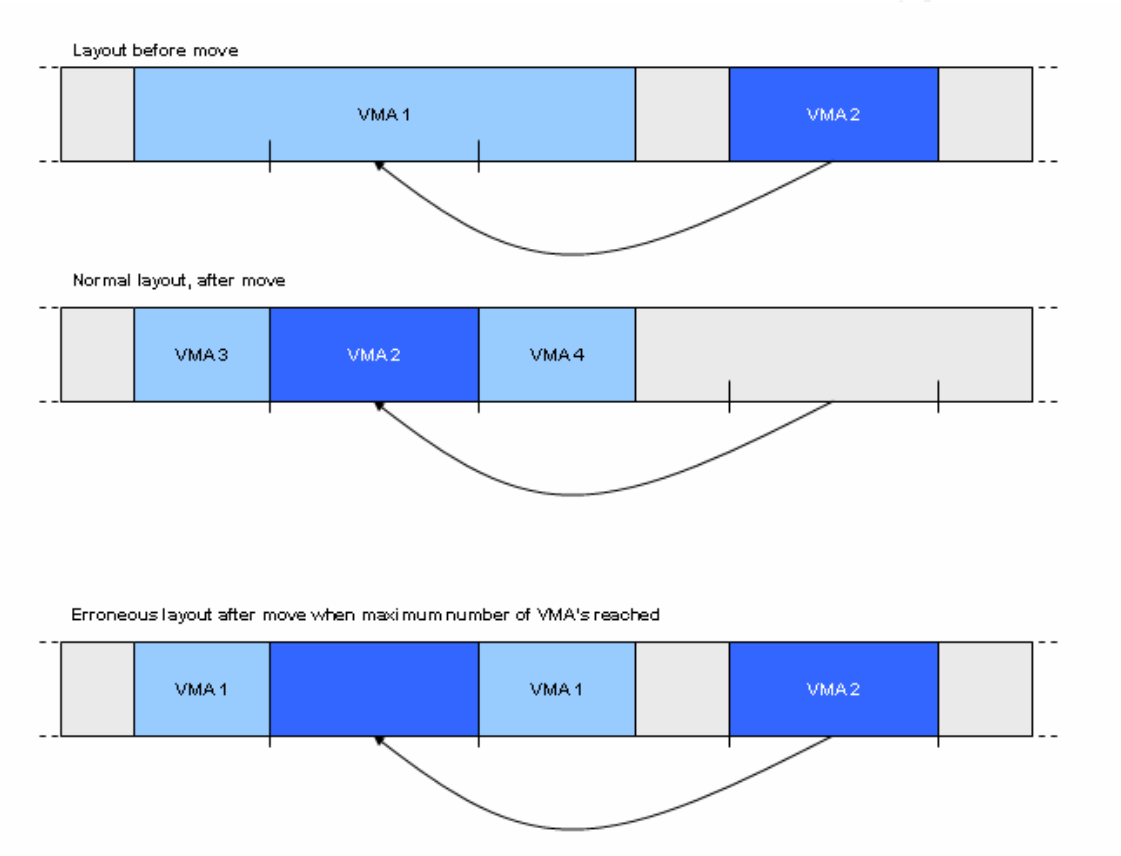

Figure 7 Moving VMAs

The do_mremap code does not check the return code of the do_munmap call and will simply assume that this call always succeeds and that the new memory mapping is available to move data to. But remember: it is still part of VMA1 with all the permissions set for VMA1 in place!

As soon as do_munmap returns, the region in VMA1 is freed and the matching page table entries (PTEs) are removed from the list of know PTEs and became ownerless. But all permissions are still stored. Moving VMA2 to this new location directly will fail because of some more low level VMA code. So unfortunately a direct exploit of this bug is not possible. But we have some more tricks on our sleeves.

3.2.7 The exploit

The main thing to remember here is: we can trick the kernel into creating ownerless memory pages with permissions that we like. We can fill these memory settings with code, for example, to execute a shell. Now if we can only trick the kernel into executing this code with a higher privilege than we should have, we are done!

Recalling that whenever an executable is loaded into memory, page tables are automatically filled and permissions are automatically set, these page table pages are created on the fly in pre 2.4 Linux kernels. So whenever new memory is needed, a new page table page of 1024 PTEs is created, supplying 4MB of virtual memory.

With release 2.4 a page table cache was created. Once a page table page was created, but not needed anymore, it was added to a linked list of page table pages. And as soon as a process needed more memory, the first page table page was taken from this list and used. Until release 2.4.19 the page table cache was available but not enabled, but in release 2.4.19 and following it is enabled by default.

As we saw just before, we can get into a situation where all VMAs are allocated and by trying to move a VMA into the memory mapping of another VMA, create some ownerless PTEs that we know exist, but that the kernel lost track of, but these PTEs are still in the page table of the process and the pages contain information. See figure 8 on the next page.

By filling some VMA with code we like (for example execute a shell), followed by creating enough VMAs to make the counter go nuts (65535) and calling mremap, we will cause PTEs to become ownerless but still existing in the page table page. Next we free enough memory again to make the page table page be freed up and put in the linked list of cached page tables. So now this page table page contains a hole that the kernel is unaware of, pointing to pages in virtual memory contain our code.

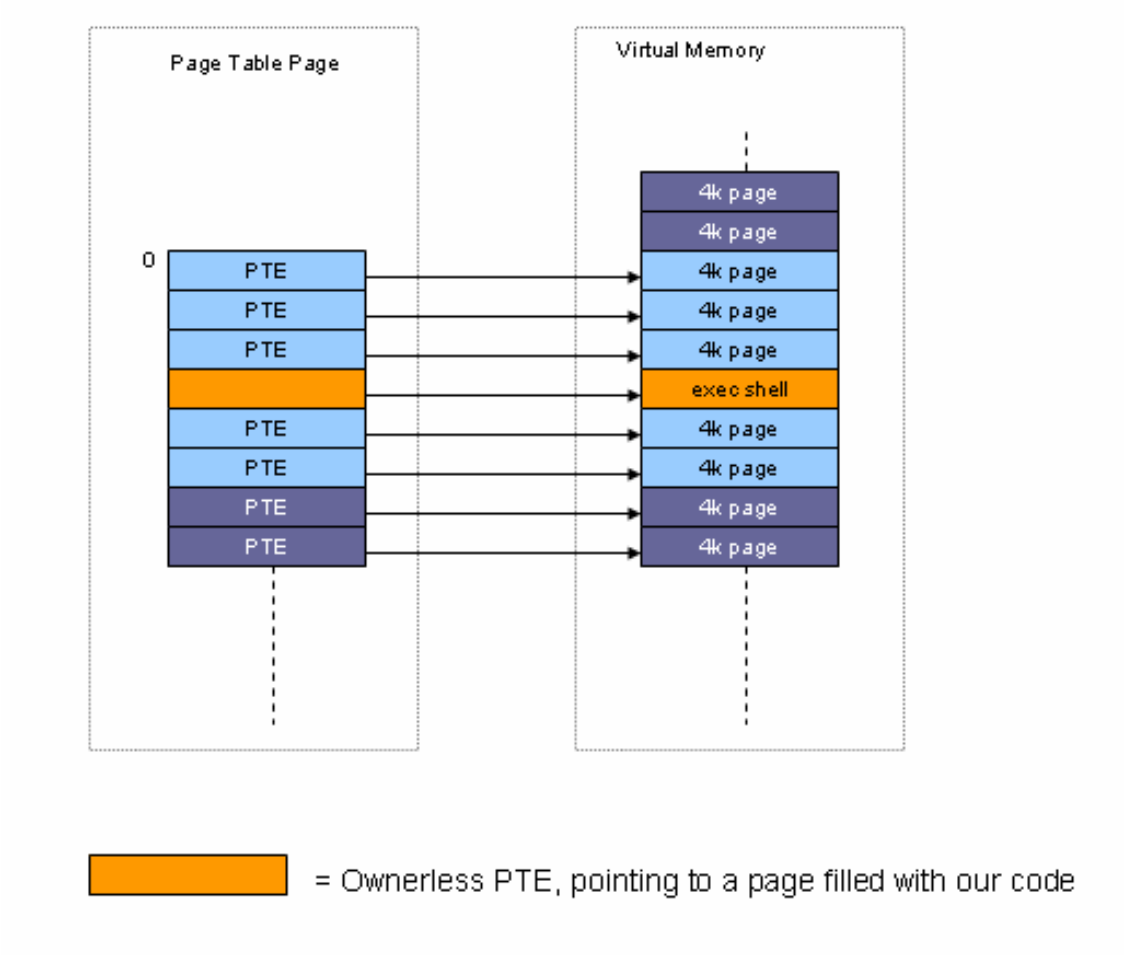

Figure 8. Simplified ownerless PTE situation

We now execute a program that has the suid bit set[3]. The code of this executable will be loaded into memory. Since we just freed enough memory to free up a page table page, our process will re-allocate the last page table page, by retrieving it from the cache. It will now start to fill the pages pointed to by the PTEs with the code from our suid program. But it will skip the ownerless PTEs! Thus we can assure that the pre-filled pages will be inserted into the code base of the suid program in memory, and as soon as the dynamic loader will start to execute this code, our injected code will be run with the suid bit set as well. Remember, our code is part of the same VMA as this suid program and therefore will have same permissions!

Now we are done. By executing a simple program with the suid bit set (in the case of our exploit, /bin/ping) and creating some code to execute a shell (/bin/sh) we now can “simply” inject this code and run our shell as root.

[3] Meaning that any user can execute it, but that it will run with root privileges.

3.2.8 Advanced programming

In order to be able to actually exploit the do_mremap function, we need some programming tricks to get access to this function. In user space, only the mremap function is available, and it cannot be used to move a VMA to a new memory location, only to resize it. It can return a new address for the VMA in question, but this is not powerful enough to exploit the found vulnerability.

Therefore we need to access the kernel function do_mremap directly, which is done by creating a new function via the _sycall5 C-macro. This can be used to map a kernel function to a user space function, and gives us access to the additional parameter, the new address where we want to move our VMA to. The details of how this is implemented, is outside the scope of this paper. More information can be found in ref. [11].

4. Stages of the Attack

4.1 Reconnaissance and scanning

In this case the hacker did part of his reconnaissance by a sort of social engineering. Through discussions on a photography forum he coincidently learned where someone worked and that he did a lot of work from home for his company. Because the forum was a photography forum, many examples of digital photos were linked by the employee to his webserver. So he started looking at the HTML source of a posting of the employee:

```
Employee wrote at 14:35 on Wed Feb 4th:

I took the following pictures during my holiday in Peru! <P>
1. Machu Picchu from below <BR>
<IMG SRC=http://employee. adslprovider.com/peru/PICT3125.jpg><BR>
2. Machu Picchu from a distance <BR>
<IMG SRC=http://employee.adslprovider.com/peru/PICT3140.jpg><BR>
<P>
All comments are welcome…
```

He discovered that the hostname (employee.adslprovider.com) was part of the domain of a well known ADSL provider, so the employee was probably running a webserver at home. By now the hacker got the smart idea to try to see if he could connect to the corporate network, via the home network of this employee. Not to do any real damage, just to prove he can!

The address of the webserver is “employee.adslprovider.com”. Using his Linux machine, a reverse DNS lookup by running the ‘host’ command, gave the following result:

```
/home/hacker> host employee.adslprovider.com
host employee.adslprovider.com has address x.y.19.119
```

From there he could easily find out which webserver and probably operating system the employee was using, by doing a telnet to port 80 and performing a GET request for HTTP version 1.0, showing webserver information as well:

```
/home/hacker> telnet employee.adslprovider.com 80
Trying x.y.19.119
Connected to x.y.19.119.
Escape character is ‘^]’.
GET / HTTP/1.0

HTTP/1.1 200 OK
```

```
Server: Microsoft-IIS/5.0
Date: Sat, 03 Apr 2004 02:03:18 GMT
Connection: Keep-Alive
Content-Length: 1270
Content-Type: text/html
Set-Cookie: ASPSESSIONIDQGGQQKJU=DFADLNADHFPJPELNLDHANHKN; path=/
Cache-control: private


<HTML>
<HEAD>
…..
```

This tells the attacker that the user is running an Internet Information Server (IIS), version 5.0. And because it is IIS, we even know that the operating system must be MS Windows. This scan is not illegal and will not cause possible alarms to trigger on the target host as we are performing a normal HTTP request, just not by a browser, but by using the 'telnet' command.

So by now, the hacker discovered, without performing any illegal activities what the address of the employee is at home, what Operating System he is running and what the version of the webserver is. Now he can start looking for exploits on this system to gain full system access.

I will show later that scanning on the internal network was not necessary either, making the possibility of detection much smaller.

4.2 Exploiting the Employees System

I will not describe in detail how the hacker got access to the system of the employee as this would be a separate paper. This paper instead will focus on how the VPN connection is abused to get to the internal company network, and how a local exploit is used to get full privileges on this company network.

A way for the attacker to gain access to the Windows system would be using the Directory Traversal Bug in IIS 5.0 (Microsoft Security Bulletin MS00-057) to run cmd.exe and upload/execute executables at will. A telnet to the system with a GET request as follows:

```
GET /scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir
```

proves that the webserver is vulnerable. Running this command, immediately shows us the underlying operating system as well: Windows 2000. Using the default installed tftp client, the attacker can now download any executable he likes and run it (ref. [5]).

For example, uploading the 'DebPloit' exploit (ref. [6]) will allow him to run any command with system privileges, in other words, full system control can easily be obtained that way. Downloading multiple files can be simplified by downloading one batch file, with commands to download all other files. Running this batch file will then download the files necessary and perhaps even install them.

For now, I will suffice by saying that the attacker was able to gain full access on the system.

Installing a backdoor that allowed for full system control was easy as no virus scanner was running on the system. The installed backdoor is "Remote Encrypted Callback UNIX Backdoor" (RECUB) (ref. [12]).

A keylogger[4] was installed on the system to show all userids and passwords the employee enters. The keylogger installed was "Windows Keylogger 5.0", freely available at http://www.littlesister.de/ . It will show the name of the application together with all keystrokes and is able to mail this invisibly to a given address.

This keylogger quickly showed what userid/password combination was used to connect from this system via the VPN to the corporate network. But it also showed the userid and password used by the employee to connect via SSH to the management server on the internal network! They appeared to be the same.

```
12:58
PuTTY Configuration (C:\Temp\Putty\putty.exe)
10.10.0.120
employee
pwd1234
cd /home/employee/work
vi application.c
```

Note: no scanning was necessary on the internal network that could have triggered alarms, to find a host to connect to as the log also revealed the IP address of the management machine. A DNS lookup from the employee's machine showed:

```
C:\Inetpub\scripts>nslookup 10.10.0.120
Server:  dns01.company.com
Address:  10.10.0.121

Non-authoritative answer:
Name:    managehost.company.com
Address:  10.10.0.120
```

[4] Small program that will capture all keystrokes entered via the keyboard and store them in a (hidden) file on the disk. This allows an attacker to easily discover passwords entered by the user of the system.

To be able to connect to the systems on the internal company network the attacker installed a command line SSH client for windows. In this case: ftp://ftp.funet.fi/pub/unix/security/login/ssh/contrib/**ssh-1.2.14-win32bin.zip**.

All the tools used were put in a hidden directory called “MS\Windows\Tools”, created in an existing hidden subdirectory:

C:\Documents and Settings\All Users\Application Data

Easily accessible via the traversal bug as path:

C:\DOCUME~1\ALLUSE~1\APPLIC~1\MS\WINDOWS\TOOLS\<cmd>

Once the Windows 2000 machine was compromised and Administrator rights were obtained, the attacker discovered that the VPN client used was a ‘Nortel Contivity VPN client’, with split tunneling enabled (according to the configuration file).

> Split tunneling in a VPN client allows simultaneous access to both the network behind the VPN tunnel and the Internet. In most cases the VPN server will push to the client which networks should be routed towards the VPN tunnel. All other networks will stay routed towards the Internet. This allows the user to access the corporate network, while at the same time, accessing local printers and the Internet. If no split tunneling is used, no local printers will be accessible and if a user wants to go to the Internet, all traffic will go via the VPN tunnel to the company network and via their Internet link to the Internet, causing extra load on the Internet link and extra delays for the user. That’s why many companies allow split tunneling, not being aware of the risks this involves.

The attacker will now have to wait for employee to connect to the company network, and because split tunneling is enabled, he will still be able to use his installed backdoor. As soon as a VPN connection to the internal network is made, the attacker will start his command-line SSH client to connect to the internal network of the company, just as the employee would do with a terminal type SSH client, but his session will not be directly visible to the user.

4.3 Exploiting the internal system

As soon as the attacker notices a VPN connection from the employees host, into the corporate network, he will start (as mentioned) the SSH client to connect to the internal management system. This is the only system from which connections can be made to all other critical systems on the network.

After a connection is setup using the account info of the employee gathered with the keylogger, the attacker will try to detect the operating system version via the ‘uname’ command. Running this command gives us the result:

```
/home/employee> uname -a
Linux ManageHost 2.4.23 #1 Sat Jan 3 10:38:08 EST 2003 i686 unknown
```

The “netstat” command can tell us which services are running on this host, so the attacker does not need to do a scan, again limiting detection. The command gives a (only a few ports listed):

```
/home/employee> netstat -a
Active Internet connections (servers and established)
Proto  Recv-Q Send-Q  Local Address      Foreign Address    State
Tcp    0      0       *:ident            *:*                LISTEN
tcp    0      0       *:ssh              *:*                LISTEN
tcp    0      0       *:smtp             *:*                LISTEN
tcp    0      0       *:time             *:*                LISTEN
```

In other words, the host is running (amongst other) an SMTP (mail) server and of course SSH server. But there is no need to try to find a remote exploit for any of these services, as we are already on the system and we know that this OS version is vulnerable for a local privilege escalation attack, based on do_mremap.

So via the host of the employee the attacker downloads a pre-compiled version of the exploit ‘mremap_pte.c” and then via the command-line version of scp (SSH copy), transfers it to the internal management system. Executing the exploit results in a shell with root privileges as shown below.

```
/home/employee> id
uid=1000(employee) gid=1000(employee) groups=1000(employee)
/home/employee> ./mremap_pte /bin/ping /bin/sh
a=2
b=4
c=23


[+] kernel 2.4.23  vulnerable: YES  exploitable YES


    MMAP $65530  0x50bfa000 – 0x50bfb000
[+] Success

sh-2.05a# id
uid=0(root) gid=0(root) groups=1000(employee)
sh-2.05a#
```

As soon as the attacker obtained root access on the management machine, the entire network is open to him. A good option for him to continue is to download the files “/etc/passwd” and “/etc/shadow” so he can run password crackers on his home system to find more accounts to continue his way through the network.

By looking at the contents of the "/etc/hosts" file it is fairly easy to find out what other systems are reachable from this network, most often with the same userid and password.

127.0.0.1	localhost ManageHost
10.0.100.3	Backup1
10.0.100.4	Backup2
10.0.100.51	Database1
10.0.100.52	Database2

The mission of the attacker is complete when he is able to use the account of the employee to connect via the management system to one of the database systems and can copy a backup of the database, containing customer information and credit card numbers! In order to do so, he just had to run the same exploit again on this system as all internal systems are running the same OS and kernel level.

4.5 Keeping Access

In order to keep access to the internal network, the hacker has multiple ideas:

1. Create a backdoor from within the network that allows him to connect directly to the internal network
2. Keep using the connection via the home PC from the employee.
3. Add a new account to the database with authorized VPN users and set up a VPN tunnel from a system on the Internet.

Option 1 is almost impossible as the only way to do so, would be exploiting the webserver on the DMZ. But only HTTP and HTTPS (SSL) traffic from the Internet to this host is allowed on the firewall, making a backdoor virtually impossible. An option would be to start a netcat session from the compromised host to a compromised system on the Internet on port 80. This could be started at regular times using "cron", allowing the attacker to connect via this "tunnel" back to the host on the internal network. Unfortunately (for the hacker that is), the company only allows the local proxy server to connect via the firewall to the Internet, closing the door for a backdoor like this.

Option 2 is not really an option to keep access, as the hacker is forced to connect to the network, when the employee is online. As soon as the employee disconnects his VPN tunnel, the hacker will be disconnected from the internal network as well. As long as this is just to proof (as originally intended) that he can access the internal network, this is fine. But of course, seeing what he can do, he wants more. So only one option is left to the hacker….

Option 3 would normally quickly be discarded as running the VPN client from a remote system is almost impossible, therefore the attacker should use his home

system. The VPN log files would then easily reveal the IP address of the attacker at home. But because he wants to transfer the customer information and creditcard numbers to make some money out of this, he needs a connection that is available for a longer time. Also, transferring all data via the machine of the employee, might make the employee suspicious, because it will significantly slow down his connection. So even though he used multiple machines on the Internet before exploiting the employee's machine, greed makes him to use his own IP address and machine to download the database. Without this mistake it will be virtually impossible to ever trace him.

As soon as the hacker gathered the necessary files, he will establish a VPN tunnel from his system at home, download the data and disconnects again. Making the probability of being noticed as small as possible, he decided to create this connection during office hours, when multiple users are connected. A VPN connection at night might look very suspicious if someone would be watching a log file.

4.6 Covering Tracks

The only suspicious tracks that the attacker left behind, will be the files he installed on or downloaded to the host of the employee and the management host.

The files on the employees host are put in a hidden directory on the system and unless a user explicitly enables "View hidden files and folders", these hidden folders will not be shown in Explorer. Putting them in an obscure directory that most users don't even know of makes them even harder to find.

On the Linux machine on the internal network, the only visible traces of the attacker would be the login times of the account of the employee (!) which normally should not be very suspicious. And of course the exploit used to escalate privileges to root. But this file does not need to stay on this machine. The attacker can simply copy the file from the employees host to the internal network, using scp. The dynamically linked executable is approx. 8 kilobytes, the statically linked executable approx. 430 kilobytes.

Running the exploit will not leave any traces in the system log files. Only the history file of the employee will show that the exploit is run. Renaming the exploit to '.test2.sh' and running the exploit from a simple batch file that supplies the correct parameters, will suffice to hide visible clues from the shell history file:

```
vi .test1.sh
chmod +x .test1.sh
./.test1.sh
rm .test1.sh  .test2.sh
```

The vi command will then be used to edit the file ".test1.sh" and give it the following contents:

```
#!/bin/sh

./.test2.sh /bin/ping /bin/sh
```

Note that even though the exploit has a ".sh" extension, this is just to confuse others; it is just a compiled executable. Extensions in Linux have no meaning.

By starting the filenames with a "." They will be hidden for the employee when he will do a quick directory listing. Looking at the history file of the shell, it is easy to miss the "." in front of the filename, making the names look very plausible.

If the attacker would like to hide its trail even more, he could simply make the history file of the shell read-only. This will make sure that none of the commands will be remembered. The disadvantage of this is the fact that the employee will also not be able to store his commands which might be suspicious to him. So he decides to leave the history file intact and try to obscure his attack by using innocent looking filenames.

5. The Incident Handling Process

5.1 Preparation

As the company was working with external customers and creditcard information, they were, as a company, very security aware. So all systems with a connection to the Internet were hardened properly, meaning only necessary services were running. Only http traffic from the Internet was allowed to a host on the DMZ and only database traffic was allowed from the DMZ host to the intranet. All telnet connections were replaced by SSH connections, as this encrypts traffic, making it harder for hackers to sniff password if they would ever get on the internal network.

But one security hole still existed in the network, the VPN service. This service was setup fairly quickly to reduce the costs of employees working from home, and to reduce workplaces in the office. No security policy was created, involving desktops connecting via the Internet using VPNs to the corporate network. So the security of such a system was left to the employee owning the system. If such an employee, like the one in our case, is not enough security aware/minded, this creates a backdoor to the corporate network, bypassing all security measures with firewalls etc.

As mentioned before, the administrators made sure that patches were applied as soon as remote vulnerabilities were discovered, and also tried to apply patches when local vulnerabilities were discovered. But in the last case, these often involve a reboot as kernel code is updated, causing valuable downtime. In the end, their network is very well protected; no hacker would be able to get access to a system, allowing him to run an exploit for a local vulnerability in the first place!

No incident handling team existed at the time our attack took place and no incident handling process either. Again, mainly because of costs. Creating procedures for such events is a time consuming process, in other words expensive, and does not show any direct cost benefits.

5.2 Identification

The incident was identified by coincidence, as is often the case. The person responsible for the Internet connection, shared between the VPN server and the firewall, installed mrtg (ref. [13]) to monitor the bandwidth usage. The mrtg-application is a simple package that creates nice graphs of anything you want to measure. It creates these graphs by collecting information on a regular basis

(often 5 minute intervals) and then creating daily, weekly, monthly and yearly graphs of this.

In this case, mrtg collected the amount of traffic that went through the interface of the router for the Internet connection, using the built in SNMP engine. Both incoming and outgoing traffic was monitored and displayed in one single graph. This monitoring was used to detect when the Internet connection needed to be upgraded, to prevent performance problems for customers connecting to the webserver on the DMZ. Because a bad performance makes customers leave and go to other parties, with a better performance.

On a Wednesday morning, Feb 1st 2004 10:30 local time, the operator looked at the daily graph (Figure 9) and noticed that the traffic towards the Internet was unusually high for the time of day.

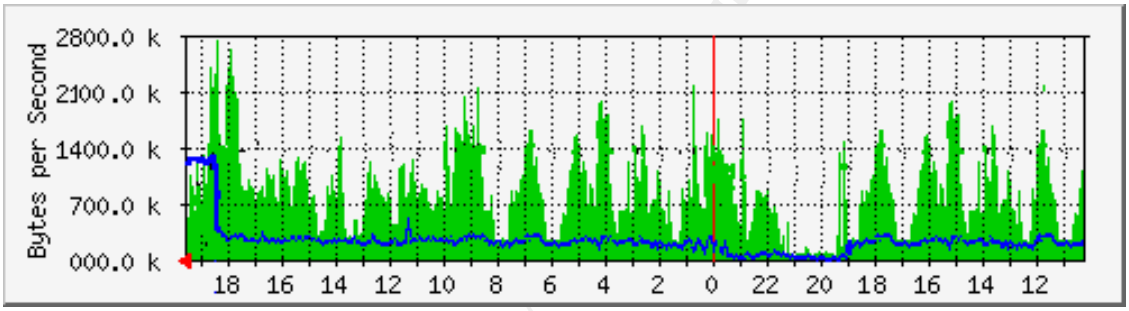

Figure 9. mrtg graph

Times are GMT, local time is GMT -9. The green graph presents incoming traffic, the blue line presents outgoing traffic. Peeks in the green line are not abnormal, traffic is not constant, but fluctuates. Incoming traffic is almost always more then outgoing traffic. But not this morning. Starting at 18:15 GMT, 9:15 local time, outgoing traffic is about four times the normal amount of outgoing traffic. As the operator never saw such a high bandwidth consumption before, he is wondering where this is coming from. The line is shared between the firewall and the VPN server, but this is too much traffic for too long to be coming from a webserver. Either a user on the internal network is sending a lot of data via HTTP to the Internet, or a VPN user is downloading a lot of data to his home. Being curious the operator connects to the VPN server and sees 5 active VPN connections:

Userid	Start Time	Bytes Out	Bytes In	External IP	Internal IP
emp03	19:01:03	152432	42123	212.1.2.3	10.10.110.5
emp03	18:15:42	43163451	142123	88.4.5.6	10.10.110.4
emp08	18:01:12	92432	12356	213.13.2.3	10.10.110.2
emp11	17:24:43	253542	35132	209.11.2.3	10.10.110.1
emp16	18:01:21	163451	35614	210.12.2.3	10.10.110.3

The operator quickly discovered that a user with userid “emp03” is sending a lot of data to his home PC. But at the same time he notices that this employee has

two VPN tunnels at the same time, from two different IP addresses on the Internet!

A mail to the employee at 10:35 confirms that he is working from home at this very moment, using IP address 212.1.2.3 and is not aware of a second tunnel to the corporate network with his UserID. This combined with the fact that a lot of traffic is send to the Internet via this tunnel made the operator nervous.

At 10:40 he decided to login to the router between the internal network and the VPN tunnel to see where all the traffic is originating from. From the VPN server overview he knows that the IP address assigned to the second tunnel is 10.10.110.4. Checking the traffic in the Cisco router reveals that the traffic is originating from the Database server!

As this should never happen he now knows for sure that they are dealing with an attack and decides to immediately shutdown the VPN server to stop the attacker. Time that the server was shutdown: 10:42. The time between the original detection and the time the hacker got disconnected was 12 minutes. The hacker was downloading from the network for a total of 1 hour and 27 minutes, a total of around 60 MB.

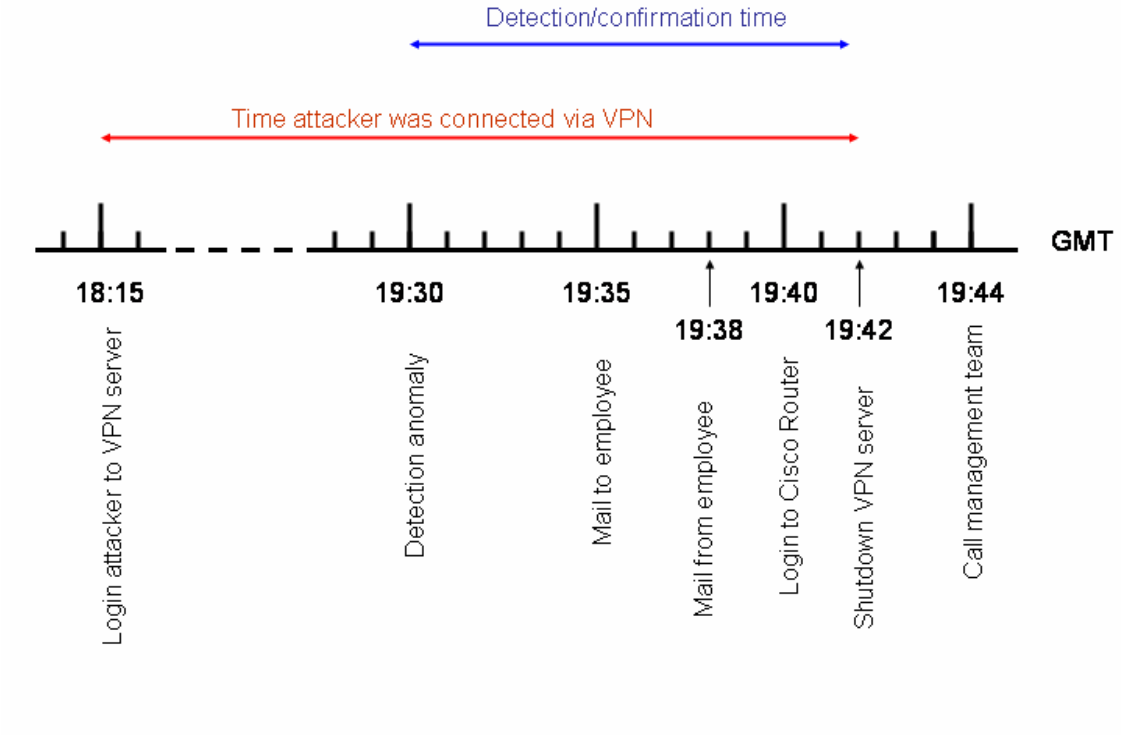

Figure 10 Timeline of identification

Times were not written down by the operator, but were later retrieved from the log files of the VPN server, Cisco router and from the timestamps of the emails send.

At 10:44 the call is made to inform management that an attacker might have had access to the database and all customer information.

5.3 Containment/Eradication/Recovery

No (certified) incident handling team exists within the company, so management decides to involve server administrators and a few security aware employees to react as a team to this incident. Management requires though, that business should be up and running again as soon as possible, but in a safe way.

First of all, it is decided that the attacker entered via the VPN server and as this path is closed, he is blocked from the network. To make sure no other paths exist, the firewall logs and the servers on the DMZ should be checked for anomalies.

The security personnel convinces the management team from shutting down the database server, as it is unknown how the attacker gained access at this time and what he left behind on the machine. Even though this will stop customers from accessing the database, in other words stop the money from coming in, management agrees to prevent further loss of confidential information. It is decided to shutdown the machine, and make an exact copy of the harddisk in the machine. The original drive will be left untouched after that. The copy will be used for analysis and a new drive will be put into the machine with a fresh install of the OS to get business up and running again asap.

The only way to gain access to the database machine is via the management host, it is decided to follow the same procedure for this machine as well. Shutdown immediately, full one-on-one copy of the disk to a new disk for analysis and reinstall the system from scratch on a new disk.

Note: as the Operating System is Linux running on Intel machines with IDE disks, the costs for new disks are acceptable in this scenario compared to the costs off being out of business.

To get business up and running again as soon as possible, it is decided to first install a new drive in the database server, and reinstall the OS and database application (of course with new passwords for all accounts). Then the copied disk is mounted to retrieve a copy of the latest running database. This is not a security risk as no authorization and/or authentication information is stored in the

database, only customer information, orders etc. Unmounting and removing the copied disk and starting the machine brings the company back in business.

Reinstallation of the management host is done after the database is back online. All accounts on this host will receive a new password, which will be communicated to the employees directly or via the phone in case of home workers.

As soon as all machines are up and running again, the VPN server will be started to allow all employees access again, except for the employee whose account got hacked. Until further investigations show how the attacker got access to his account credentials he will be not be allowed to reconnect again.

After 4 hours, the database system was back online and after 8 hours the management host as well, be it in a minimal setup.

5.4 Investigation

Investigation was done in a controlled environment, an isolated LAN with two machines running from the copied disks of the management host and the database system.

To find out how the attacker gained access to the corporate environment, the VPN log files were retrieved from the management host and a small perl script was created to check if a) the IP address of the attackers was seen before and b) to see if it happened before that any userid was connected multiple times at the some moment. Both came out negative.

As the only system accessible to the employee was the management host, it was checked (using the Linux command ‘last’) when he logged in to this machine and how often. It appeared that in the last few days the employee often had multiple SSH connections open at the same time via his VPN connection. Asking the employee if this was usual behavior, he denied. According to him he started an SSH session every day during the start of his workday to connect via the management host to the accounting host.

Looking at the home directory of the employee, the following directory listing was retrieved:

```
root@temphost:~$ ls –al /home/emp03
drwxr-xr-x    6 emp03    employee     4096 Feb  1 18:18 .
drwxrwsr-x    3 root     staff        4096 Dec 29 22:12 ..
-rw-------    1 emp03    employee     7849 Feb  1 18:18 .bash_history
-rw-r--r--    1 emp03    employee      509 Dec 29 12:12 .bash_profile
-rw-r--r--    1 emp03    employee     1093 Dec 29 12:12 .bashrc
drwx------    2 emp03    employee     4096 Jan  8 13:46 .ssh
-rwx------    1 emp03    employee     8872 Feb  1 18:16 .test.sh
-rwx------    1 emp03    employee       40 Feb  1 18:17 .test2.sh
```

Showing only two executable files last modified the morning of the attack. The fact that these are 'hidden' files makes them already suspicious. They appear to be shell scripts for testing purposes but a closer look reveals that ".test.sh" actually is a binary file as the command "file .test.sh" shows:

```
root@temphost:~$ file test.sh
.test.sh: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked
(uses shared libs), not stripped
```

The shell script contains:

```
root@temphost:~$ cat test2.sh
#!/bin/sh
./.test.sh /bin/ping /bin/sh
```

Before starting this script, the "strings" command is run on this executable to see if we can find what it will try to do:

```
root@temphost:~$ strings .test.sh
/lib/ld-linux.so.2
libc.so.6
waitpid
printf
stdout
munmap
memcpy
fflush
mmap
uname
__deregister_frame_info
sscanf
memset
__errno_location
_IO_stdin_used
_exit
__libc_start_main
strlen
mprotect
__register_frame_info
__gmon_start__
GLIBC_2.0
PTRh
QVhd
/bin/ping
/bin/sh
   MMAP #%d  0x%.8x - 0x%.8lx
```

```
[+] Success
[-] Failed
%d.%d.%d
[-] invalid kernel version string
a=%d
b=%d
c=%d
[+] kernel %s  vulnerable: %s  exploitable %s
```

This does not tell us exactly what the executable does, but the last line tells us that this is most likely a kernel exploit. Looking at the other strings in the executable gives us the impression that some sort of memory exploit is used. Using a search engine on the Internet (google) quickly reveals that this is probably a kernel exploit for the “do_mremap” vulnerability.

The kernel version of the Linux system was 2.4.23 which is vulnerable to this attack. Running the exploit via the test2.sh shell script indeed gives a shell with root privileges, confirming the investigation.

Now it is becoming clear how the attacker gained access to other machines on the network even though our employee ‘emp03’ did not have access to this machine. He most likely got a copy of the password file and cracked this, gaining other userid/password combinations that allowed him to connect to the database system. As only 2 administrators had access to the database system, he must have retrieved at least one of these passwords.

A quick check on the copy of the database systems shows two suspicious files in the directory of admin2: “.test.sh” and “.test2.sh”, one binary and one shell script, both the same size as the ones found in the directory of the employee. As this machine also runs kernel version 2.4.23 the same exploit was used to gain root privileges and access to the database.

Now it is clear how the attacker found his way through the machines on the internal network (local exploits and password cracking), but how did he get the userid/password of the employee in the first place.

After a phone call with the employee at home, the team discovered that the employee never wrote down his password (let alone, told anyone about it) and also did not have a very easy to guess password. We still expect that the employee is not the attacker (innocent till proven guilty) because the IP address of the attacker is from a subnet of a different ADSL provider, in a different country as well.

So this leaves only one possibility, the host of the employee at home is compromised. Another phone call to the employee is made in which he is requested to bring his home PC to the office for further analysis. As this is not

company property, they do not have the right to check this PC, but the employee grants them access, to see if the host is compromised and what happened. He will (for now) not allow a copy of the system to be taken as it contains many private files; remember that he was using peer-to-peer file sharing? In case of a prosecution this will be a big obstacle!

As soon as the PC arrives, the security team notices that the machine is not fully patched (windows update is never used) and that no virus scanner is running. Installing a virus scanner (with permission from the employee) almost immediately shows that the host is compromised and contains a keylogger, backdoor utility as well as other tools hidden in a 'hidden' subdirectory. Realizing that the VPN client allows split tunneling, they now know how the attacker penetrated the corporate network.

They are only wondering how the attacker compromised the machine in first instance, since the ADSL modem used (Speedtouch) contains a firewall that will block incoming connections to all home PC's. Asking the employee about the configuration of the modem, they learn that he forwarded port 80 on the modem to this PC, as he is running a webserver (IIS) to host his photos for a photography forum. Knowing that an unpatched IIS server contains many vulnerabilities that are easily exploitable from a remote location, the first compromise must be done this way.

Fortunately the company decided to store all sensitive customer information in the database in an encrypted way, using 3DES encryption. So not the database was encrypted, but only fields containing sensitive information like company info, orders and credit card numbers. Only 6% of the database was copied which makes it virtually impossible able to read the database as no valid indexes can be found. Because of the encryption, it is very hard to get any sensible data out of the partial database, so the damage seems to be limited because of the early detection.

One last step was trying to find the identity of the attacker, using the IP address from which he connected to the VPN server using the account of the employee. A DNS lookup of the IP address learned that the attacker is located in Russia. Even though they should be able to find him, taking steps against him will be an almost impossible task. So instead of going public with this, management of the company decides not to prosecute the attacker, also because no real damage seems to have been done and using the PC of the employee as evidence will be very hard.

5.5 Lessons learned

After a first investigation performed by the team in the days directly after the attack, they discovered how the attacker first gained access to the machine of

one of their employees, how he next used the VPN infrastructure to gain access to the corporate network and how he escalated his privileges to become root and furthered penetrated the network, ending in access to the database systems and retrieving a (partial) copy of the customer database.

During the setup of the VPN infrastructure, cost reduction was the main driver. Security looked at the setup and decided that the setup was secure enough:

- The VPN server itself only accepted VPN connections and dropped all other traffic.
- The home employees were using ADSL with a modem/router using a built-in 'firewall'.

No other requirements were made to be able to use the VPN service. Partially because it is very hard to demand that end-users comply with their desktops to a standard company policy, if the desktops are not property of the company but of the employee. And partially because they were not aware of the risks of split tunneling and local vulnerabilities. Why protect the machines from local vulnerabilities if the network itself is very well protected against intruders. If an intruder cannot get access to a system, why bother patching systems (with all risks involved) against local vulnerabilities.

This attack taught the company a couple of things the hard way:

- If the company network is extended to include home offices, the corporate security policy should be extended to include those as well. This means, always keep workstations up to date with respect to patches, always run a virus scanner, do not run any unnecessary services, and preferably run a firewall on the desktop.

- Having split tunneling enabled allowed the attacker to stay connected with the backdoor on the computer of the employee, while using that same backdoor to connect to the internal company network, using the employee's machine as a stepping stone. Disabling split tunneling would force the employee to browse the Internet via the company's Internet connection, but would never allow using the remote desktop as a stepping stone.

- Even though encrypted SSH sessions were used to connect to machines, this does not automatically mean that passwords cannot be found by attackers. On its turn this implies that it is still good practice to enforce strong passwords and regular password changes. But in this case the attacker would still be able to grab the new password as well, as the keylogger was still in place. So instead of using a password, a hardware token generating a one-time password would be an even better solution.

- Do not only base your network security on perimeter security. In this case the perimeter can be the boundary between the Internet and the company network, but also the boundary between network and host. The fact that only employees should have access to a host does not guarantee that an attacker will not get access to a host! The host itself should be kept up to date as well. And how about employees trying to harm the company…?

- Juridical prosecution of the attacker was planned in first instance, so exact one-on-one copies of the compromised machines were made. But when they discovered that the employees personal PC was abused and that he was not willing to give them a copy of the harddisk, they discovered that prosecution would be very hard. Add the fact that the person was living in a different country, made them realize that prosecution is very hard and might do more damage to the company's reputation than the attack itself did.

5.6 Advices to improve security

Some advices from the author's side on how the lessons learned could be used to improve security, following the lessons from the previous paragraphs.

5.6.1 Remote Policy Enforcement

If a company cannot enforce the employees to comply with the corporate security policies because the remote desktop is property of the employee, a desktop should be supplied to the employees (if they do not have one already). It will stay property of the company and the employee should be asked to sign a contract that they will comply to the corporate security policies if they use this desktop. Otherwise the employee will not be allowed to connect to the office from home. This means extra costs, but it will increase security significantly and will allow for prosecution in case of another attack.

If possible, the company should try to enforce the policy as well. This means, as soon as a VPN connection is setup, a scan of the connecting workstation will be done to see if it complies with the company policies towards virus scanners (running and up to date), personal firewalls (running and proper ruleset) and patch level of the desktop. Different products from different suppliers exist that can enforce this.

It can be done using the Zonelabs Integrity suite (ref. [14]), consisting of a small client on the desktop and a server on the corporate network, communicating with the VPN server. This product is suitable to medium sized companies. Smaller companies that would like to implement a cheaper product could use the built-in features from the Nortel VPN server: TunnelGuard (ref. [15]). Or if the employees do not receive the administrator password for their machines client

only versions exist as well, that can differentiate between being connected to the corporate network and not being connected, by enforcing different rules for each situation. Other vendors like InfoExpress's Cybergatekeeper (ref. [16]) and Network Associates (ref. [17]) offer similar products as well. A similar solution is offered by Microsoft with Active Directory. If all desktops are registered within Active Directory and users are forced to logon to the AD domain, policies can be set on the machine and updates can be installed automatically. Even when the desktop then disconnects from the domain, the policies will stay active.

The minimal list of items to check should be:

- Is windows update enabled and are the latest patches installed
- Is the virus scanner up and running and up to date (minimum version of .dat files)
- Is a firewall installed and preferably is the required rules active

Enabling remote policy enforcement could in this case have prevented the user from installing the IIS server and would have never allowed the attacker to gain access to the machine of the employee. Let alone install all the malicious software.

5.6.2 Limit Split Tunneling

Enabling split tunneling makes life a lot easier for employees working from home. But if the remote desktop is not protected allows potential attackers to use the desktop as a stepping stone to the internal company network. The disadvantage of disabling split tunneling is dual:

- All traffic to the Internet has to go via the tunnel to the internal network and back out again via the companies Internet connection.
- The local network is not reachable anymore, so employees are not able to reach the local network printers anymore, unless they disconnect from the network.

Therefore the advice is to enable split-tunneling, but only allow RFC1918 (ref. [18]) networks to stay outside the tunnel. All other traffic should be routed towards the internal network. This allows employees to setup their own LAN at home, with for example a 192.168.0.0/24 subnet and reach the network printer whilst staying connected to the internal network.

Note. It still includes a small risk though: if an attacker gains control over one of the machines on the employees home network, he is in the trusted zone of this machine and could still connect to the company desktop. Therefore it is important that this desktop is save, running a firewall to block all incoming traffic

(except VPN traffic of course) while still allowing access from the desktop to the local network.

5.6.3 Strong and/or Central Authentication

As we saw, an attacker can easily grab userids and passwords using a keylogger on a desktop. Another way to get valid account information is installing a network sniffer and looking into the IP packets (if not encrypted). No matter how unguessable the password is or how often it is changed, with a keylogger installed, the attacker will always be able to retrieve it. If hardware tokens are used to generate a one-time password, the attacker can read the password but will never be able to use it as it is only valid for exactly one authentication attempt.

For example RSA Security's SecurID tokens (ref. [19]) generate a one-time password that is even valid for 60 seconds. After 60 seconds the password cannot be used anymore, but using the same password again will also block the second attempt. Race attacks[5] will also fail as the RSA server will block both authentication attempts if they happen within 2 seconds. This solution has only one disadvantage, implementing it for a small amount of users is very expensive. Other solutions like Vasco tokens, using a similar strong authentication method, have the same cost disadvantage.

A cheaper way to increase security, at least at host level, is to use a central authentication server, where all passwords are stored, instead of storing passwords on every system. For example, VPN servers can use the RADIUS protocol to authenticate against a central RADIUS server. Linux supports Pluggable Authentication Modules (PAM, ref. [20]), one of which allows authentication via the RADIUS protocol. By using a central RADIUS server where only administrators have access to, all passwords are stored in one location on a machine with very limited access. Trying to download and crack a password file like happened to this company will not be possible then. Free open source RADIUS servers exist and could very well be used for this purpose. Examples are FreeRADIUS (ref. [21]) and Gnu RADIUS (ref. [22]).

5.6.4 Maintenance windows

Applying patches to the kernel are often not performed for a number of reasons:

- If you prevent attackers from access to the machine, local exploits cannot be used.

[5] An attacker knows the entire password and tries to connect with it, just before the user will try to connect with it. Sometimes this is even done by guessing the last digit/character, and sending multiple authentication requests at once.

- Patches to the kernel will only become active after the machine is rebooted; causing downtime and downtimes are expensive.
- Installing kernel patches might make a stable kernel unstable, which is unwanted on a production system. So acceptance systems are needed, increasing time and costs.

But this attack clearly shows that it is sometimes possible for attackers to gain access to a system in a way that no security officer thought off. In case of this company the acceptance environment probably already existed to be able to do upgrades to applications etc. If not, the costs for a new machine are minimal and the machine can then be used for other acceptance tests as well.

It is also advisable to create a service window during which services might be unavailable for a short period, to allow the installation of security updates and/or software upgrades. By taking a good look at the log files of the applications used by customers, it is often clear at what times the minimal amount of customers is connected, making that a good time to do upgrades. Ensure that these times are published and that customers are notified of such a downtime, by sending out e-mail notifications and/or showing a clear message on the webserver stating that the services are down for maintenance and how long it will take when they get back online.

Make sure that procedures exist for regular maintenance windows and also procedures exist for emergency updates when remote vulnerabilities are published and patches are available, will minimize the downtime.

7. References

This chapter contains all references to documentation and products that can are referred to in this paper. If you want to learn more about the vulnerability and exploit used, read the papers mentioned in [8] – [10], focusing on the last one, the paper written by the discoverer of this vulnerability.

[1] Nortel Contivity Products
http://www.nortelnetworks.com/products/01/contivity/

[2] Speedtouch 610 Multiport Datasheet
http://www.speedtouch.com/ST610%5Cdatasheet.pdf

[3] Manual page of 'iptables'
http://www.cl.cam.ac.uk/cgi-bin/manpage?8+iptables

[4] IIS 5.0 Directory Traversal Vulnerability
http://www.microsoft.com/technet/security/bulletin/MS00-057.asp

[5] Exploiting the IIS 5.0 Directory Traversal Vulnerability using TFTP
http://www.ists.dartmouth.edu/IRIA/knowledge_base/exploit_analysis/hacking.ppt

[6] 'Deploit' Windows 2000 local exploit
http://www.securiteam.com/windowsntfocus/5EP0Q0K6UI.html

[7] BO2k Remote Administration Tool
http://www.bo2k.com/

[8] iSec Security Research: Linux Kernel do_brk Vulnerability
http://www.isec.pl/papers/linux_kernel_do_brk.pdf

[9] http://kos.enix.org/pub/linux-vmm.html

[10] Linux kernel do_mremap() local privilege escalation vulnerability
Paul Starzetz (ihaquers@isec.pl)
http://isec.pl/vulnerabilities/isec-0014-mremap-unmap.txt

[11] _syscall5 description
http://cs-pub.bu.edu/fac/richwest/cs591_w1/notes/wk1.pdf

[12] Remote Encrypted Callback UNIX Backdoor
http://www.securiteam.com/tools/5MP0L1PBPW.html

[13] Homepage of mrtg
http://people.ee.ethz.ch/~oetiker/webtools/mrtg/

[14] Zonelabs Integrity Suite product information
http://www.zonelabs.com/store/content/company/corpsales/intOverview.jsp

[15] Nortel Contivity: Tunnelguard
http://www.nortelnetworks.com/corporate/programs/cap/collateral/tunnelguard_configuration_v1.pdf

[16] InfoExpress Cybertgatekeeper
http://www.infoexpress.com/products/cgr_overview.php

[17] Network Associates
http://www.networkassociates.com/us/products/mcafee/host_ips/category.htm

[18] RFC 1918, Address Allocation for Private Internets
http://www.faqs.org/rfcs/rfc1918.html

[19] RSA Security: SecurID tokens
http://www.rsasecurity.com/products/securid/tokens.html

[20] Pluggable Authentication Modules
http://www.kernel.org/pub/linux/libs/pam/

[21] FreeRADIUS
http://www.freeradius.org

[22] Gnu RADIUS
http://www.gnu.org/software/radius/radius.html

Appendix A – Vulnerable Operating Systems

A blatant copy of the full lists of vulnerable operating systems from the Security Focus website: http://www.securityfocus.com/bid/9686

Linux kernel 2.2
Linux kernel 2.2.1
Linux kernel 2.2.2
Linux kernel 2.2.3
Linux kernel 2.2.4
Linux kernel 2.2.5
Linux kernel 2.2.6
Linux kernel 2.2.7
Linux kernel 2.2.8
Linux kernel 2.2.9
Linux kernel 2.2.10
  + Caldera OpenLinux 2.3
Linux kernel 2.2.11
Linux kernel 2.2.12
Linux kernel 2.2.13
  + S.u.S.E. Linux 6.3
  + S.u.S.E. Linux 6.4
Linux kernel 2.2.14
  + RedHat Linux 6.2
  + SCO eDesktop 2.4
  + SCO eServer 2.3.1
  + Sun Cobalt RaQ 4
Linux kernel 2.2.15 pre20
Linux kernel 2.2.15 pre16
Linux kernel 2.2.15
  + MandrakeSoft Corporate Server 1.0.1
  + MandrakeSoft Linux Mandrake 7.1
Linux kernel 2.2.16 pre6
Linux kernel 2.2.16
  + RedHat Linux 7.0
  + Sun Cobalt Qube 3
  + Sun Cobalt RaQ XTR
  + Trustix Secure Linux 1.1
Linux kernel 2.2.17
  + MandrakeSoft Linux Mandrake 7.2
  + S.u.S.E. Linux 7.0
  + Trustix Secure Linux 1.2
Linux kernel 2.2.18
  + Caldera OpenLinux 2.4
  + Conectiva Linux ecommerce
  + Conectiva Linux graficas
  + Conectiva Linux 4.0
  + Conectiva Linux 4.0 es
  + Conectiva Linux 4.1
  + Conectiva Linux 4.2
  + Conectiva Linux 5.0
  + Conectiva Linux 5.1
  + Conectiva Linux 6.0
  + Debian Linux 2.2
  + Debian Linux 2.2 68k
  + Debian Linux 2.2 alpha
  + Debian Linux 2.2 arm
  + Debian Linux 2.2 powerpc
  + Debian Linux 2.2 sparc

+ MandrakeSoft Linux Mandrake 6.0
+ MandrakeSoft Linux Mandrake 6.1
+ MandrakeSoft Linux Mandrake 7.0
+ MandrakeSoft Linux Mandrake 7.1
+ MandrakeSoft Linux Mandrake 7.2
+ RedHat Linux 6.0
+ RedHat Linux 6.0 alpha
+ RedHat Linux 6.0 sparc
+ RedHat Linux 6.1 alpha
+ RedHat Linux 6.1 i386
+ RedHat Linux 6.1 sparc
+ RedHat Linux 6.2 alpha
+ RedHat Linux 6.2 i386
+ RedHat Linux 6.2 sparc
+ RedHat Linux 7.0 alpha
+ RedHat Linux 7.0 i386
+ RedHat Linux 7.0 sparc
+ S.u.S.E. Linux 6.0
+ S.u.S.E. Linux 6.1
+ S.u.S.E. Linux 6.1 alpha
+ S.u.S.E. Linux 6.3
+ S.u.S.E. Linux 6.3 alpha
+ S.u.S.E. Linux 6.3 ppc
+ S.u.S.E. Linux 6.4
+ S.u.S.E. Linux 6.4 alpha
+ S.u.S.E. Linux 6.4 ppc
+ S.u.S.E. Linux 7.0
+ SCO eDesktop 2.4
+ SCO eServer 2.3.1
+ Slackware Linux 4.0
+ Slackware Linux 7.0
+ Slackware Linux 7.1
+ Wirex Immunix OS 6.2
+ Wirex Immunix OS 7.0
+ Wirex Immunix OS 7.0 -Beta

Linux kernel 2.2.19
+ EnGarde Secure Linux 1.0.1
+ Immunix Immunix OS 7+
+ MandrakeSoft Linux Mandrake 8.0
+ MandrakeSoft Linux Mandrake 8.0 ppc
+ MandrakeSoft Linux Mandrake 8.1
+ MandrakeSoft Single Network Firewall 7.2
+ S.u.S.E. Linux 6.3
+ S.u.S.E. Linux 6.4
+ S.u.S.E. Linux 7.0
+ Trustix Secure Linux 1.5

Linux kernel 2.2.20
+ MandrakeSoft Linux Mandrake 8.2
+ MandrakeSoft Linux Mandrake 8.2 ppc

Linux kernel 2.2.21
Linux kernel 2.2.22
+ Trustix Secure Linux 1.1
+ Trustix Secure Linux 1.2
+ Trustix Secure Linux 1.5

Linux kernel 2.2.23
Linux kernel 2.2.24
Linux kernel 2.4 .0-test9
Linux kernel 2.4 .0-test8
Linux kernel 2.4 .0-test7
Linux kernel 2.4 .0-test6
Linux kernel 2.4 .0-test5
Linux kernel 2.4 .0-test4

Linux kernel 2.4 .0-test3
Linux kernel 2.4 .0-test2
Linux kernel 2.4 .0-test12
Linux kernel 2.4 .0-test11
Linux kernel 2.4 .0-test10
Linux kernel 2.4 .0-test1
Linux kernel 2.4
Linux kernel 2.4.1
Linux kernel 2.4.2
  + Caldera OpenLinux Server 3.1
  + Caldera OpenLinux Workstation 3.1
  + RedHat Linux 7.1 alpha
  + RedHat Linux 7.1 i386
Linux kernel 2.4.3
  + MandrakeSoft Linux Mandrake 8.0
  + MandrakeSoft Linux Mandrake 8.0 ppc
Linux kernel 2.4.4
  + S.u.S.E. Linux 7.2
Linux kernel 2.4.5
  + Slackware Linux 8.0
Linux kernel 2.4.6
Linux kernel 2.4.7
  + RedHat Linux 7.2
  + S.u.S.E. Linux 7.1
  + S.u.S.E. Linux 7.2
Linux kernel 2.4.8
  + MandrakeSoft Linux Mandrake 8.0
  + MandrakeSoft Linux Mandrake 8.1
  + MandrakeSoft Linux Mandrake 8.2
Linux kernel 2.4.9
  + RedHat Enterprise Linux AS 2.1
  + RedHat Enterprise Linux AS 2.1 IA64
  + RedHat Enterprise Linux ES 2.1
  + RedHat Enterprise Linux ES 2.1 IA64
  + RedHat Enterprise Linux WS 2.1
  + RedHat Enterprise Linux WS 2.1 IA64
  + RedHat Linux 7.1 alpha
  + RedHat Linux 7.1 i386
  + RedHat Linux 7.1 ia64
  + RedHat Linux 7.2 alpha
  + RedHat Linux 7.2 i386
  + RedHat Linux 7.2 ia64
  + Sun Linux 5.0
  + Sun Linux 5.0.3
  + Sun Linux 5.0.5
Linux kernel 2.4.10
  + S.u.S.E. Linux 7.3
Linux kernel 2.4.11
Linux kernel 2.4.12
  + Conectiva Linux 7.0
Linux kernel 2.4.13
  + Caldera OpenLinux Server 3.1.1
  + Caldera OpenLinux Workstation 3.1.1
Linux kernel 2.4.14
Linux kernel 2.4.15
Linux kernel 2.4.16
  + Sun Cobalt RaQ 550
Linux kernel 2.4.17
Linux kernel 2.4.18 pre-8
Linux kernel 2.4.18 pre-7
Linux kernel 2.4.18 pre-6
Linux kernel 2.4.18 pre-5

Linux kernel 2.4.18 pre-4
Linux kernel 2.4.18 pre-3
Linux kernel 2.4.18 pre-2
Linux kernel 2.4.18 pre-1
Linux kernel 2.4.18 x86
  + Debian Linux 3.0 ia-32
Linux kernel 2.4.18
  + Astaro Security Linux 2.0 16
  + Astaro Security Linux 2.0 23
  + Debian Linux 3.0 alpha
  + Debian Linux 3.0 arm
  + Debian Linux 3.0 hppa
  + Debian Linux 3.0 ia-32
  + Debian Linux 3.0 ia-64
  + Debian Linux 3.0 m68k
  + Debian Linux 3.0 mips
  + Debian Linux 3.0 mipsel
  + Debian Linux 3.0 ppc
  + Debian Linux 3.0 s/390
  + Debian Linux 3.0 sparc
  + MandrakeSoft Linux Mandrake 8.0
  + MandrakeSoft Linux Mandrake 8.1
  + MandrakeSoft Linux Mandrake 8.2
  + RedHat Advanced Workstation for the Itanium Processor 2.1 IA64
  + RedHat Enterprise Linux AS 2.1 IA64
  + RedHat Linux 7.3
  + RedHat Linux 8.0
  + S.u.S.E. Linux 7.1
  + S.u.S.E. Linux 7.2
  + S.u.S.E. Linux 7.3
  + S.u.S.E. Linux 8.0
  + S.u.S.E. Linux 8.1
  + S.u.S.E. Linux 8.2
  + S.u.S.E. Linux Connectivity Server
  + S.u.S.E. Linux Database Server
  + S.u.S.E. Linux Enterprise Server 7
  + S.u.S.E. Linux Enterprise Server 8
  + S.u.S.E. Linux Firewall on CD
  + S.u.S.E. Linux Office Server
  + S.u.S.E. Linux Openexchange Server
  + S.u.S.E. SuSE eMail Server 3.1
  + S.u.S.E. SuSE eMail Server III
  + Turbolinux Turbolinux Server 7.0
  + Turbolinux Turbolinux Server 8.0
  + Turbolinux Turbolinux Workstation 7.0
  + Turbolinux Turbolinux Workstation 8.0
Linux kernel 2.4.19 -pre6
Linux kernel 2.4.19 -pre5
Linux kernel 2.4.19 -pre4
Linux kernel 2.4.19 -pre3
Linux kernel 2.4.19 -pre2
Linux kernel 2.4.19 -pre1
Linux kernel 2.4.19
  + Conectiva Linux 8.0
  + Conectiva Linux Enterprise Edition 1.0
  + MandrakeSoft Corporate Server 2.1
  + MandrakeSoft Corporate Server 2.1 x86_64
  + MandrakeSoft Linux Mandrake 9.0
  + MandrakeSoft Multi Network Firewall 8.2
  + S.u.S.E. Linux 8.1
  + Slackware Linux -current
Linux kernel 2.4.20

+ CRUX CRUX Linux 1.0
+ Gentoo Linux 1.2
+ RedHat Linux 9.0 i386
+ Slackware Linux 9.0
+ WOLK WOLK 4.4 s

Linux kernel 2.4.21 pre7
Linux kernel 2.4.21 pre4
+ MandrakeSoft Linux Mandrake 9.1
+ MandrakeSoft Linux Mandrake 9.1 ppc

Linux kernel 2.4.21 pre1
Linux kernel 2.4.21
+ Conectiva Linux 9.0
+ MandrakeSoft Linux Mandrake 9.1
+ MandrakeSoft Linux Mandrake 9.1 ppc
+ S.u.S.E. Linux 9.0
+ S.u.S.E. Linux 9.0 x86_64
+ S.u.S.E. Linux Enterprise Server 8

Linux kernel 2.4.22
+ Devil-Linux Devil-Linux 1.0.4
+ Devil-Linux Devil-Linux 1.0.5
+ MandrakeSoft Linux Mandrake 9.2
+ MandrakeSoft Linux Mandrake 9.2 amd64
+ RedHat Fedora Core1
+ Slackware Linux 9.1

Linux kernel 2.4.23 -pre9
Linux kernel 2.4.23
+ Trustix Secure Linux 2.0

Linux kernel 2.4.24
Linux kernel 2.6 -test9-CVS
Linux kernel 2.6 -test9
Linux kernel 2.6 -test8
Linux kernel 2.6 -test7
Linux kernel 2.6 -test6
Linux kernel 2.6 -test5
Linux kernel 2.6 -test4
Linux kernel 2.6 -test3
Linux kernel 2.6 -test2
Linux kernel 2.6 -test11
Linux kernel 2.6 -test10
Linux kernel 2.6 -test1
Linux kernel 2.6
Linux kernel 2.6.1 -rc2
Linux kernel 2.6.1 -rc1
Linux kernel 2.6.2
Netwosix Netwosix Linux 1.0
RedHat kernel-2.4.20-8.athlon.rpm
+ RedHat Linux 9.0 i386

RedHat kernel-2.4.20-8.i386.rpm
RedHat kernel-2.4.20-8.i686.rpm
+ RedHat Linux 9.0 i386

RedHat kernel-bigmem-2.4.20-8.i686.rpm
+ RedHat Linux 9.0 i386

RedHat kernel-BOOT-2.4.20-8.i386.rpm
+ RedHat Linux 9.0 i386

RedHat kernel-doc-2.4.20-8.i386.rpm
+ RedHat Linux 9.0 i386

RedHat kernel-smp-2.4.20-8.athlon.rpm
+ RedHat Linux 9.0 i386

RedHat kernel-smp-2.4.20-8.i686.rpm
+ RedHat Linux 9.0 i386

RedHat kernel-source-2.4.20-8.i386.rpm
+ RedHat Linux 9.0 i386

SGI ProPack 2.4
SmoothWall Corporate Guardian 0.3 (Fixes2)
SmoothWall Corporate Server 0.3 (Fixes7)
SmoothWall Express 2.0
Trustix Secure Linux 1.5
Trustix Secure Linux 2.0
VMWare ESX Server 1.5.2
VMWare ESX Server 2.0
VMWare ESX Server 2.0.1