Aufbau eines geschützten Subnetzes im TUNET

Werner Scholz, Thomas Schrefl und Josef Fidler
Institut für Angewandte und Technische Physik
werner.scholz@tuwien.ac.at, thomas.schrefl@tuwien.ac.at, fidler@tuwien.ac.at
http://magnet.atp.tuwien.ac.at/

Die Vergrößerung der Arbeitsgruppe und besondere Anforderungen an die Netzwerk-Infrastruktur haben uns veranlasst, ein geschütztes Subnetz innerhalb des TUNET aufzubauen. Über die dabei gemachten Erfahrungen soll im Folgenden berichtet werden.

1 Motivation

Im Lauf der letzten Jahre hat sich die Arbeitsgruppe von Prof. Fidler und Doz. Schrefl kontinuierlich vergrößert. Mittlerweile ist sie auf knapp ein Dutzend Mitarbeiter und rund 20 Computer (Arbeitsplatz-Rechner und Hochleistungs-Workstations) angewachsen.

Damit ist auch der Aufwand für Wartung und Administration gewachsen, ohne jedoch eine grundlegende Verbesserung z.B. auf dem Gebiet der Datensicherheit zu bringen. Die Umstellung auf twisted-pair-Verkabelung und die Erneuerung einiger Computer machte ein grundsätzlich neues Konzept notwendig.

Die wichtigsten Aspekte bei der Planung der neuen Netzwerk-Lösung waren Einbruchsicherheit, Datensicherheit und optimale Ausnützung der Netzwerk und Rechenkapazitäten bei vereinfachter Administration.

Einbruchsicherheit sollte vor Angriffen von Hackern aus dem Internet bewahren und damit vor den leidvollen Erfahrungen, die andere Institute bereits machen mussten. Datensicherheit bedeutet für uns nicht nur Schutz vor Datenverlust, sondern auch einen vereinfachten Zugriff auf die Daten innerhalb des heterogenen Netzwerks. Die vorhandenen und neu anzuschaffenden Rechner sollten dabei optimal ausgenützt werden und ein leistungsfähiges Umfeld zur Durchführung der mikromagnetischen Simulationen bereitstellen [1].

Wie sind die einzelnen Anforderungen zu erreichen? Wir wollen (und können) nicht alle Möglichkeiten aufzeigen und vergleichen, sondern beschränken uns auf die Beschreibung der Lösung, die wir für unsere Arbeitsgruppe gefunden haben.

Von Anfang an war klar, dass die Einbruchsicherheit nur durch eine Firewall-Lösung gewährleistet werden kann. Dazu ist es notwendig, das Netzwerk so aufzubauen, dass es genau eine Verbindung zwischen unserem Subnetz und dem Rest des TUNET gibt. Dazu kann man entweder ein „virtuelles LAN“ einrichten oder die Netze einfach physisch trennen. An die Schnittstelle der beiden Netze wird die Firewall gesetzt, die die Verbindung zwischen den beiden herstellt.

Datensicherheit wollten wir durch einen zentralen Fileserver erreichen. Über NFS können Unix und Linux Clients auf die Daten zugreifen, während Windows PCs über Samba [2] die zentralen Daten erreichen. Dadurch müssen sich die Benutzer nicht mehr mit „verteilten“ Daten auf einer Vielzahl von Rechnern herumschlagen, sondern haben alles an einem Ort vereint und können doch von allen Rechnern darauf zugreifen. Außerdem werden regelmäßige zentrale Backups möglich, die alle wichtigen Daten sichern.

Trotz der „bunten“ Mischung aus DEC Alpha Workstations, Linux/Alpha Workstations, Linux/Intel PCs und Windows PCs, wollten wir möglichst viele Verwaltungsaufgaben zentral auf einem Server erledigen und die Clients mit einer einfachen „Standardinstallation“ ins Netz einbinden. Vor allem die Benutzer- und Ressourcenverwaltung sollten zentral erfolgen. Für die Benutzerverwaltung bietet sich unter Unix/Linux NIS/yellow pages an, das eine zentrale Verwaltung ermöglicht und den Benutzern erlaubt, sich am nächsten freien Rechner anzumelden und die gewohnte Umgebung mit allen persönlichen Einstellungen vorzufinden. Ressourcenverwaltung bedeutet für uns die Verteilung der Computersimulationen auf die verschiedenen Rechner. Ein zentrales Queueing-System für Batch-Jobs sollte diese Anforderung auch in einer heterogenen Netzlandschaft erfüllen.

Letzteres ist auch das eigentliche Ziel unseres Entwurfs: Der Aufbau eines leistungsfähigen Netzwerks für CPU- und speicher-intensive Computersimulationen. Beim Entwurf hat uns Dr. Robert Lorenz (Institut für Materialphysik der Universität Wien) mit seiner langjährigen Erfahrung ausgezeichnet beraten. Die Firma init.at [3] hat die neuen Maschinen vorinstalliert geliefert und konfiguriert. Wartungsverträge sichern uns dabei den reibungslosen Betrieb und entlasten uns von Fehlersuche und Problembehebung. Das wichtigste Argument war (neben dem Preis) vor allem der Komfort, alles aus einer Hand zu erhalten, und damit bei Problemen nur einen Ansprechpartner zu haben.

2 Entwurf

Zuerst stellte sich die Frage, ob wir weiterhin die Infrastruktur des TUNET benützen oder unser Subnetz auch physisch vom TUNET trennen. Da wir eine NFS-Anbindung der Clients (auf denen die Computersimulationen durchgeführt werden) an einen zentralen Fileserver planten, war klar, dass die Verbindung der Rechner über die Switches des ZID mit 10 MBit nicht ausreichen würde. Vor allem das letzte Teilstück zum Fileserver (auch wenn es mit 100 MBit angebunden würde) könnte zum Flaschenhals werden. Daher haben wir uns entschlossen, die Rechner mit eigenen Switches zu verbinden und physisch vollkommen vom TUNET zu trennen. In Anbetracht der möglichen Netzwerkbelastung haben wir uns schließlich für eine 100 MBit full-duplex Anbindung der Clients und ein 1 GBit full-duplex Netzwerkinterface zum Fileserver entschieden.

Die physische Trennung vereinfacht auch den Einbau einer Firewall, da man ohne Einrichtung eines „virtuellen LANs“ auskommt. Für Firewalls bieten sich verschiedene Lösungen an. Eine einfache und ausfallssichere Firewall wird vom ZID angeboten [4]. Wir haben uns für eine Lösung von init.at entschieden und damit alle neuen Geräte aus einer Hand. Diese Firewall erlaubt auch den Aufbau eines „virtual private network (VPN)“ und damit die Einbindung entfernter Rechner (auch außerhalb des TUNET z. B. über chello StudentConnect, Teleweb), für die ein VLAN nicht mehr möglich ist (siehe Kap. 5.2). Außerdem verstecken wir durch IP-Masquerading unser gesamtes Subnetz hinter einer einzigen IP-Adresse. Dadurch haben wir hinter der Firewall die Freiheit, in einem Class-C-Netz bis zu 254 IP-Adressen selbst zu vergeben.

Der Fileserver ist das „Herzstück“ unseres Subnetzes. Er sollte mehrere Aufgaben übernehmen:

- Datenspeicherung, für die wir rund 200 GB vorgesehen haben. Der Zugriff auf die Daten muss über NFS und Samba ermöglicht werden.
- Benutzerverwaltung und Authentifizierung mittels NIS/ yellow pages.
- Nameserver für unser Subnetz
- Master für das Queueing-System
- zentrale Installation verschiedener Applikationen

Als Clients haben wir leistungsstarke PCs vorgesehen, die einerseits als Arbeitsplatzrechner dienen und gleichzeitig im Hintergrund die Computersimulationen durchführen. Dafür benötigen sie eine schnelle CPU und ausreichend Hauptspeicher. Als Betriebssystem haben wir uns für Linux entschieden, da es

- stabil genug ist, um eine gleichzeitige Nutzung der Computer als Arbeitsplatzrechner und „Rechenknecht“ zu ermöglichen,
- die gewünschte Client/Server-Architektur optimal unterstützt,
- von den von uns verwendeten Programmpaketen unterstützt wird,
- ein ausgezeichnetes Preis/Leistungsverhältnis hat
- und wir das Know-how haben, um die notwendigen Administrationsaufgaben selbst zu erledigen.

3 Implementierung

Bevor wir uns an die Umsetzung unserer Pläne machten, besprachen wir sie noch ausführlich mit dem ZID. Dabei wurde uns erlaubt, die bestehende twisted-pair-Verkabelung zu benützen und an unseren eigenen Switch, der in das Rack des ZID eingebaut wurde, anzuschließen.

3.1 Das Netzwerk

In Abbildung 1 ist die Struktur unseres Netzwerks skizziert. Die Firewall stellt die Verbindung zwischen dem TUNET und dem internen Subnetz über Switch 2 her. An Switch 2 sind einige Simulationsrechner (Linux/ Alphas) mit 100 MB und der Fileserver mit 1 GB angeschlossen. Eine zweite Glasfaserleitung stellt die Verbindung zu Switch 1 her, an den die bestehende twisted-pair Verkabelung mit den Arbeitsplatzrechnern angeschlossen ist. Der Mail- und Webserver ist außerhalb unseres Subnetzes direkt an das TUNET angeschlossen.

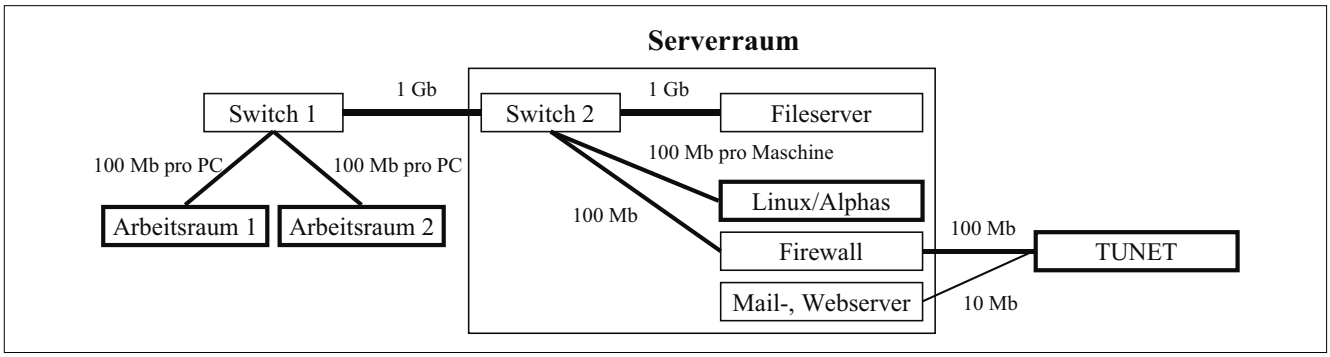

Abbildung 1

3.2 Die Switches

Da unsere Server in einem anderen Raum stehen, als das Rack, in dem die twisted-pair-Kabel der Arbeitsplätze zusammenlaufen, mussten wir zwei Switches (hp procurve 2512 und 2324) kaufen und mittels Glasfaserstrecke (1 GBit) verbinden. Die Anschlüsse der Arbeitsplätze wurden einfach von den Switches des ZID auf unseren verlegt, sodass wir hier eine „saubere“ physikalische Trennung des TUNET von unserem Subnetz erreichen. Solange das private Netz relativ klein und überschaubar ist, und damit keine spezielle Konfiguration (VLANs etc.) erfordert, kann auf Management- und Fernwartungsfähigkeiten (SNMP) der Switches verzichtet werden. Dem zweiten (zentralen) Switch, an den die Firewall und der Fileserver angeschlossen sind, haben wir diese Fähigkeiten spendiert, um einfachere Möglichkeiten zur Fehlerdiagnose zu haben. Unbedingt notwendig ist SNMP für uns, wie sich gezeigt hat, jedoch nicht.

3.3 Der Fileserver

Wir haben einen Dual Pentium III 733 MHz mit 256 MB ECC RAM ausgewählt, der unter SuSE Linux 7.0 [5] betrieben wird. Die geforderte Datensicherheit wird dabei durch mehrere Maßnahmen erreicht: Ein Hardware-RAID-Controller verbindet 4 SCSI Festplatten zu einem RAID Level 5 System mit einer Kapazität von 80 GB, auf dem das Betriebssystem und die Home-Verzeichnisse der Benutzer abgelegt sind. Der Einsatz des „journaling filesystem“ „ReiserFS“ [6] beschleunigt die Rekonstruktion des Filesystems in einen konsistenten Zustand auch bei Absturz des Fileservers. Zwei große IDE-Festplatten werden als Software-RAID mit einer Kapazität von 110 GB ebenfalls unter ReiserFS betrieben.

Zur Datensicherung haben wir einen Sony TDL-11000 Bandwechsler für 8 Backup-Bänder mit einem eingebauten SDL-11000 DDS4 Bandlaufwerk, das komprimiert bis zu 40 GB pro Band speichern kann. Ein Bandwechsler ist zwar eine teure aber sehr komfortable Investition, die durch vollautomatische tägliche Backups die Datensicherheit garantiert.

3.4 Die Clients

3.4.1 Reine Arbeitsplatzrechner

Die vorhandenen Arbeitsplatzrechner wurden natürlich an die neue Infrastruktur angeschlossen und profitieren von der schnellen 100 MB full-duplex Anbindung, sofern die jeweilige Netzwerkkarte das unterstützt.

3.4.2 Kombinierte Arbeitsplatz/Simulationsrechner

Fünf neue Athlon PCs mit 900 MHz und 512 MB RAM wurden als Arbeitsplatzrechner, die gleichzeitig im Hintergrund Simulationen durchführen, angeschafft und die Linux-Distribution von RedHat [7] in der Version 6.2 installiert. Schnelle 100 MB Netzwerkkarten und schnelle IDE-Festplatten, die im UDMA-Modus betrieben werden, erlauben diesen kombinierten Betrieb. Solange der Hauptspeicher ausreicht, ist kaum zu merken, dass die CPU im Hintergrund mit einer Simulation beschäftigt ist. Erst wenn auf den virtuellen Speicher auf der Festplatte ausgelagert wird, ist ein Performance-Einbruch spürbar.

3.4.3 Simulationsrechner

Sechs Alpha-Clones (Alpha EV56 Prozessoren mit 533 MHz auf AlphaPC 164 LX und SX Boards) wurden ebenfalls in unser Netzwerk eingebunden. Da wir mit RedHat 5.2 (Linux Kernel 2.0.25), das bisher installiert war, immer wieder Probleme mit NFS hatten, wurde ein Update auf RedHat 6.2 durchgeführt. Der Standard-Kernel wurde auch hier durch einen selbst kompilierten (2.2.18) ersetzt.

Außerdem wurde ein Hochleistungsrechner (UP2000: 2x Alpha EV67 666 MHz mit 1 GB ECC RAM auf einem AlphaPC 264 DP Board mit Tsunami Chipsatz [8]) gekauft, um besonders große Probleme rechnen zu können. Als Betriebssystem kommt wieder SuSE 7.0 mit selbstkompiliertem Kernel 2.2.17 SMP zum Einsatz.

4 Konfiguration und Administration

4.1 Netzwerkkonfiguration

Da wir so eine „saubere“ Trennung unseres Subnetzes vom TUNET vorgesehen hatten, wurden unserer Pläne vom ZID ohne Änderungen akzeptiert. Bei der Besprechung wurden im Wesentlichen nur die IP-Adresse der externen Netzwerkkarte unserer Firewall im TUNET und der Adressraum unseres Subnetzes festgelegt. Dabei wurde uns das Class-C-Netz 192.168.45.0/24 mit 254 freien IP-Adressen zugewiesen. Da auch Class-C-Adressen innerhalb des TUNET voll geroutet werden (!), ist eine entsprechende Koordinierung unbedingt notwendig. Dies ist vor allem im Falle eines „Lecks“ der Firewall, bei dem interne Pakte nach außen gelangen, wichtig, um den Urheber schnell ausfindig machen zu können.

Durch die Aktivierung der Firewall mit IP-Masquerading sind natürlich einige Rechner aus dem TUNET „verschwunden“, sodass wir sie in der TUNET-Datenbank [9] abmelden und die entsprechenden IP-Adressen freigeben konnten. In unserem Subnetz 128.130.45.0 wäre es gar nicht mehr möglich gewesen, alle neuen Rechner anzumelden, da der freie Adressraum bereits ausgeschöpft ist.

Will man sich die Administration von IP-Adressen ersparen, so kann man auch einen DHCP-Server installieren. Den Clients wird dann beim Booten automatisch eine IP-Adresse vom Server zugewiesen. Meist gibt es aber einige Geräte, die diese Funktionalität nicht unterstützen, und denen ohnedies eine IP-Adresse fix zugewiesen werden muss. Den wichtigen Servern werden auch meist fixe Adressen zugewiesen. Daher haben wir uns entschlossen, gleich alle IP-Adressen händisch zu vergeben.

Die Konfiguration der Firewall und des Fileservers wurde von init.at durchgeführt. Diese beiden Rechner übernehmen mehrere (zentrale) Funktionen in unserem Netzwerk und sollen daher im Folgenden näher beschrieben werden.

4.2 Firewall-Konfiguration

Die Firewall ist ein einfacher PC (Intel Celeron 633 MHz mit 64 MB RAM und 10 GB Festplatte), der mit einer von init.at modifizierten Linux-Distribution von Debian läuft. Eine einzige Datei enthält alle Informationen und Regeln zur Konfiguration der Firewall.

Die Firewall ist damit genauso sicher und unsicher wie der Linux-Kernel, der darauf läuft, aber sie ist für uns ein großer Fortschritt im Vergleich zur früheren Situation, als alle Rechner „direkt“ von der ganzen Welt aus erreichbar waren und damit ein potentielles Ziel für Angriffe aller Art.

Auch eine Sicherheitsüberprüfung durch das ZID [10], die die gängigsten Sicherheitsmängel aufdecken kann, hat uns diesbezüglich beruhigt.

Von außen ist (im Prinzip) ein einziger Port erreichbar, damit wir uns überhaupt in unser Netz einloggen können (siehe Kap. 5.1). Auch der Zugriff von innen nach außen ist auf die notwendigen Dienste (z. B. telnet, ftp, ssh, http, https, news, ntp, pop2, pop3, smtp, domain, ping, finger etc.) eingeschränkt.

Auf der Firewall laufen aus Sicherheitsgründen keine weiteren Dienste wie WWW-Proxies, E-Mail-Server, WWW-Server oder Ähnliches. Diese laufen alle auf anderen Maschinen. Öffentlich zugängliche Dienste wie WWW-Server oder E-Mail-Server kann man in einer „DMZ“ (demilitarized zone), die zwar durch die Firewall überwacht wird, in der aber bestimmte Dienste von außen zugänglich sind, unterbringen. Wir haben vorerst darauf verzichtet und den E-Mail- und WWW-Server der Arbeitsgruppe in die „freie Wildbahn“ gestellt, können aber bei Bedarf jederzeit eine DMZ einrichten.

4.3 Netzwerkoptimierung

Die Firewall erfüllt auch noch einen zweiten Zweck: Sie hält unerwünschten Netzwerkverkehr fern und entlastet damit die Anbindung unserer Arbeitsplatzrechner. Beispielsweise die „Fluten“ an Broadcast-Messages, die sich bei jedem Öffnen der Netzwerkumgebung auf einem Windows-PC ins Netz ergießen, werden von der Firewall gefiltert und nicht nach innen weitergeleitet. Natürlich sind unsere Windows-PCs damit von außen nicht mehr sichtbar, aber erstens liegen unsere wichtigen Daten ohnedies nur noch am Fileserver und zweitens soll auch sonst niemand auf unsere Windows-PCs zugreifen. (Wie wir von außen auf unsere Daten zugreifen können, ist in Kap. 5.1 beschrieben.)

Da wir mit unseren eigenen Switches alle Rechner mit 100 Mb full-duplex anschließen können, wurde bei jenen Rechnern, die das nicht automatisch mit dem Switch „ausgehandelt“ haben, händisch nachgeholfen. Sehr hilfreich waren dabei die LEDs auf unseren Switches, die die eingestellte Geschwindigkeit für jeden Port anzeigen. Natürlich haben wir auch ein paar ältere Modelle, die nur 10 MBit unterstützen, wobei diese alle im half-duplex Modus betrieben werden. Die Umstellung einer DEC AlphaStation in den 10 MBit full-duplex Modus, der theoretisch von unseren Switches unterstützt wird, war beispielsweise nicht möglich.

Schließlich haben wir auch den Netzwerk-Verkehr, der durch NFS-Verbindungen erzeugt wird, reduziert, indem wir „automount“ verwenden. Dazu mehr in Kapitel 4.4.3 über unsere NFS Konfiguration.

4.4 Fileserver-Konfiguration

Der Fileserver spielt eine zentrale Rolle in unserem Subnetz. Daher haben wir für ihn auch einen Wartungsvertrag abgeschlossen. Natürlich ist es gefährlich, einer einzigen Maschine alle im Folgenden beschriebenen Aufgaben zu übertragen, für uns hat es aber mehrere Vorteile. Erstens haben wir die Maschine fertig konfiguriert gekauft und mit dem Wartungsvertrag dem Lieferanten die Verantwortung für den reibungslosen Betrieb übertragen. Zweitens sind fast alle diese Dienste notwendig, um überhaupt in unserem Netz arbeiten zu können. Würden wir diese Dienste einer anderen Maschine übertragen, müssten wir für diese ebenfalls einen Wartungsvertrag abschließen. Dies kommt wieder nur bei einem neuen Gerät in Frage, was weitere Investitionen notwendig gemacht hätte. Schließlich war es unser Ziel, die Daten nur noch zentral am Fileserver abzulegen. Sollte dieser ausfallen, können wir ohnedies nicht auf unsere Daten zugreifen und sind in unserer Arbeit stark behindert, sodass er möglichst schnell wieder in Betrieb genommen werden muss.

4.4.1 NIS/yellow pages

Die zentrale Verwaltung der Benutzeraccounts erfolgt am Fileserver. Dieser stellt über NIS/yellow pages allen Clients die Benutzerdaten zur Verfügung. Ein neuer Benutzer muss damit nur ein Mal am Fileserver angelegt werden und kann sich sofort auf jedem beliebigen Unix/Linux-Rechner anmelden. Die home-Verzeichnisse liegen natürlich auch am Fileserver und werden über NFS exportiert, sodass jeder Benutzer seine gewohnte Umgebung und seine persönlichen Einstellungen vorfindet, egal an welchem Rechner er arbeitet. Dies funktioniert bei uns sogar im Mischbetrieb von Athlons und Alphas unter RedHat und SuSE Linux und einer DEC AlphaStation, die unter Compaq Tru64 4.0F läuft, reibungslos.

Wenn man viele Windows NT Rechner in einem heterogenen Netz mit Unix/Linux Rechnern zu administrieren hat, lohnt es sich sicher, diese in die zentrale Benutzerauthentifizierung über NIS/yellow pages einzubinden. Dies konnte bei uns unterbleiben und unter Windows 95 und Windows 98 kann sich ohnedies „jeder selbst“ seinen Benutzeraccount anlegen, wobei es jedoch vorteilhaft ist, die selben Usernamen zu verwenden (siehe Kap. 4.4.4).

4.4.2 DNS

Der Fileserver hat bei uns auch die Aufgaben eines Nameservers für die internen IP-Adressen übernommen. Dieser Dienst könnte noch durch einen zweiten Nameserver abgesichert werden. Diese Sicherheit bietet uns aber auch die Verteilung der IP-Adressen und Namen in statischen „hosts“-Dateien. Bei der Größe unseres Subnetzes

ist die Verwaltung dieser statischen Tabellen noch möglich und es reduziert nebenbei wieder den Netzwerkverkehr durch Einsparung von Anfragen an den Nameserver.

4.4.3 NFS

Die wichtigsten Verzeichnisse werden über NFS exportiert und so den Unix/Linux Clients zur Verfügung gestellt. Mit der 2.2.x Serie des Linux-Kernels wurde die NFS-Implementierung stark verbessert und hat sich als sehr stabil und zuverlässig erwiesen. Da NFS-Verbindungen auch dann (geringen) Netzwerkverkehr erzeugen, wenn keine Daten übertragen werden, wird auf allen Clients „automount" verwendet. Dabei laufen auf den Unix/Linux-Clients Dämonen im Hintergrund, die erst dann eine NFS-Verbindung herstellen (das gewünschte Verzeichnis mounten), wenn ein Zugriff darauf erfolgt. Nach einer (konfigurierbaren) Zeitspanne, in der keine Daten übertragen wurden, wird die Verbindung automatisch wieder abgebaut (umount des jeweiligen Verzeichnisses).

Auch die meisten Anwendungen haben wir nicht auf den Clients, sondern nur zentral am Fileserver installiert (siehe Kap. 4.4.6). Dies vereinfacht und verkürzt die Installation enorm.

NFS Dienste sind ein beliebtes Ziel für Angriffe. Unsere Firewall schützt uns vor derartigen Gefahren, da NFS-Verbindungen über die Firewall (in beiden Richtungen) unterbunden sind. Welche Probleme das (und die Verwendung von IP-Masquerading) mit sich bringt, ist in Kap. 5.6 beschrieben.

4.4.4 Samba

Damit auch Windows-PCs von der zentralen Speicherung der Daten profitieren, wurde am Fileserver ein Samba-Server installiert [2]. Bei der Einrichtung eines neuen Benutzers wird auch gleich ein Samba-User angelegt und ein entsprechendes Passwort festgelegt.

In unserem Netzwerk hat damit jeder Benutzer nur zwei Accounts und zwei Passworte, die aber problemlos ident gewählt werden können: Ein Unix/Linux-Account mit Passwort, mit dem er sich auf den Unix/Linux-Rechnern authentifiziert, und ein Benutzername mit entsprechendem Passwort für den Zugriff auf den Fileserver über Samba. Werden der Benutzername und das Passwort auf den Windows-Rechnern ident mit jenen für den Samba-Zugriff gewählt, erhält der Benutzer nach erfolgter Anmeldung an einem Windows-PC ohne weitere Passworteingabe Zugriff auf seine Daten am Fileserver. Damit die Samba-Passworte nicht im Klartext übertragen werden, muss unter Windows die Übertragung unverschlüsselter Passworte deaktiviert sein. Bei Windows 95 erfolgt dies mit einem Patch von Microsoft, bei Windows 98 und Windows NT 4.0 ab SP3 werden die Passworte standardmäßig nur verschlüsselt übertragen [11]. Letzteres kann durch einen Registry-Eintrag wieder aufgehoben werden, ist aber natürlich nicht zu empfehlen.

Für Windows-Rechner sind vier Samba-Shares eingerichtet. Unter [home] finden unsere Benutzer alle home-Verzeichnisse, von denen täglich ein Backup gezogen wird, wobei sie nur im eigenen Verzeichnis Schreibrechte haben. Unter [scr] erhält jeder Benutzer Schreib- und Lesezugriff auf seine Daten im Scratch-Bereich des Fileservers und Lesezugriff auf die Daten aller anderen. Zusätzlich wurde [groups] eingerichtet, das zwar physikalisch im Home-Directory des Fileservers liegt, jedoch mit speziellen Optionen in der Samba-Konfiguration versehen wurde. Diese führen dazu, dass die Dateien und Verzeichnisse, die unter Windows unter [groups] erstellt werden, von allen Benutzern gelesen und beschrieben/ modifiziert/gelöscht werden können. Damit wird unter Windows der gemeinsame Zugriff auf bestimmte Daten und beispielsweise das Erstellen und Bearbeiten von Dokumenten in der Gruppe vereinfacht. Schließlich gibt es noch einen [temp] Bereich (der physikalisch auf der Scratch-Partition am Fileserver liegt), in dem ebenfalls jeder alle Rechte hat.

4.4.5 DQS

Als Queueing-System, das die Jobs automatisch an die verfügbaren Rechner verteilt, verwenden wir DQS [12] in der Version 3.3.2. Wir haben in unserem Netz zwei binär-inkompatible Architekturen, nämlich Intel-kompatible Rechner (AMD Athlons) und Alphas. Diese haben wir trotzdem in einem Queueing-System zusammengefasst und verwalten dieses mit nur einem „Queue-Master". Dieser läuft am Fileserver und überwacht die einzelnen Queues und verteilt die Jobs.

DQS ermöglicht die Konfiguration mehrerer Zellen, die von mehreren Queue-Mastern verwaltet werden. Die Trennung der beiden Architekturen erfolgt aber einfacher durch die Definition von sog. „complexes". Diese sind im Wesentlichen Schlüsselworte, die bei verschiedenen Queues, die aus bestimmten Gründen zusammengefasst werden sollen, definiert sind. In unserem Fall wurde für die Intel-kompatiblen Rechner das Schlüsselwort „intel" definiert und den Queues der entsprechenden Rechner zugeordnet. Analog erhielten die sechs Queues der Alpha-Clones das Schlüsselwort „alpha". Zusätzlich wurden diese „complexes" als „required" konfiguriert, sodass der Benutzer beim Abschicken seines Jobs eines der beiden Schlüsselworte angeben muss. Damit ist der Benutzer gezwungen, sich für eine Architektur zu entscheiden, und es kann nicht vorkommen, dass ein Job unvorhergesehen auf der falschen Architektur (erfolglos) zur Ausführung kommt.

Spezielle Rechner, die für bestimmte Aufgaben reserviert sind, wurden nicht in obiges Schema aufgenommen, sondern wurden im „complex" „reserved" zusammengefasst. Dazu gehören unsere DEC AlphaStation, die hauptsächlich zum Kompilieren verwendet wird, die UP2000, die für besonders große Probleme verwendet wird, oder der Fileserver, auf dem keine Simulationen ausgeführt werden, sondern der z. B. große Datenmengen in einem Batch-Job komprimieren kann. Letzteres ist natürlich doppelt sinnvoll, da die Daten dann nicht zweimal über das Netz laufen, sondern direkt am Fileserver komprimiert werden.

4.4.6 Anwendungen

Alle Anwendungen, die zusätzlich zur RedHat-Standardinstallation benötigt werden, werden ebenfalls zentral am Fileserver installiert.

Unsere Strategie soll am Beispiel des wissenschaftlichen 2D-Plotprogramms Grace [13] demonstriert werden:

Nach dem Kompilieren des Quellcodes oder dem Auspacken aus dem tar.gz- oder RPM-Archiv wird auf der Partition „/pd“ des Fileservers ein Unterverzeichnis „grace-5.1.3“ angelegt. Darin werden die Unterverzeichnisse „bin“, „doc“, „include“, „lib“ etc. angelegt und die notwendigen Dateien in das entsprechende Unterverzeichnis kopiert. (Wenn man den Quellcode selbst kompiliert, kann man als Option für „configure“ oder im „Makefile“ das gewünschte Installationsverzeichnis meist frei wählen. In unserem Fall wäre dies „/pd/grace-5.1.3“.) Um die Installation späterer Updates zu erleichtern, wird noch ein symbolischer Link, der nur den Namen, aber keine Versionsnummer enthält, auf das aktuelle Installationsverzeichnis angelegt: „ln -s /pd/grace-5.1.3 /pd/grace“. Zuletzt wird die Anwendung in „/usr/local“ installiert, indem symbolische Links auf die entsprechenden Dateien angelegt werden: z. B. „ln -s /pd/grace/bin/grace /usr/local/bin/grace“.

Die Verzeichnisse „/pd“ und „/usr/local“ werden vom Fileserver mit NFS exportiert und von allen (Intel-kompatiblen) Clients gemountet. Die Umgebungsvariable „$PATH“ muss natürlich den Pfad „/usr/local/bin“ enthalten und der dynamische Linker für „shared libraries“ auch den Pfad „/usr/local/lib“ berücksichtigen (in „/etc/ld.so.conf“).

Die beschriebene Strategie hat mehrere Vorteile:

a) Anwendungen müssen nur ein Mal am Fileserver installiert werden und stehen danach sofort allen Clients zur Verfügung.

b) Saubere Trennung von Standardinstallation und selbst installierten Anwendungen.

c) Dadurch einfache Installation und Einbindung neuer Clients bzw. Neuinstallation oder Update des Betriebssystems der Clients, da nur eine Standardinstallation mit „ein bisschen“ Anpassung der Konfiguration notwendig ist.

d) Einfaches Update der Anwendungen durch Installation in ein neues Verzeichnis und Umsetzen des symbolischen Links (z. B. „/pd/grace -> /pd/grace-5.1.4“).

e) Schneller Überblick über alle installierten Anwendungen durch ein zentrales Verzeichnis („/pd“).

4.4.7 Beowulf Tools

Wie lässt sich nun die Konfiguration und Administration vieler gleichartiger Clients am einfachsten bewerkstelligen?

Im Rahmen des Beowulf-Projekts [14] wurden einige Werkzeuge entwickelt, die diese Aufgaben wesentlich vereinfachen. Wir verwenden vor allem „brsh“ [15], die „Beowulf remote shell“, die wir zu einer „bssh“, „Beowulf secure shell“, umgeschrieben haben. Damit lässt sich der Reihe nach auf allen gewünschten Maschinen ein Kommandozeilenbefehl ausführen, ohne sich auf jeder Maschine einzeln einzuloggen und den Befehl ausführen zu müssen.

Die Clients müssen auf diese Art des Zugriffs vorbereitet sein, indem eine entsprechende „.rlogin“-Datei im Home-Verzeichnis von root angelegt wird, oder der öffentliche Schlüssel des Benutzers, der bssh aufruft, den „authorized_keys“ der Clients hinzugefügt wird. „brsh“ ist nur ein einfaches Shell-Skript, das sich mittels rsh oder ssh auf jeder Maschine einloggt, den gewünschten Befehl ausführt und die Verbindung wieder beendet, ehe mit der nächsten Maschine fortgesetzt wird.

Die Konfiguration und Administration der Clients besteht meist in einem Editieren entsprechender Dateien. Von den Konfigurationsdateien (z. B. „/etc/bashrc“) haben wir beispielsweise eine Kopie auf dem Fileserver (z. B. „/pd/Athlon.config/etc/bashrc“). Ist nun eine Änderung notwendig, so wird die gewünschte Datei am Fileserver editiert und mit dem einfachen Befehl „bssh cp /pd/Athlon.config/etc/bashrc /etc/“ auf alle Clients kopiert.

Damit lässt sich ein Großteil der Administrationsaufgaben für alle Clients „gleichzeitig“ durchführen und der Aufwand ist unabhängig (!) von der Anzahl der Clients.

Natürlich ist die beschriebene Konfiguration nur in einem (von einer Firewall) geschützten Netzwerk empfehlenswert. Andernfalls wird es einem Angreifer sehr leicht gemacht, nach einer kompromittierten Maschine auch noch die Kontrolle über alle anderen zu übernehmen.

5 Probleme und Lösungen

5.1 Zugriff von außen

Hat man einmal so ein durch eine Firewall gesichertes Netzwerk aufgebaut, so war man bestrebt, alle Lücken, durch die ein Einbrecher in das Netzwerk eindringen kann, zu schließen. Hat man es wirklich gründlich gemacht, dann sollte man auch selbst nicht mehr von außen in sein Netzwerk hineingelangen. Da es aber meist doch erwünscht ist, authorisierten Personen den Zugang zu ermöglichen, muss man wohl oder übel wieder einen Weg öffnen.

Dafür gibt es wieder mehrere Möglichkeiten, die stark von den jeweiligen Erfordernissen abhängen. Auf keinen Fall sollte man ein Login direkt auf der Firewall erlauben. Die Firewall sollte vielmehr die Verbindung auf eine bestimmte Maschine im Netz weiterleiten (z. B. durch Port-Forwarding), die dann die Authentifizierung durchführt. Es ist damit sehr einfach und auch sehr empfehlenswert, die Anmeldungen im Netz zu protokollieren und zu überwachen.

In jedem Fall sollten nur verschlüsselte Verbindungen zugelassen werden, da sonst die Gefahr, dass Passworte belauscht (gesnifft) werden, zu groß ist. Heute ist das

Standardwerkzeug für verschlüsselte Verbindungen „ssh“ [16]. Im Gegensatz zu telnet, ftp und pop werden alle übertragenen Daten (vor allem auch die Passworte!) verschlüsselt. Damit wird die Übertragung vertraulicher Daten über ein unsicheres Netzwerk möglich. Natürlich muss die Verbindung vom Anfang bis zum Ende verschlüsselt sein. Die in der Praxis immer wieder verwendete Methode „*Ich-hab-auf-Rechner-A-gerade-kein-ssh-darum-log-ich-mich-auf-Rechner-B-mit-telnet-ein-und-mach-dann-ein-ssh-auf-Rechner-C*“ führt natürlich alle Sicherheitsmaßnahmen ad absurdum, da die Verbindung von Rechner A zu Rechner B nicht verschlüsselt ist und damit alle Informationen auf dieser Verbindungsstrecke belauscht werden können. Um solchen Situationen vorzubeugen, sind Werkzeuge wie Mindterm’s Java-Implementierung eines ssh Clients [17] hilfreich, da, auch wenn kein ssh installiert ist, fast immer ein Webbrowser mit Java-Unterstützung zur Verfügung steht.

Es gibt mehrere (auch freie) Implementierungen des ssh-Protokolls (z. B. OpenSSH [18]), und in den meisten Linux-Distributionen ist die eine oder andere enthalten.

Wie bereits erwähnt, ist auch ftp ein unsicheres Protokoll, bei dem Passworte im Klartext übertragen werden, und damit relativ einfach belauscht werden können. Als Ersatz bietet sich „scp“, „secure copy“, an, das Dateien über einen verschlüsselten Kanal kopieren kann. Es bietet die selbe Sicherheit und die selben Authentifizierungsmechanismen wie ssh und ist in den meisten ssh-Paketen enthalten.

Natürlich gibt es auch für Windows-Rechner entsprechende Programme. Wir verwenden als ssh-Client für Windows Putty [19] und als ftp-Ersatz WinSCP [20] oder den iXplorer [21]. Weitere Implementierungen auch für andere Plattformen findet man im WWW [22].

Idealerweise sollten für die Anmeldung von außen andere Benutzernamen und Passworte verwendet werden, als innerhalb des geschützten Netzes. Diese sollten vom Administrator vorgegeben und so gewählt sein, dass sie nicht leicht erraten werden können. Es sollten keine „normalen“ Wörter sein, sondern auch Ziffern und Sonderzeichen enthalten. Damit ist man beispielsweise vor „brute force“ Angriffen, die einfach ganze Wörterbücher durchprobieren, sicher. Eine regelmäßige Änderung der Passworte ist ein weiterer bedeutender Sicherheitsfaktor.

Will man trotzdem auch den unverschlüsselten Zugang ermöglichen, so bietet sich beispielsweise telnet mit (maschinengenerierten) Einmal-Passworten an. Jeder Benutzer kann dann eine Liste mit solchen Einmal-Passworten anfordern, die innerhalb eines bestimmten Zeitraums gültig sind und nur einmal verwendet werden können.

5.2 VPN

Ist ein einfacher ssh/telnet-Zugang von außen, wie im vorigen Kapitel beschrieben, nicht ausreichend, so kann durch „virtual private networking“ eine sichere, verschlüsselte Verbindung aufgebaut werden, die eine völlig transparente Einbindung in das lokale Subnetz ermöglicht. Beim Aufbau eines VPN wird eine verschlüsselte Punkt-zu-Punkt Verbindung hergestellt, die einen Client über ein unsicheres (öffentliches) Netzwerk (z. B. dem Internet) mit einem VPN-Server und dem angeschlossenen privaten Netz verbindet.

Wir verwenden VPN für Fernwartung und „Teleworking“. Im ersteren Fall erleichtert ein VPN unserem Lieferanten notwendige Konfigurations- und Administrationsarbeiten und damit die Erfüllung des Wartungsvertrags. Durch die Einrichtung eines VPN kann man auch von zu Hause über Modem-Dialin oder Teleweb in unserem Subnetz arbeiten, als wäre man direkt angeschlossen. Damit ist beispielsweise der direkte Zugriff auf unseren Fileserver möglich.

In welcher Form VPN unterstützt wird, hängt von der Firewall und dem VPN-Server ab, der verwendet wird. Will man Windows-Rechner über VPN einbinden, so kann man dazu den „Microsoft Windows VPN Adapter“ verwenden [23]. Abhängig von der Windows-Version werden verschiedene Verschlüsselungsalgorithmen mit verschiedenen Schlüssellängen unterstützt.

5.3 E-Mail

Sind die Standardprotokolle SMTP (zum Versenden) und POP bzw. IMAP (zum Abholen von E-Mails) auf der Firewall freigegeben, so kann man mit entsprechenden Client-Programmen ganz normal auf E-Mail-Server zugreifen. Probleme ergeben sich, wenn innerhalb des Subnetzes ein E-Mail-Server läuft, der E-Mails über die Firewall nach außen senden will. Dieser verwendet als Domain des Absenders nämlich die interne, die ja willkürlich gewählt wurde. Eine vom Benutzer „scholz“ am Fileserver weggeschickte E-Mail wird in unserem Netz mit dem Absender „scholz@fs.lan“ versehen. Die Domain des Absenders wird von den meisten E-Mail-Servern auf ihre Gültigkeit überprüft. Da es die Domain „lan“ natürlich nicht als offiziell registrierte Domain gibt, wird die E-Mail abgelehnt und kann nicht zugestellt werden. Dieses Problem kann durch Domain-Masquerading umgangen werden [24]. Dabei wird die Domain des Absenders ersetzt (sinnvollerweise durch den offiziellen E-Mail-Server der Benutzer), sodass die E-Mails erfolgreich zugestellt werden können.

5.4 WWW

Beim Zugriff auf das World Wide Web ergibt sich ein anderes Problem. Der Standard-Port für das http-Protokoll ist der Port 80. Ist dieser auf der Firewall geöffnet, so kann man problemlos auf das WWW zugreifen. Dokumente, die nur über eine verschlüsselte https-Verbindung erreicht werden können, erfordern das Öffnen des Port 443 für das https-Protokoll.

Zu allem Überfluss gibt es aber viele Dienste im WWW, die über andere Ports aufgerufen werden. Dazu zählen z. B. der Online-Katalog des Österreichischen Bibliothekenverbundes (Port 4505) [25], das „Hypertext Webster Gateway at UCSD“ (Port 5141) [26] und der Mirror von „Scientific Applications on Linux“ am Goodie Domain Service der TU Wien (Port 8050) [27]. Die entsprechenden Ports für die gewünschten Rechner ein-

zeln auf der Firewall freizugeben ist natürlich nicht praktikabel.

Eine elegante Lösung ist die Installation eines Proxy-Servers außerhalb des privaten Subnetzes. Für diesen muss nur ein bestimmter Port auf der Firewall geöffnet werden. Alle Anfragen des WWW-Browsers werden an den Proxy geschickt, der sie seinerseits an die gewünschte Maschine und an den gewünschten Port weiterleitet. Squid [28] ist so ein Proxy-Server, der auch noch als Cache fungieren kann, und die Protokolle http, https und ftp unterstützt.

Ist ein Proxy für das http-Protokoll ausreichend, so kann man auch den „webwasher“ [29] einsetzen. Dieser wurde eigentlich zum Filtern von Web-Inhalten entwickelt und kann mit oder ohne diese Funktionalität betrieben werden. Mit sehr feinen Einstellungsmöglichkeiten kann man definieren, welche Web-Inhalte nicht weitergeleitet werden sollen (z. B. Werbebanner, bestimmte URLs, Scripts, Animationen) bzw. welche Informationen an die Webserver weitergeleitet werden sollen („referers“, „cookies“).

5.5 Lizenzserver

Ein weiterer Dienst, der bei der Konfiguration der Firewall berücksichtigt werden muss, ist die Anforderung von Lizenzen bei Lizenzservern außerhalb des Subnetzes.

Einige Anwendungsprogramme (z. B. AVS, Patran), die im Rahmen von Campusverträgen vom ZID zur Verfügung gestellt werden, fragen die Lizenzen bei einem zentralen Lizenzserver im ZID ab. Damit diese Abfrage funktioniert, müssen die entsprechenden Ports auf der Firewall geöffnet werden. Dies ist kein Problem, solange die Kommunikation über fest vorgegebene Ports läuft. Der oft verwendete Lizenzmanager FLEXlm [30] erlaubt diese fixe Einstellung.

Trotzdem ist es uns passiert, dass AVS eines Tages nicht mehr laufen wollte, da es keine Lizenz mehr erhielt. Patran hingegen funktionierte weiter klaglos. Was war passiert? Eine Stromabschaltung im gesamten Freihaus der TU und damit auch im ZID erforderte die Abschaltung des Lizenzservers. Nach dem erneuten Hochfahren wurden auch die Lizenzserver mit den vorgegebenen Ports neu gestartet. AVS und Patran verwenden zur Abfrage der Lizenz aber nicht nur einen Port (der bei FLEXlm vorgegeben werden kann), sondern zwei. Der zweite hat sich bei Patran (zufällig?) nicht geändert, bei AVS aber sehr wohl, wie eine Analyse mit tcpdump [31] gezeigt hat. Nachdem der neue Port geöffnet war, konnte AVS auch wieder gestartet werden.

Auch wenn die Firewall sonst die problemloseste Maschine in unserem Subnetz ist, solche Schwierigkeiten sind nur mit einiger Erfahrung (zu der vielleicht auch dieser Artikel ein wenig beitragen kann) und mit Hilfe geeigneter Tools (wie tcpdump) zu lösen.

5.6 Campus- und Plattform-Software

Die Verteilung von Campus-Software erfolgt zumeist auf zwei Arten:

Windows-Software wird über den swd-Server verteilt, indem man unter Windows das entsprechende Netzlaufwerk verbindet und dann Zugriff auf sämtliche lizenzierte Software hat. Da beim Verbinden des Netzlaufwerks eine Benutzer-Authentifizierung mit Name und Passwort erforderlich ist, ist sichergestellt, dass nur autorisierte Personen Zugriff haben. Damit das funktioniert, müssen auf der Firewall einfach die entsprechenden Ports für SMB/NetBIOS-Verbindungen geöffnet werden.

Unix-Software wird meist durch NFS-Export der Verzeichnisse auf den entsprechenden Servern zur Verfügung gestellt. Da eine Benutzerauthentifizierung fehlt, muss genau festgelegt werden, welche Rechner die Verzeichnisse mounten dürfen. Da sich alle unsere Rechner mittels Masquerading hinter der Firewall verstecken, müsste als berechtigter Rechner die Firewall in den exports-Listen eingetragen werden. Damit kann aber nicht sichergestellt werden, dass nur berechtigte Maschinen die NFS-Verzeichnisse mounten. Damit ist es auch nicht mehr möglich, etwa Betriebssystemdokumentation auf den Installationsservern zu belassen und nur bei Bedarf mittels automount verfügbar zu machen. Da es für diese Problematik im Moment noch keine universelle Lösung gibt, ist bei jedem Zugriff auf die gewünschte Software Rücksprache mit den Verantwortlichen im ZID notwendig.

6 Zusammenfassung

Unser Netzwerk ist seit vier Monaten im Vollbetrieb und hat unsere Erwartungen bestens erfüllt. Die Firewall und der Fileserver laufen seit Beginn ohne Absturz und auch die Arbeitsplatzrechner verrichten trotz der Doppelbelastung (die vorhandene Rechnerkapazität wird mittlerweile voll genützt) sehr stabil ihren Dienst. Durch die zentrale Speicherung ist die Organisation der Daten stark vereinfacht worden und die regelmäßigen Backups haben die Gefahr von Datenverlust minimiert. Auch das dritte Ziel, die Einbruchsicherheit, haben wir durch die sehr restriktiv konfigurierte Firewall erreicht. Dabei ist es natürlich wichtig, regelmäßig die Sicherheitsbulletins zu lesen (z.TB. [32], [33]) und notwendige Patches einzuspielen. Trotz einer starken Erweiterung der Rechnerkapazität hat sich der Administrationsaufwand vereinfacht und ein ausgezeichnetes Umfeld für unsere wissenschaftliche Arbeit geschaffen.

7 Referenzen

[1] T. Schrefl, Finite elements in numerical micromagnetics I: Granular hard magnets. J. Magn. Magn. Mater., 207 (1999) 45-65.

T. Schrefl, Finite elements in numerical micromagnetics II: Patterned magnetic elements. J. Magn. Magn. Mater., 207 (1999) 66-77.

[2] samba - opening windows to a wider world
http://www.samba.org/
http://at.samba.org/samba/samba.html

[3] init.at informationstechnologie GmbH
http://www.init.at/

[4] W. Selos, „Eine einfache Firewall-Lösung“,
ZIDline Nr. 4, Dezember 2000
http://linux.tuwien.ac.at/Firewall.html

[5] SuSE Linux AG
http://www.suse.de/
Mirror: http://gd.tuwien.ac.at/linux/suse.com/

[6] Reiserfs
http://www.namesys.com/

[7] Red Hat, Inc.
http://www.redhat.com/
Mirror: http://gd.tuwien.ac.at/linux/redhat/

[8] API NetWorks Inc.
http://www.alpha-processor.com/
products/up2000-board.shtml

[9] TUNET-Datenbank
http://nic.tuwien.ac.at/tunetdb/

[10] ZID / DI Udo Linauer
http://www.zid.tuwien.ac.at/
mitteilungsblatt/mb02-2001.html#4

[11] Microsoft Support
http://support.microsoft.com/support/kb/articles/
q165/4/03.asp
http://support.microsoft.com/support/kb/articles/
Q166/7/30.asp

[12] DQS - Distributed Queueing System
http://www.scri.fsu.edu/~pasko/dqs.html

[13] Grace
http://plasma-gate.weizmann.ac.il/Grace/

[14] The Beowulf Project
http://www.beowulf.org/
http://buweb.parl.clemson.edu/software.html

[15] The Beowulf Underground
http://buweb.parl.clemson.edu/
http://buweb.parl.clemson.edu/software.html

[16] SSH Communications Security
http://www.ssh.fi/

[17] MindTerm - pure java ssh Client
http://www.mindbright.se/

[18] OpenSSH
http://www.openssh.org/

[19] PuTTY: A Free Win32 Telnet/SSH Client
http://www.chiark.greenend.org.uk/~sgtatham/
putty.html

[20] WinSCP - secure data transmission
http://winscp.vse.cz/

[21] Secure iXplorer - Windows Front End for PSCP
http://www.i-tree.org/ixplorer.htm

[22] SSH Clients für andere Betriebssysteme
http://www.openssh.org/windows.html
http://www.openssh.org/unix.html
http://www.at.openbsd.org/openssh/java.html
http://www.at.openbsd.org/openssh/palmos.html
http://opensores.thebunker.net/pub/mirrors/ssh-faq/
ssh-faq-2.html#ss2.1

[23] Virtual Private Networking
http://www.microsoft.com/technet/win2000/
win2ksrv/
reskit/intch09.asp

[24] sendmail.org - Masquerading and Relaying
http://www.sendmail.org/m4/masquerading.html

[25] Online-Katalog des Österreichischen Bibliothekenverbundes
http://bvzr.bibvb.ac.at:4505/ALEPH

[26] Hypertext Webster Gateway at UCSD
http://work.ucsd.edu:5141/cgi-bin/http_webster

[27] „Scientific Applications on Linux“ am Goodie Domain Service der TU Wien
http://gd.tuwien.ac.at:8050/

[28] Squid Web Proxy Cache
http://www.squid-cache.org/

[29] webwasher.com
http://www.webwasher.com/

[30] GLOBEtrotter Software
http://www.globetrotter.com/

[31] tcpdump
ftp://ftp.ee.lbl.gov/tcpdump.tar.Z

[32] CERT Coordination Center
http://www.cert.org/

[33] SANS Global Incident Analysis Center
http://www.sans.org/