# INCORE Metadatabase Server - Technical Aspects and Constraints

*by Patrick Curran*[1]
*INCORE, University of Ulster, N. Ireland.*

The INCORE server was set up by the University of Ulster and the United Nations University to act as a central resource for academics, policy-makers and others concerned with conflict resolution and ethnicity across the globe. The server utilises Internet resources to make as widely available as possible information on ethnic conflict. However, like a lot of tools the Internet can be used for more than one purpose. In one sense it is a very flexible and open resource that allows users to log into numerous systems around the world, transferring files, searching file systems and executing programs. However once you are connected to the Internet it is also a way for other people to get into your system. If you restrict your workstation's access only to parts of the Internet , then you may find that there is a lot of useful resources and facilities that you can no longer access and use. In order to take advantage of the Internet you must be a part of it. However, as this paper will try to demonstrate, by doing this you put your computer at risk, so you need to constantly keep abreast of the security holes in the network software installed in your system and protect it.

**Background**
INCORE [2] was established by the University of Ulster (UU) and the United Nations University (UNU) to provide a systematic approach to the study of ethnic conflict and to encourage links between research, training, policy, practice and theory.

The INCORE metadatabase server [3] was set up to act as a central resource for academics, policy-makers and others concerned with conflict resolution and ethnicity across the globe, but particularly to those operating in areas of conflict. It is central to INCORE's mandate that it serve those who have most difficulty in getting access to information. In this respect the Internet can be both the most and least useful medium for connecting people to information. The advantage of placing information on the Internet is that it is almost instantaneously available, uncensored, to anyone anywhere in the globe who is connected to the net. The disadvantage is that, unless one is connected to the Internet, which involves expensive hardware, technical expertise and effective maintenance, this information is inaccessible. INCORE has to be concerned not merely to provide a state of the art server but also to ensure that as much of the information as possible on the server is accessible to those with the least connectivity to the Internet, through E-mail and FTP as well as through gopher and World Wide Web. However in setting up this server we need to be aware of adopting a responsible attitude towards security in order to maintain a continued uninterrupted service. We need to consider where our system is at risk [4], where the possible threats and breaches of security are coming from and how to prevent these becoming problems.

**Introduction**
As the volume of Internet use increases dramatically (30,000 plus interconnected networks with 2.5 million or more connected computers daily swap gigabytes of information based on nothing more than a digital handshake with a stranger) [5, 6] and as we connect our organisation's networks to thousands of other computer networks, serious security issues arise. There are a lot of considerations for the system administrator and the following list outlines some of the main issues :

- Unauthorised users accessing data and information on our system
- Authorised users causing inadvertent or malicious damage
- Interception of data transmission
- Virus attacks
- Backup and storage policy

It is difficult to fully understand and appreciate just how much the Internet depends on collegial trust and a group effort by the whole community. One technique, for instance, that intruders use is to break into a chain of computers, e.g., break into A, use A to break into B, B to break into C, etc., thus covering their tracks. So, it would be very unwise and foolish to think that your little part of the network is safe because you believe that there is very little information there of a nature that would entice someone to break into it. Even if there is nothing of use on your computer, it could prove a worthwhile intermediate staging post for someone who wants to break into other more useful systems.

In the early days of the ARPAnet only researchers had access to the net, and they shared a common set of goals and ethics. Data packets were forwarded along network links from computer to computer. A packet may have made a number of hops and every intermediate machine could read its contents. Nowadays many Internet packets start their journey on a local area network (LAN), where privacy is even less protected. Only a gentleman's agreement assures the sender that the recipient and no-one else will read the message. The lack of security on the ARPAnet did not bother anyone, because that was part of the package. As the Internet developed and expanded, the user population began to change, with a lot of the newcomers having little idea of the importance of the complex social contract guiding the use of this new and exciting tool. Nowadays anyone with a computer, a modem and a small monthly connection fee can have a direct link to the Internet and be subject to break-ins or launch attacks on others.

Every day computer networks and hosts are being broken into with varying levels of sophistication. While it's generally believed that most break-ins succeed due to weak passwords, there are advanced and sophisticated techniques that are more difficult to detect. The article looks firstly at the shortcomings of the password mechanism (concentrating on the Unix system) and then discusses the more sophisticated techniques available to intruders, including some examples of the misuse of commonly used Internet resources, e.g., ftp, sendmail, telnet, rlogin, rsh, etc. It also analyses the range of responses to network intrusion techniques, from software policing solutions like Kerberos and COPS to the hardware solution of firewall installation.

**Security Breaches**

System administrators can safely configure workstations on their network to allow connections to other workstations. They can also set up their network file system to export widely used file directories to "world", allowing everyone to read them. It doesn't take much imagination to see what can happen when such a trustworthy environment opens up its digital doors to the Internet. Suddenly, 'world' means the entire globe and "any computer on the network" means "every computer on any network". There have been a lot of computer security problems and breaches of security in recent years, some more serious than others. Some of the more widely known incidents include break-ins on NASA's SPAN network [7] and the IBM "Christmas Virus", but the most widespread breach of network security occurred in 1988 when the Internet came under attack from within, later to be known as the Internet Worm Incident.

**Internet Worm**

On November 2, 1988, a self-replicating program, called a worm* appeared on the Internet [8] . This program copied itself from machine to machine, causing the machines it infected to labour under huge loads, and denying service to the users of those machines. The program spread quickly, and while many system administrators were aware that something like this could theoretically happen (the security holes exploited by the worm were well known) the scope of the worm's break-ins came as a great surprise to many.

The worm itself did not destroy any files, steal any information (other than account passwords), intercept private mail, or plant other destructive software [9]. However, it did manage to severely disrupt the operation of the network. Several sites, including parts of MIT, NASA's Ames Research Centre and Goddard Space Flight Centre, the Jet Propulsion Laboratory, and the Army Ballistic Research Laboratory, disconnected themselves from the Internet to avoid recontamination. In addition, the Defense Communications Agency ordered the connections between the MILNET and ARPANET shut down, and kept them down for nearly 24 hours [10]. Ironically, this was perhaps the worst thing to do, since the first fixes to combat the worm were distributed via the network.

This incident was perhaps the most widely described computer security problem ever. The worm was covered in many newspapers and magazines around the country including the New York Times and most computer oriented technical publications.

**Security Considerations**

The incidents above demonstrate quite clearly that computer security is an important topic. Every day computer networks and hosts are being broken into with varying levels of sophistication. When you hear the term "security" the first thing that comes to mind is the password. However, while it's generally believed that most break-ins succeed due to weak passwords, there are, moreover, a large number of unauthorised attacks that use more advanced and sophisticated techniques. Less is known about these latter techniques because they are more difficult to detect. We will look firstly at the advantages and disadvantages of the password mechanism (concentrating on the Unix system [11]) and then discuss the more sophisticated techniques available to intruders.

**Passwords**
An underlying goal of the Unix system has been to provide password security at a minimal inconvenience to the users of the system. For example, those who want to run a completely open system without passwords, or to have passwords only at the option of the users, can do so, whilst those who require all their users to have passwords gain a high degree of security against penetration of their system by unauthorised users. A good password system must be able not only to prevent access to the system by unauthorised users, but it must also prevent users already logged on from doing things that they are not authorised to do. The "super user" password, e.g., is especially critical because it allows all sorts of permissions and provides unlimited access to all system resources.

Passwords are important because they are generally the first line of defence against interactive attacks. In simple terms, if a cracker[@] cannot interact with your system, and he has no access to read or write to the information contained in the password file, then he has almost no avenue of attack left open to break your system. Unfortunately the Unix passwd program [12] doesn't place a lot of restrictions on what might be used as a password. It generally requires 5 lowercase letters ( or 4 characters) but if the user insists on using a shorter password (by entering it three times) the program allows it.

The object when choosing a password is to make it difficult for a cracker to make educated guesses, thus leaving him with no alternative but to try every possible combination of letters, characters, special characters and numbers. A search of this sort, even on the most powerful computer would take at least 100 years to complete.

Robert Morris and Ken Thomson carried out an interesting survey determining typical users' habits in the choice of passwords [13]. Out of 4,000 passwords 16% contained 3 characters or less, and 86% were what could generally be described as insecure. Grammp & Morris [14] in another experiment showed that by trying the 20 most common female names, followed by a single digit, at least 1 password was valid in each of the machines surveyed. They also found that by trying variations of the login name, user's first and last name, and a list of nearly 1800 common first names, that up to 50% of the passwords on any given system can be cracked in a matter of 2 or 3 days.

There are ways to improve the security of using passwords. According to Schweitzer [15] a high quality password has the following characteristics :

At least eight characters length
The password is randomly generated
The password has no personal relationship to the user, his family or job
The password must be kept secret
It must be changed at least every three months

The Internet worm, in trying to break into new systems attempted to crack user passwords [7]. First of all it tried simple choices (user names, names, etc) and then tried an internal dictionary of 432 words. If all failed it tried going through the system dictionary /usr/dict/words, trying each in turn. So, by adhering to the guidelines above, you will make it very difficult for an intruder to break into your system using the first line of defence.

Other problems associated with passwords are :

*Expired accounts* : Accounts lying around due to people leaving the organisation. These cause problems because since nobody is using the account anymore it's unlikely that a break-in will be noticed. A simple preventative measure is to place an expiration date on every account. If there is any doubt about an account, then replacing the encrypted password with an asterisk (*) will make it impossible for anyone to log into the account.

*Guest accounts* : These are usually made available for expediency. They should only be used for the period required and deleted immediately. It's important not to give them simple passwords, e.g., "guest" or "temp".

**Network Security**
Besides cracking passwords on machines crackers have other means at their disposal, the success of which depends on your awareness and adoption of preventative measures. It's important to be careful about techniques that bypass password requirements. There are two common ones, i.e., the .rhosts and the hosts.equiv files :

hosts.equiv: the file /etc/hosts.equiv can be used to indicate trusted hosts. If a user remotely logs into your system using rlogin and does so from a host listed in this file, access is permitted without requiring a password. The default

file has the entry '+' in a single line indicating that every host should be considered a trusted host. This could prove a major security problem since hosts outside the local organisation should never be trusted. Only specific host names should be included in the file, or the file deleted altogether.

*.rhosts:* allows access to specific host-user combinations. Each user may create a .rhosts file in his/her directory, allowing access to their account without supplying a password. For example, the entry - host.com fred - tells the computer on which the file resides to bypass password requirements when it sees someone logging in from the login name "fred" on the machine "host.com". This means, obviously, that anyone who manages to break into fred's account can also break into this machine.

The Internet worm made use of the trusted host concept to spread itself throughout the network [8].

**Secure Terminals**
Unix introduced the concept of a "secure" terminal, which prohibits 'root' from logging in from a non-secure terminal. The file /etc/ttyab controls which terminals are considered secure. The default is to consider all terminals secure allowing root to login from anywhere in the network. A more secure configuration would be to consider as secure only directly connected terminals, or only the console device. The most secure method is to remove the secure designation from all terminals including the console, so users with 'root' privilege must first login as themselves and then use the 'su' command.

**NFS**
The Network File System (NFS) allows several hosts to share files over the network, generally used to provide file server access to diskless workstations in a small network. The file /etc/exports lists which file systems are exported. This file contains access specifications, e.g. :

*root=keyword* specifies the list of hosts allowed "super-user" access to the files in the named file system

*access=keyword* specifies the list of hosts that are allowed to mount the named file system.

For example, the line - /export/root/client -access=client,root=client - allows the host "client" to access the named file system with root privileges. If the file isn't properly configured then anyone on the Internet may have access to your files via NFS, whether you trust them or not.

**E-Mail**
Whilst this is one of the net's basic services, forging e-mail is a trivial exercise. An electronic letter consists simply of a text file with a header specifying the sender, receiver, subject, date and routing information, followed by a blank line and the message body. Mail programs fill in the header lines with routing information, but there's nothing to stop a malicious person inserting whatever he wants into the mail. Protocols have been developed to verify the source of e-mail messages, but spoofers are also improving their techniques. Some systems bar connection to their system from untrustworthy parts of the Internet. The problem with this strategy is that the trusted "domain name servers" they rely on are just ordinary computers, and as such are also vulnerable to deception or intrusion. A cracker can modify the name server's database so that it tells any computer querying it that the address belonging to e.g.,"cracker@breach.com" is instead that of"president@whitehouse.gov". A computer allowing connection from whitehouse.gov will allow the cracker in as well.

The "sendmail" bug has reappeared time and time again over the years, due to the fact that most mail programs make it possible to route messages not only to users but also directly to particular files or programs. People, e.g., forward mail to a program called "vacation" which sends a reply telling the correspondent that the recipient is out of town. Other people route mail through filter programs that forward the message to various locations. This same mechanism can thus be subverted to send e-mail to programs that are designed to execute shell scripts. Such a script could cause a copy of the receiving computer's password file to be sent to an intruder for analysis, or simply wreak havoc on the recipient's file system.

**Intrusion Techniques**
Many system administrators are often unaware of the dangers presented by anything beyond the most trivial attacks. The purpose of this section is to present a few of the techniques available to the system cracker in his efforts to gain access to a shell[#] process on a Unix host [16]. There are so many methods and techniques that it would be impossible to cover them all in this paper. However, I will try to outline some of the most common techniques. Fig. 1 illustrates some of the more useful services and facilities that an intruder may avail of to ferret out information about a system and set up an interactive link with that system.

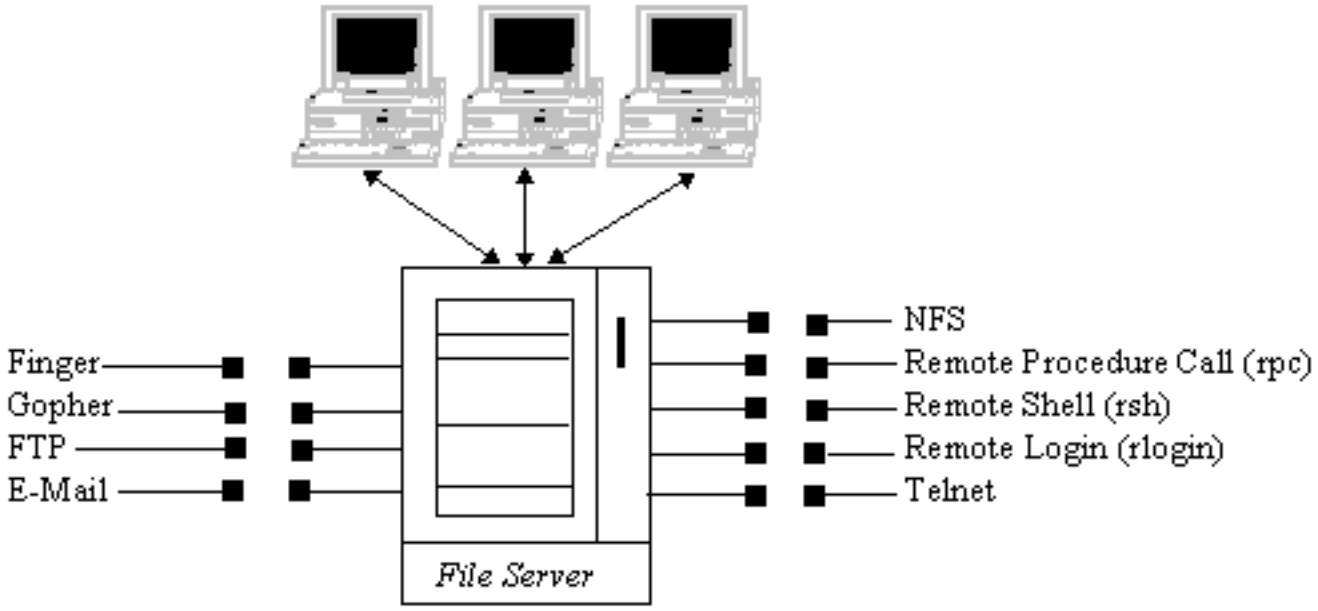


**Fig. 1 : Network Intrusion Techniques**

The finger services, provided by the finger program outputs information about the users logged on to the system, and the fingerd program extends this facility to remote hosts, e.g.,

```
host% finger@incore.ac.uk
Login     Name      TTY  Idle When      Where
pat   P.Curran co   21    Fri 10:38 i    inf.ac.uk
```

The most revealing information divulged are the account names, home directories and the host they last logged in from. This information can be supplemented by using the rusers command (with the -l flag), which produces, e.g. :

| Login | Home-dir | Shell | Last login, from where |
|---|---|---|---|
| root | / | /bin/sh | Fri Apr 15 from inf.ac.uk |
| pat | /home/pat | /bin/csh | On since Thur Apr 14 from inf.ac.uk |
| guest | /export/guest | /bin/sh | Never logged in |
| ftp | /home/ftp | | Never logged in |

Finger is one of the most dangerous services because it is very useful for investigating and receiving information about possible target machines, especially when used in conjunction with other data. A bug in the fingerd program was exploited by the Internet worm to overrun the buffer that the daemon (background process that fingerd is intended to run as) used for input, thus altering the behaviour of the program.

The *showmount* command can be used on an NFS fileserver to display the names of all the hosts that currently have something mounted from the server. Running showmount on a target, e.g., reveals :

```
host%  showmount -e incore.ac.uk
export list for incore.ac.uk
/export       (everyone)
/var          (everyone)
```

Since /export/guest is exported to the world, and this is the user guest's home directory, we have a possible break-in scenario. The intruder could mount the home directory of ëguest', and create a ëguest' account in his local password file. By putting a

.rhosts entry in the remote guest home directory he allows himself to login to the target machine without having to supply a password.

If the target machine has a '+' wildcard in its /etc/hosts.equiv file, then any non-root user with a login name on the target's password file can rlogin to the target without a password. So the intruder's next line of attack would be to login to the target host and modify the password file to allow root access :

```
host%  rsh incore.ac.uk csh -i
host%  ls -ldg /etc
drwxr-xr-x    8bin    staff    2048 Jul 24 18:02  /etc
host%  cd /etc
host%  mv passwd pw.old
host%  (echo attack::0:1:instant root shell:/:/bin/sh;cat pw.old)> passwd
host%  ^D
host%  rlogin incore.ac.uk -l attack
Welcome to the Incore Server !!
incore#
```

The rsh -i gets one on to the system but doesn't leave any traces in the wtmp or utmp system auditing files, making the remote shell invisible to the finger and who commands.

Going back to the finger output we can see that there is an 'ftp' account, which usually means that anonymous ftp is enabled. The File Transfer Protocol (ftp) allows users to connect to remote systems and transfer files back and forth. This can be an easy way to get access to a system as it is often misconfigured. In some cases the target machine has a complete copy of the /etc/passwd file in the anonymous ftp ~ftp/etc directory. If, however, this isn't the case then there are other avenues for the would-be attacker. If, for instance, the ftp directory is writable, an intruder can remotely execute a command, e.g., mailing the password file back to himself simply by creating a .forward file that executes a command when mail is sent to the ftp account.

If none of the methods above have worked then the attacker could use rpcinfo to see if the target is running NIS or NFS. Once you know the NIS domainname of a server you can get any of its NIS maps using a simple rpc query. Also, just like easily guessed passwords many systems use easily guessed NIS domainnames [17]. The showmount output usually divulges information on domainnames, which can then be checked with the 'ypwhich' command to see if the domainname exists, e.g.

```
host%  ypwhich -d incore incore.ac.uk
Domain incore not bound
```

This proved unsuccessful, but if it was guessed properly it would have returned with the hostname of incore.ac.uk's NIS server. If an attacker has control of the NIS master then he effectively has control of the client hosts, and can execute arbitrary commands, e.g., mailing password files to himself.

**Security Tools**

There are a lot of tools available that deal with the security management aspects of computer systems, more than can be adequately covered in this article [18]. Because there are so many tools and techniques available to implement security controls you should, in the first place, identify the requirements of your network service and what you are willing to accept. The following sections provide only a flavour of the types of solutions available, from software based system auditing, authentication and checking techniques to the hardware solution of firewall installation.

**Computer Oracle and Password System (COPS)**

COPS [19] is a UNIX security status checker, written as a suite of shell scripts which forms an extensive security testing system. Basicially it checks various files and software configurations to see if they have been compromised (e.g., edited to plant a trojan horse[&] or back door[$] ), and checks to see that files have the appropriate modes and permissions set to maintain the integrity of your security level (making sure that your file permissions don't leave themselves wide open to attack). There's a rudimentary password cracker, and routines to check the file store for suspicious changes in setuid programs, and to identify software behaving in ways which could cause problems.

The current version of COPS makes a limited attempt to detect bugs that are posted in CERT advisories. Also, it has an option to generate a limited script that can correct various security problems that are discovered.

**Kerberos**

Kerberos [20, 21] is a DES-based encryption scheme that encrypts sensitive information, such as passwords, sent via the network from client software to the server daemon process. When a user logs in, Kerberos authenticates that user (using a password), and provides the user with a way to prove her identity to other servers and hosts scattered around the network. This authentication is then used by programs such as rlogin to allow the user to log in to other hosts without a password (in place of the .rhosts file). The authentication is also used by the mail system in order to guarantee that mail is delivered to the correct person, as well as to guarantee that the sender is who he claims to be. The overall effect of installing Kerberos and the numerous other programs that go with it is to virtually eliminate the ability of users to "spoof" the system into believing they are someone else.

**Firewall**

A firewall is a machine which is usually attached between your site and a wide area network. It provides controllable filtering of network traffic, allowing restricted access to certain internet port numbers (i.e., services that your machine would otherwise provide to the network as a whole) and blocks access to pretty well everything else. They are an effective "all-or-nothing" approach in dealing with external access security, and are fast are becoming very popular, particularly with the rise in Internet connectivity [22].

The firewall doesn't send out routing information about the internal network, making the internal network "invisible" from the outside. It doesn't advertise routes which means that users on the internal network must log in to the firewall before accessing hosts on remote networks. Also, in order to remotely log in to a host on the internal network from the outside, a user must first log in to the firewall machine, which may prove to be inconvenient, but, nevertheless, more secure.

Outgoing e-mail is forwarded to the firewall machine before being delivered outside the internal network and the firewall receives all incoming e- mail, before redistributing it. It provides extra security by not mounting any file systems via NFS, or making any of its file systems available to be mounted. Password security is rigidly enforced and the firewall does not trust any other hosts regardless of where they are. Finally, anonymous ftp and other similar services is only provided by the firewall host, if at all.

The purpose of the firewall is to prevent crackers from accessing other hosts on your network which means that security must be strictly and rigidly enforced. It is important to remember that a firewall can't provide complete safeguards against intrusion - if someone manages to subvert the firewall then he can subsequently break into any host behind it.

Many organisations, and more recently universities, have adopted firewalls to examine the packets entering and leaving a domain and to restrict certain Internet connections. However, proposing a firewall and constructing it are two different things entirely. There are some things that you just can't do securely. Gopher and Mosaic, for instance, are two programs of a trusting nature that defy the attempts of a firewall design to provide safety. Additionally, a firewall must pass mail and

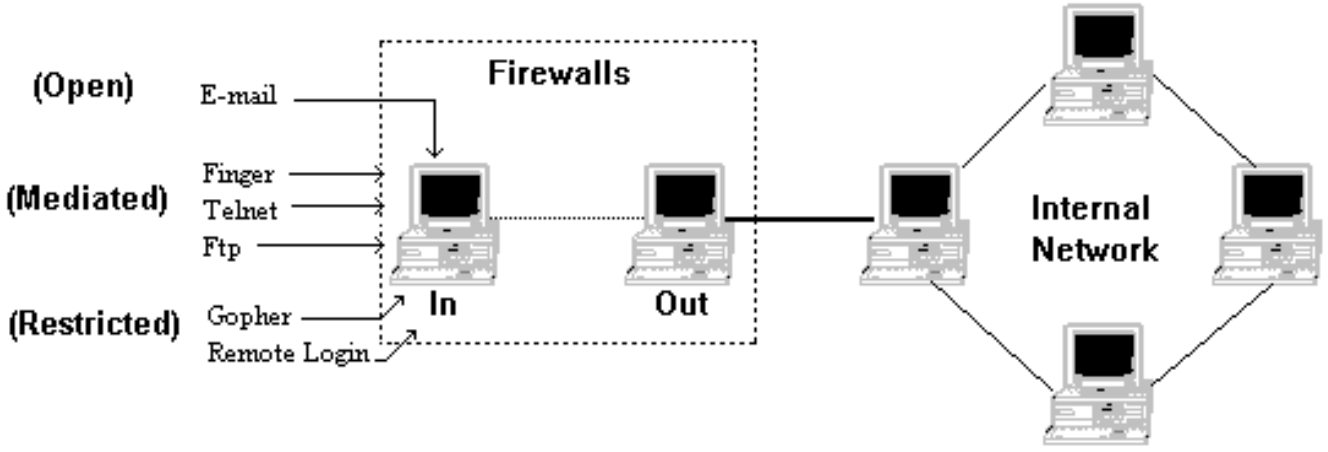


**Fig. 2 : Firewall Set-up**

mailers can be very insecure. Users also need to log into log into various public archive sites to receive files. One solution to provide this type of functionality is outlined in Fig. 2, where two dedicated computers or gateways are used - one connected to the local network and the other to the Internet :

The external computer examines all incoming traffic, forwarding only “safe” packets to the internal machine. An attacker , thus, could only break into machines on the local network by compromising the internal gateway. The internal gateway only accepts messages from the external computer, so that, if unauthorised packets do reach it they will not be able to pass.

Conventional password techniques, when used with firewalls, reduce the efficiency of the overall security provided. Hackers gained access to Panix, a public-access Internet site in New York (Oct. 1993), and installed “packet sniffers”. These programs watched data going by and recorded user names and passwords as people logged on to hundreds of other computer systems. Connections, therefore, within a firewall require, for ultimate security, a different kind of authentication mechanism that cannot be recorded by sniffers, e.g., “one-time password” or “challenge-response password”.

**Conclusions**

Even if newcomers to the Internet try to secure their systems it’s not always easy to find the information they need. Hardware and software vendors are usually loathe to discuss their security problems. CERT [23] generally issue advisories only after manufacturers have developed a definitive fix - usually weeks or months later. Spafford [10] points out that people don’t know the risks - between half and three quarters of the security holes currently known to hackers have yet to be openly acknowledged. People only know the benefits. Many of these benefits come from programs such as Gopher, Netscape, World Wide Web or Mosaic, which allow simple menu-driven navigation of the Internet. These tools are drawing thousands of people to the net. Yet, the rapid evolution of these tools have bypassed steps that could lead to security breaches. The popular Gopher problem, for instance, according to CERT advisories, has security problems that make it possible to access not only public files but private ones as well. Again, Gopher is only insecure if it is misconfigured. Whilst Gopher servers can be confined so that they have access only to public information, by default they have a free rein.

The Internet attracts more recruits every day hoping to reap such rewards and benefits as connections with other people and organisations, file access and information interchange. Yet, Internet connection can also prove to be the source of very real risks and dangers to your workstation or network. While this article, hopefully, raises the reader’s awareness of the importance of security in a network situation, one shouldn’t get paranoid. It’s worth remembering that security, in most cases, is an elusive goal. Don’t forget as well that the nature of Unix and the Internet helped to defeat the Internet worm as well as spread it.

The sensible approach is to secure your system according to its needs, keeping danger at a manageable level. In other words, don’t stop travelling but do wear a seat belt.

**References**

[1] Paper presented at IASSIST95 May 1995 Quebec City, Quebec, Canada.

[2] “Data on Ethnic Conflict and Conflict Resolution : The Work of INCORE”, Journal of Ethno- Development, Forthcoming Publication

[3] Metadata on Ethnic Conflict and Conflict Resolution - INCORE Metadatabase Project, Iassist Quarterly, Forthcoming Publication

[4] Kay, R. : “Distributed and Secure”, BYTE, pp.165-178, (June 1994)

[5] MIDS, :”MIDS Press Release : New Data on the Size of the Internet and the Matrix”, Matrix News, Vol. 5, No. 1, 1.5. Matrix Information and Directory Services, Inc., Austin, (January 1995), Available from mids@tic.com

[6] Lottor, DNS ZONE SURVEY (RFC 1296) through January 1995

[7] McLellan, V. : “NASA Hackers : There’s More to the Story”, Digital Review, p. 80. (Nov. 1987)

[8] Spafford, E.H. : “The Internet Worm Program : An Analysis”, Computer Communications Review, 19, No. 1, ACM SIGCOM, (January 1989)

[9] Seeley, D. : “A Tour of the Worm”, Proceedings of 1989 Winter Usenix Conference, Usenix Association, San Diego, California, (February 1989)

[10] Eichin, M.W., Rochlis, J.A. : “With Microscope and Tweezers : An Analysis of the Internet Virus of November 1988”, Proceedings of the Symposium on Research in Security and Privacy, IEEE-CS, Oakland, California, (May 1989)

[11] Garfinkel, S., Spafford, S. :”Practical Unix Security”, O’Reilly & associates (1992)

[12] Todino, G., Strang, J., Peek, J. : “Learning the Unix Operating System”, O’Reilly & Associates (1991)

[13] Morris, R., Thomson, K. : “Password Security : A Case History”, Communications of the ACM, 22(11), pp. 594-597, (November 1979). Reprinted in Unix System Manager’s Manual, 4.3 Berkeley Software Distribution, University of California, Berkeley (1986)

[14] Grammp, F.T., Morris, R. : “Unix Operating System Security”, AT&T Bell Laboratories Technical Journal, 63 (8), pp. 1649-1672, (October 1984)

[15] Schweitzer, J.A. : “Managing Information Security. Administrative, Electronic and Legal Measures to Protect Business Information”, 2nd. ed. Massachussetts, Butterworth Publishers (1990)

[16] Farmer, D., Venema, W :”Improving the Security Of Your Site By Breaking Into It”, Available by ftp from win.tue.nl as /pub/security/admin-guide-to-cracking.Z

[17] Schuba, C. : “Addressing Weaknesses in the Domain Name System Protocol”, Purdue University (August 1993)

[18] Reinhard, R.B. :”An Architectural Overview of Unix Network Security (Specifically oriented towards Internet Connectivity)”, v.4 (Feb. 1993), Available from gopher.near.net in /security/papers

[19] Available via anonymous FTP from cert.org in ~/pub/tools/cops.

[20] Available via anonymous FTP from athena-dist.mit.edu in ~/pub/kerberos

[21] Bellovin, :”Limitations of the Kerberos Authentication System”, Available via anonymous ftp from research.att.com

[22] Ranum, M. :”Thinking About Firewalls”, Proceedings of the 2nd. International Conference on Systems and Network Security and Management, Available via anonymous ftp from ftp.tis.com as /pub/firewalls/firewall .ps.Z

[23] The Computer Emergency Response Team (CERT) advisories are available by ftp from cert.org

FOOTNOTES * a worm is an independently operating program that actively propogates by spreading copies of itself throughout a network.

@ In Usenet parlance a ”*hacker*” is a person possessing a great deal of knowledge and expertise, and exercises this with great finesse, whereas a “cracker” is a person who persistently breaks into other people’s computer systems, for a variety of reasons. For further information refer to Steele, G.L., Woods, D.R., Finkel, R.A., Crispin, M.R., Stallman, R.M., Goodfellow, G.S. :”The Hacker’s Dictionary”, New York: Harper and Row, 1988.

# a *shell script* is a Unix program containing a series of commands that perform system functions.

& a *block of undesired code* (intentionally hidden within a desireable block of code) which does things that the user does not intend, e.g., a program that simulates a computer’s logon procedure, but, rather than logging the user on, it simply records and steals the user’s id and password.

$ a feature built into programs by the designers allowing them special privileges which are denied to the normal users of the program, e.g., a *back door* in a logon program would enable the designer to log onto a system without an authorised account.