The following paper was originally published in the

Proceedings of the 8th USENIX Security Symposium

Washington, D.C., USA, August 23–26, 1999

A SECURE STATION FOR NETWORK MONITORING AND CONTROL

Vassilis Prevelakis

A Secure Station for Network Monitoring and Control

Vassilis Prevelakis
vp@unipi.gr
Network Management Centre
University of Piraeus, Greece

Abstract

The task of managing large networks is seldom accomplished using a single tool. Instead, network administrators usually compile collections of tools that can be used, in isolation or combined with others, to identify and correct problems. Equally important, however, is the platform used for the execution of those tools. In this paper we will be describing the Network Monitoring Station which has provided us with a safe base from which to troubleshoot network problems. This platform combines strong security with reduced configuration, administration and maintenance overheads. Moreover, it employs free, off-the-shelf software and runs on low-cost PCs.

1. Introduction

During the 80's, networks were smaller, but, most importantly, problems were accidental rather than intentional. The tools that were developed then, assumed that the entire user population had a stake in the correct operation of the network and associated resources. Nowadays, we find ourselves having to consider hostile action as a possibility when trying to resolve problems. This is extremely distracting, since, even now, most problems are still accidental. Nevertheless, being able to eliminate malice as the cause of an incident is a big step towards its successful resolution.

Another aspect of our heritage is that most network elements (switches, routers, etc.) have not been designed with strong security in mind. Even in production networks one can still gain useful information using the SNMPv1 public community. [Stal95] More worrying is that many such devices assume that administration can be carried out via a password protected telnet connection, despite the fact that this method has been shown to be insecure. [Garf96]

In the academic community, where a strong security policy is difficult, if not impossible to enforce, these problems are more evident. However, even in private companies or organisations, the use of firewalls and other security mechanisms is mostly directed towards outside threats, while leaving the interior network elements vulnerable to internal attacks. [Chap95]

Another handicap, shared by organisations outside the U.S, is that it difficult to get strong security out of the box on products that originate from the U.S. due to the well known export restrictions of cryptographic mechanisms.

To make matters worse, even in cases where vendors offer security features and strong authentication for their products, these are usually applicable only to the product range of the particular vendor. Thus, the configuration and management of the security features is a constant headache for the management team. [Shah97]

2. Requirements

At the University of Piraeus we realised that there was no point in trying to impose a global security policy because its implementation via firewalls and router access lists would not be acceptable to the local community. An earlier attempt to provide restricted services to dial-up users had to be abandoned because of user dissent.

In the end, we decided to retreat to higher ground by creating smaller networks within the larger campus area network that offer increased security. For example, the network management workgroup LAN is screened behind a firewall while key departmental servers, as well as the University Internet server are also kept in similar secure networks. In order to monitor activity outside these networks we decided to investigate the feasibility of producing a

This project has been carried out within the EPEAEK series of programmes which are jointly funded by the Greek Ministry of Education and the European Community.

configuration for a personal computer (PC) that would enable us to create a vendor independent secure network monitoring station. These stations would be cheap and versatile so that they could be placed in various parts of our network and interfaced to different networking gear (see figure 1). The requirements for a machine of this class would be as follows:

- Low cost, preferably constructed from parts taken from decommissioned PCs.

- Minimal administrative overhead. This implied easy configuration and no administrator intervention after installation. Moreover, the bulk of the work for the construction of the software distribution for the network monitoring station should be devoted to integration of existing tools and packages, rather than the development of new code that would have to be maintained.

- Offer secure (encrypted) network connections with other similar stations and with the workstations of the network management staff.

- Be resistant to tampering. In the case where there are indications that the station has been hacked, its original configuration must be easily restored.

- Offer a standard platform for the execution of common network management and monitoring tools. It must also support the SNMP protocol.

- It must offer ways of establishing connections with network elements of various vendors for the purposes of administration and configuration.

- Finally, for troubleshooting purposes, it must be able to be deployed with minimal overheads in any part of the network.

In short, our intention was to construct something that could be used like meteorological balloons or sonar buoys: off-the-self and easily redeployable after use.

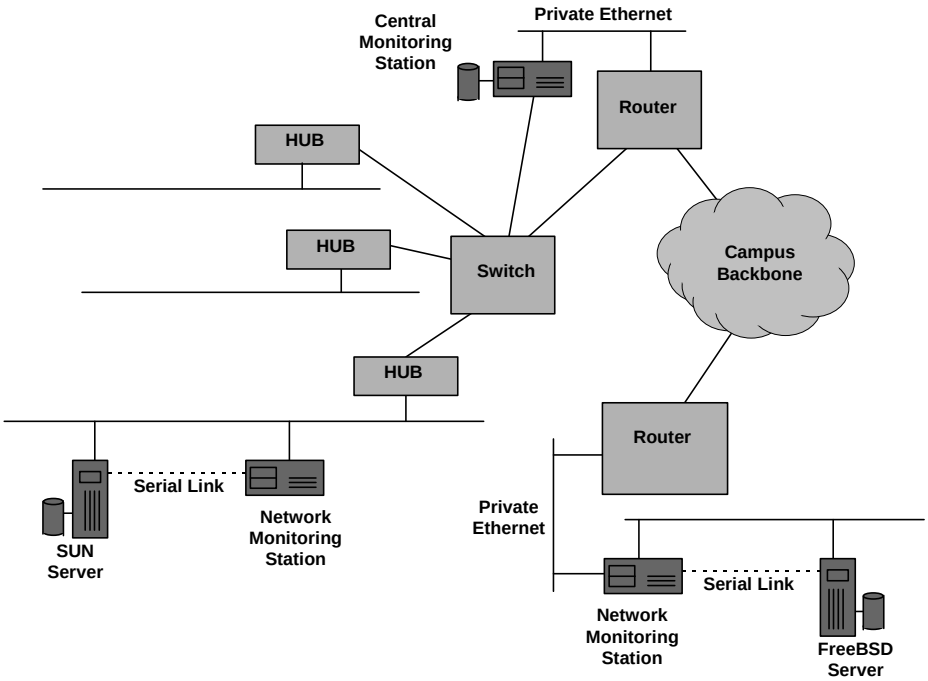

Figure 1: Network Monitoring Stations located in various parts of the network.

3. Network Monitoring Station

From the very beginning, the design team wanted a platform that could accommodate a large number of tools for network monitoring and management. The requirement that the station should operate in wiring closets without a monitor, keyboard or mouse effectively disqualified all Windows platforms. From the available UNIX or UNIX-like systems we eventually chose OpenBSD 2.3 for the following reasons:

- Like other free UNIX-clones, a large number of programs like tcpdump, snmpd, ssh, etc. are either supported in the base release or can be easily ported.

- The designers of OpenBSD have paid a lot of attention to the security profile of the system, creating a robust environment that is resistant to security related attacks (*http://www.openbsd.org/goals.html*)

- Most importantly, the system supports IPsec out of the box.

The next big decision that we took, was to dispense with a hard disk. The reason behind this decision was twofold, reliability and support. Older equipment, like the ones we use, tend to have problems with their hard drives, especially in the kind of hot environments that we let them operate. Greece is pretty hot in the summer and these PCs are left running in offices without air-conditioning for extended periods of time. Hard disks contribute a fair amount of heat and are also more prone to failure in these conditions.

The second and more important reason was related to the way that these machines were intended to be used. For our purposes, hard disks are already huge and are getting bigger all the time. This free space can cause all kinds of trouble; for example, it can be filled with data that should not be stored in the monitoring station in the first place. This means that stations can no longer be redeployed easily because this information must be backed up, or processed. Secondly, if a station is compromised, the intruders will be able to use this space as a bridgehead, transferring and installing tools that will enable them to attack other network assets.

On the other hand, diskless machines bring with them a whole collection of problems and administrative headaches. They are also basically incompatible with our intention of using standalone machines with encrypted tunnels for all communications between the monitoring stations.

Instead, we adopted the techniques used by the PICOBSD project which is a collection of FreeBSD configurations that can be accommodated within a single boot floppy (*http://www.freebsd.org/~picobsd*). The PICOBSD project provides configurations for a dial-up router, dial-in router (ISP access server), general purpose router and firewall. The PICOBSD technique links the code of all the executables that we wish to be available at runtime in a single executable using the *cruchgen* utility. [Silv98] The single executable alters its behaviour depending on the name under which it is run (argv[0]). By linking this executable to the names of the individual utilities we can create a fully functional /stand directory. The root of the runtime file system together with the executable and associated links are placed in a ramdisk that is stored within the kernel binary. The kernel is then compressed (using *gzip*) and placed on a bootable floppy. This floppy also contains the /etc directory of the running system in uncompressed form to allow easy configuration of the runtime parameters. At boot time, the kernel is copied from the floppy disk to main memory, uncompressed and executed. The file system root is then located in the ramdisk. The floppy disk is mounted and the /etc directory copied to the ramdisk. At this point the floppy is no longer needed and may be removed. The system is running entirely off the ramdisk and goes multi-user running the /etc/rc* scripts. Once the boot process is complete, user logins from the console or the network are allowed. The floppy is usually write-protected so changes in the system configuration do not survive reboots. However, there exists a utility that can copy the contents of the ramdisk /etc directory to the floppy, thus making the running configuration, permanent.

The aggregation of the system executables in a single file and the compression of the entire kernel allows a surprising number of facilities to be made available despite the small size of the boot medium. Some of these services are described below:

- **Accessing the system**
  The system establishes IPsec tunnels to a number of other secure hosts while shutting non-essential conventional IP ports through a packet filter (*ipf*). The initial configuration assumes statically assigned IP addresses and /etc/hosts table. Once the tunnels are running, a connection to the DNS server of the secure network may be established.

  Administrators on workstations that do not support IPsec (for example running Microsoft Windows 9x or NT) must rely on ssh for logging-on to the secure network hosts. This implies that the monitoring stations also run *sshd*. (*http://www.ssh.fi*). Over the IPsec links we run normal *telnet*.

  The Windows workstations run Teraterm which is a free telnet program that takes an ssh plug-in

(*http://www.zip.com.au/~roca/ttssh.html*). Additionally, there is a commercial version of ssh for these platforms (*http://www.ssh.com*).

- **Logging**
  Since the systems do not have any permanent storage, we have to send the system logs to the central monitoring station. This is a conventional (with disk) OpenBSD system that has IPsec links to all the other stations. The transfer of logging information to the central station is performed using *syslogd* over the IPsec links.

- **SNMP agent**
  Another way of controlling the system is via the SNMP protocol. The UCD *snmpd* program (*ftp://ftp.ece.ucdavis.edu:/pub/snmp/ucd-snmp.tar.gz*) through a custom MIB, supports remote execution of commands as well as a number of status monitoring facilities.

- **Remote console agent**
  This agent allows serial ports to be accessed from other stations connected to the secure network.

4. Examples of use

In this section we give two examples of the use of the system we have just described. The first example describes the way this system is used within the University network to manage routers and other network elements, while the second example describes the use of this system to handle the communication of network management teams of GUnet, a network linking Greek Universities.

4.1 Campus Network

The network monitoring stations have been deployed within the University of Piraeus network in three roles:

Controller: the monitoring station is connected directly to a router or other network device (e.g. switch, access server, etc.) so that configuration and administration of the device is carried out through the secure network. These connections can be serial links connecting one of the station built-in serial ports to the router console. Logins to the network device through its network ports are disabled so that administration can be carried out only via the console port. Hosts with serial consoles (e.g. SUNs) can also be controlled in this way.

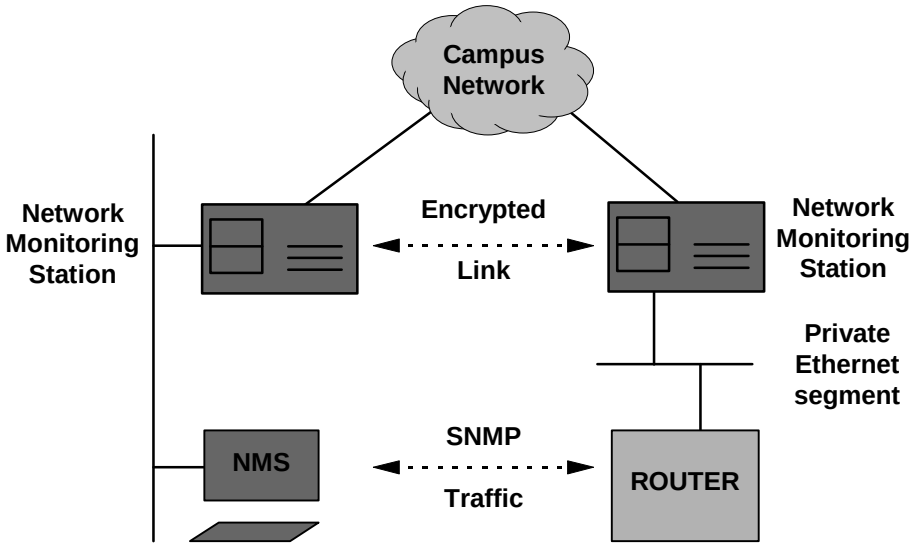

Figure 2: Patching external devices into the secure network

Some configurations contain so many tools that they simply cannot fit within the boundaries of a single floppy. In these cases we use a CDROM that contains the standard OpenBSD /usr partition and we either copy the utilities we need to the ramdisk or we leave the CDROM mounted on /usr.

An alternative means of connection is via back-to-back Ethernet links (see figure 2). In this case we dedicate an Ethernet port from the router and an

Ethernet port from the network monitoring station and link them via a crossed UTP cable. Via appropriate configuration of the router (e.g. ACLs) we can make sure that logins are allowed only from that Ethernet port. Similarly SNMP traffic may also be directed to that port. The Network Monitoring Station then relays the SNMP traffic through the IPsec links to the NMS workstation.

Traffic Monitor: We can monitor traffic on various LAN segments using *tcpdump* and send the output via *syslogd* to a central logging host. The syslog traffic goes over the IPsec links so that it cannot be intercepted (see figure 3)

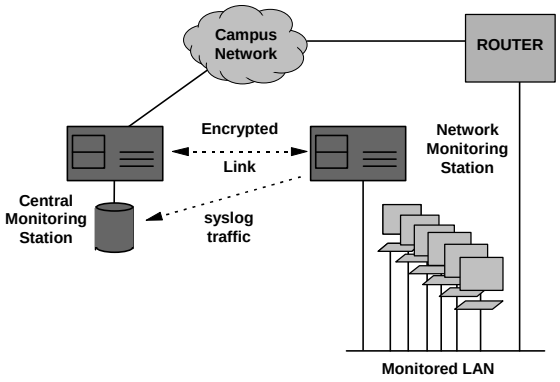

Figure 3: Monitoring remote parts of the LAN

Router: by adding high speed serial cards to the Network Monitoring Station we have created an emergency router for 2Mbps connections to buildings located outside the main campus and linked to the main building via leased lines (see figure 4). The OpenBSD kernel can support IP routing and through the addition of routing software (e.g. *gated*), it can exchange routing information (OSPF) with dedicated routers.

4.2 Monitoring stations in a WAN

Our second example concerns the Greek University Network (GUnet). In Greece, all public institutions of higher education have received funding in order to connect to a single high speed network (GUnet). Funding has also been provided for a local router, a server (in most cases a SUN workstation) and a network administrator stationed in each institution. These network administrators need to be in contact with the central network administration team in Athens.

Traditionally, these communications are carried out using PGP or some form of encrypted email scheme. However, for network monitoring purposes the administrators in the central site also need to be able to monitor the state of the routers located in the remote institutions. In cases of network problems, after consultation with the local network staff, the central site administrators may also need to run tests or reconfigure the GUnet routers and servers in the remote institutions. These operations would normally be run over insecure links, thus, creating opportunities for various exploits.

Due to the above considerations a small scale pilot

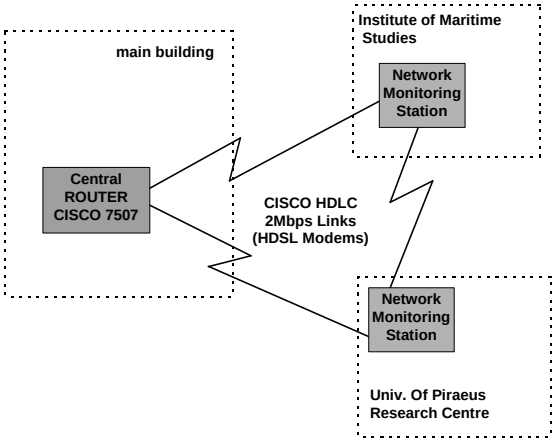

Figure 4: Network Monitoring Stations may act as routers

has been commissioned by GUnet to investigate the possibility of employing network monitoring, stations similar to the ones mentioned in the previous section, in all the GUnet sites. The pilot project involves the installation of a central monitoring station at the University of Piraeus and remote stations in four other institutions.

The configuration of the remote stations is displayed in figure 5. The central site is hosting the logging facility and also provides email and bulletin board services (for CERT and other security advisories) only to hosts within the secure network.

The serial links to the routers and hosts allow remote reconfiguration of the devices through a secure connection via the IPsec links. Also using the backup WAN links that most institutions have with different ISPs (not shown in the diagram), the secure network can be used to debug even the GUnet routers via their serial consoles.

The pilot project is expected to be complete by the end of March 1999, at which point a study will be carried out to determine the feasibility of creating a production system covering all institutions participating in the GUnet project. If all goes well, the full system will go on-line by the end of this year.

5. Evaluation

The system within the University of Piraeus has been in operation for more than six months and is used to control most of the network elements that under the administration of the network management centre. The GUnet pilot has been in operation for about two months.

During this time we have acquired significant experience with respect to the behaviour of the systems and the needs of the administrators who use them. In this section we will discuss some of these issues.

5.1 Securing the perimeter

Setting up a collection of IPsec tunnels does not automatically create a VPN. Each IPsec endpoint should also act as a firewall. Thus packet filters must be configured using the `ipf` utility and in some cases application-level gateways must also be installed.

This is by necessity a very precise task, but, at least, the resulting configuration files can be used on all the stations. You cannot, for example, simply turn off IP packets and allow only AH or ESP packets through, because this will shutdown the loopback interface as well. Also some services must go through (e.g. ssh, snmp, etc) and you may want to be able to ping the secure hosts from the outside, so they should respond to ICMP packets. Finally, what use would a monitoring station have, if it could not see the monitored hosts and these machines are by definition outside the secure network perimeter.

Moreover, IPsec configuration is also complex, since the connections are unidirectional, thus requiring, two secure associations for each pair of hosts.

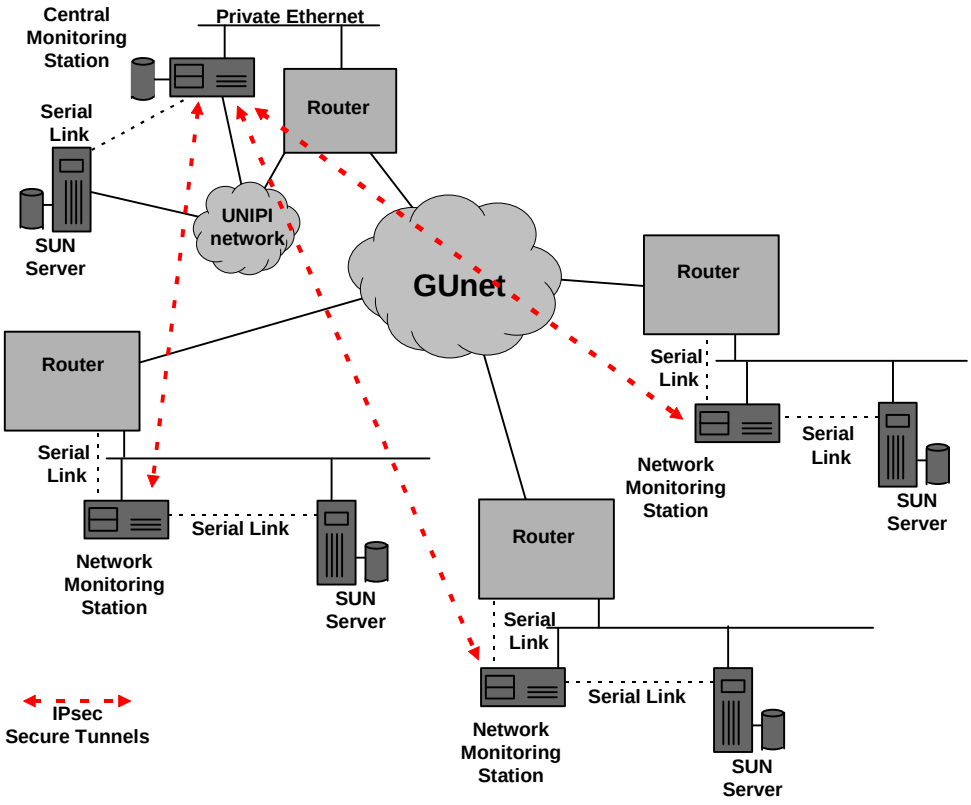

Figure 5: GUnet Network Monitoring Stations.

There are also a few pitfalls in configuring these tunnels. When we created the first few nodes we noticed that pinging one host resulted in the ICMP reply being sent in the clear (i.e. outside the IPsec link). This turned out to be a peculiarity of the IP stack whereby the system defers inserting the source address in a packet until the outgoing interface is determined. This way the packet gets the IP address of the correct network interface. The result of this is that the IPsec routing code needs an additional rule to catch such packets.

Using `tcpdump` to trace connections between two secure stations will often reveal holes either in the packet filter or the IPsec tunnels.

5.2 Automation

In the case of the GUnet pilot, we wanted to create a fully connected secure network, so that even if multiple nodes were disabled or locked out due to communication problems, the surviving nodes would still be able to communicate. The resulting mesh was likely to create a real mess because the number of tunnels exploded as the number of stations increased (2*n! to be exact).

Since it was clear that we couldn't possibly do this work manually, we wrote a set of shell scripts that created all the /etc configuration files from a single table containing information about each of the stations (e.g. name, IP address, netmask, etc.).

Each host was given a three digit number that was used for the generation of the secure associations. If hosts xxx and yyy wanted to set up a tunnel, then two SPI's would be assigned (xxxyyy and yyyxxx). Thus, the tunnels themselves could be easily identified and the scripts could generate the tunnel configurations automatically.

Similar scripts are used within the University of Piraeus network to create configurations for the diskette-based stations. In this case, all these stations are only connected to the main administration station which is also receiving their syslog records. Each subnet is given a colour code, so by typing `make blue`, we can create a standard floppy for the blue subnet.

5.3 Nostalgia

After entering the secure network the user is presented with an environment free from restrictions and checks. For example, the infamous r* utilities (`rsh, rwho, ruptime`, etc) all work without the need for specifying passwords. This is because we have deliberately made the assumption that the secure network should be considered as a logically contiguous secure region.

Obviously, this means that if an intruder gains access to a single host, then the entire network is compromised. This may be unacceptable in other environments, but in the case of our secure network, all the nodes are identical, so all nodes share the same vulnerabilities and hence a successful attack against one node will almost certainly succeed against any other node.

Nevertheless, the absence of internal barriers makes the network invisible to the user and thus improves the effectiveness of the administrators. For example, when was the last time you typed

```
tar cf - xxx |  rsh host tar xfp -
```

and it worked?

5.4 Key Management

No discussion on IPsec could be complete without mentioning key management. In our case we assign keys for the IPsec tunnels statically. We only use symmetric encryption (DES or 3DES) so we just create a file with random numbers and the scripts use these to generate the secure associations.

This creates the problem of distribution of the floppies since they contain the keys. The alternative scenario of assigning each institution a private key and using a key management protocol to create the session keys was considered in the initial stages of the design but was deemed to add too much complexity to the system. In any case, if the number of hosts increases so much that key generation and distribution becomes a problem, we can revise our original decision.

5.5 What we'd do differently

This title is often a euphemism for what we did wrong. In retrospect we spent way too much time with the floppy distribution. That was a mistake because floppies are a relic of the past and there are already

quite cost effective replacements offering capacities in excess of 100Mb. However running from a ramdisk is probably a good idea anyway, since it allows the use of read-only distribution media.

Another really big mistake was interfering with the OpenBSD startup files in /etc. Since we had a number of FreeBSD servers already running and we used PICOBSD (which is also based on FreeBSD), we decided to stick with the FreeBSD style startup files. Don't do it folks! There are some "minor" differences between the two systems that make porting these files a nightmare. Also, changing these files in a substantial way, creates a platform that cannot be easily upgraded to the next OpenBSD release since all this work will have to be redone. The worst is that, by that time, we'll have probably forgotten most of what we have learned in the first port so that recurring mistakes will bound to happen. All this without considering the wrath of the OpenBSD crowd. In retrospect, I would advise creating a list of the actions that will have to be performed on startup. This list can be drawn by examining the PICOBSD files. Then using this list we can modify the OpenBSD startup files creating diffs that can ease the next operating system upgrade.

6. Conclusions – Future Plans

Most of what the tools used in our project are well known and widely used. There are numerous network monitoring programs, and the notion of having a monitoring station installed in a LAN, has been proposed before [Ches94]. Also the use of IPsec, ssh and the other tools mentioned in this paper is common practice.

However, putting all this stuff on a floppy and deploying numerous stations both in our university network and in the institutions participating in the GUnet pilot is to our knowledge the first clear demonstration of such a network monitoring station.

It is difficult to stay still in an ever changing world, so we intend to keep maintaining the software distribution for the network monitoring station and adding features to make our lives easier. Taking advantage of the fact that the entire system is based on free software, we plan to make both bootable floppies and the development system that produces them available on our ftp site, so that other users may benefit from our efforts.

We are also looking into ways of abandoning the floppies as distribution media and replacing them with flash RAM cards because they offer higher capacity than floppies and, due to the complete lack of moving parts, are far more reliable.

Finally, as vendors slowly adopt the IPsec protocol, we hope to integrate equipment that support the protocol into our secure network. This will give us first hand experience with interoperability and integration between different implementations of the IPsec protocol.

References

[Chap95] Chapman D.Brent and Elizabeth D. Zwicky, "Building Internet Firewalls," Second Edition, O'Reilly & Associates, Inc. 1995.

[Ches94] Cheswick, William and Steven Bellovin, "Firewalls & Internet Security, Repelling the Wily Hacker," Addison-Wesley Professional Computing Series, 1994.

[Garf96] Garfinkel, Simpson and Gene Spafford, "Practical UNIX and Internet Security," Second Edition, O'Reilly & Associates, Inc. 1996.

[Reis93] Reiss, Levi and Joseph Radin, "Unix System Administration Guide," Osborne McGraw Hill Inc. 1993.

[Scot98] Scott, Charlie, Paul Wolfe and Mike Erwin, "Virtual Private Networks," O'Reilly & Associates, Inc. 1998.

[Shah97] Shah, Deval and Helen Holzbaur, "*Virtual Private Networks: Security With an Uncommon Touch*," Data Communications, Sept. 97,

[Silv98] da Silva, James, "Cruchgen," OpenBSD User Manual, 1998.

[Stal95] Stallings, William, "Network and Internetwork Security, Principles and Practice" Prentice Hall, 1995