```
Path Loss = 40 + 20log(5000) = 113 dB
```

Subtracting the path loss from the total gain

```
40 dB - 113 dB = -73 dB
```

Since -73 dB is greater than the minimum receive sensitivity of the client radio (-82 dBm), the signal level is just enough for the client radio to be able to hear the access point. There is only 9 dB of margin (82 dB - 73 dB) which will likely work fine in fair weather, but may not be enough to protect against extreme weather conditions.

Next we calculate the link from the client back to the access point:

```
  15 dBm (TX Power Radio 2)
+ 14 dBi (Antenna Gain Radio 2)
-  2 dB  (Cable Losses Radio 2)
+ 10 dBi (Antenna Gain Radio 1)
-  2 dB  (Cable Losses Radio 1)
————————
  35 dB = Total Gain
```

Obviously, the path loss is the same on the return trip. So our received signal level on the access point side is:

```
35 dB - 113 dB = -78 dB
```

Since the receive sensitivity of the AP is -89dBm, this leaves us 11dB of fade margin (89dB - 78dB). Overall, this link will probably work but could use a bit more gain. By using a 24dBi dish on the client side rather than a 14dBi sectorial antenna, you will get an additional 10dBi of gain on both directions of the link (remember, antenna gain is reciprocal). A more expensive option would be to use higher power radios on both ends of the link, but note that adding an amplifier or higher powered card to one end generally does not help the overall quality of the link.

Online tools can be used to calculate the link budget. For example, the Green Bay Professional Packet Radio's Wireless Network Link Analysis (*http://my.athenet.net/~multiplx/cgi-bin/wireless.main.cgi*) is an excellent tool. The Super Edition generates a PDF file containing the Fresnel zone and radio path graphs. The calculation scripts can even be downloaded from the website and installed locally.

The Terabeam website also has excellent calculators available online (*http://www.terabeam.com/support/calculations/index.php*).

Tables for calculating link budget

To calculate the link budget, simply approximate your link distance, then fill in the following tables:

Free Space Path Loss at 2.4 GHz

Distance (m)	100	500	1,000	3,000	5,000	10,000
Loss (dB)	80	94	100	110	113	120

For more path loss distances, see **Appendix C**.

Antenna Gain:

Radio 1 Antenna	+ Radio 2 Antenna	= Total Antenna Gain

Losses:

Radio 1 + Cable Loss (dB)	Radio 2 + Cable Loss (dB)	Free Space Path Loss (dB)	= Total Loss (dB)

Link Budget for Radio 1 → Radio 2:

Radio 1 TX Power	+ Antenna Gain	- Total Loss	= Signal	> Radio 2 Sensitivity

Link Budget for Radio 2 ➔ Radio 1:

Radio 2 TX Power	+ Antenna Gain	- Total Loss	= Signal	> Radio 1 Sensitivity

If the received signal is greater than the minimum received signal strength in both directions of the link, as well as any noise received along the path, then the link is possible.

Link planning software

While calculating a link budget by hand is straightforward, there are a number of tools available that will help automate the process. In addition to calculating free space loss, these tools will take many other relevant factors into account as well (such as tree absorption, terrain effects, climate, and even estimating path loss in urban areas). In this section, we will discuss two free tools that are useful for planning wireless links: Green Bay Professional Packet Radio's online interactive network design utilities, and RadioMobile.

Interactive design CGIs

The Green Bay Professional Packet Radio group (GBPRR) has made a variety of very useful link planning tools available for free online. You can browse these tools online at *http://www.qsl.net/n9zia/wireless/page09.html .* Since the tools are available online, they will work with any device that has a web browser and Internet access.

We will look at the first tool, **Wireless Network Link Analysis**, in detail. You can find it online at *http://my.athenet.net/~multiplx/cgi-bin/wireless.main.cgi.*

To begin, enter the channel to be used on the link. This can be specified in MHz or GHz. If you don't know the frequency, consult the table in **Appendix B**. Note that the table lists the channel's center frequency, while the tool asks for the highest transmitted frequency. The difference in the ultimate result is minimal, so feel free to use the center frequency instead. To find the highest transmitted frequency for a channel, just add 11MHz to the center frequency.

Next, enter the details for the transmitter side of the link, including the transmission line type, antenna gain, and other details. Try to fill in as much data as you know or can estimate. You can also enter the antenna height and elevation for this site. This data will be used for calculating the antenna tilt

angle. For calculating Fresnel zone clearance, you will need to use GBPRR's Fresnel Zone Calculator.

The next section is very similar, but includes information about the other end of the link. Enter all available data in the appropriate fields.

Finally, the last section describes the climate, terrain, and distance of the link. Enter as much data as you know or can estimate. Link distance can be calculated by specifying the latitude and longitude of both sites, or entered by hand.

Now, click the Submit button for a detailed report about the proposed link. This includes all of the data entered, as well as the projected path loss, error rates, and uptime. These numbers are all completely theoretical, but will give you a rough idea of the feasibility of the link. By adjusting values on the form, you can play "what-if?" to see how changing various parameters will affect the connection.

In addition to the basic link analysis tool, GBPRR provides a "super edition" that will produce a PDF report, as well as a number of other very useful tools (including the Fresnel Zone Calculator, Distance & Bearing Calculator, and Decibel Conversion Calculator to name just a few). Source code to most of the tools is provided as well.

RadioMobile

Radio Mobile is a tool for the design and simulation of wireless systems. It predicts the performance of a radio link by using information about the equipment and a digital map of the area. It is public domain software that runs on Windows, or using Linux and the Wine emulator.

Radio Mobile uses a ***digital terrain elevation model*** for the calculation of coverage, indicating received signal strength at various points along the path. It automatically builds a profile between two points in the digital map showing the coverage area and first Fresnel zone. During the simulation, it checks for line of sight and calculates the Path Loss, including losses due to obstacles. It is possible to create networks of different topologies, including net master/slave, point-to-point, and point-to-multipoint. The software calculates the coverage area from the base station in a point-to-multipoint system. It works for systems having frequencies from 100 kHz to 200 GHz. ***Digital elevation maps*** (***DEM***) are available for free from several sources, and are available for most of the world. DEMs do not show coastlines or other readily identifiable landmarks, but they can easily be combined with other kinds of data (such as aerial photos or topographical charts) in several layers to obtain a more useful and readily recognizable representation. You can digitize your own maps and combine them with DEMs. The digital elevation maps can be merged with

scanned maps, satellite photos and Internet map services (such as Google Maps) to produce accurate prediction plots.

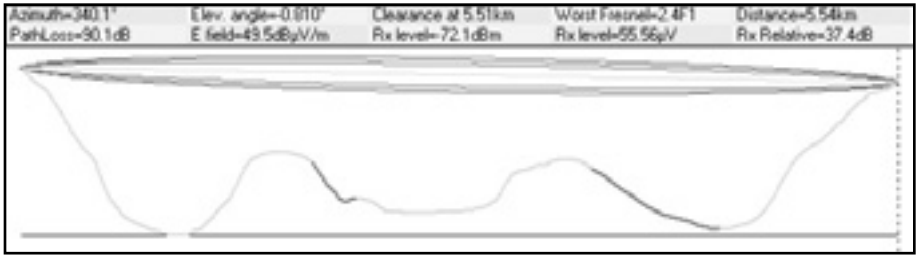

Figure 3.20: Link feasibility, including Fresnel zone and line of sight estimate, using RadioMobile.

The main Radio Mobile webpage, with examples and tutorials, is available at: *http://www.cplus.org/rmw/english1.html*

RadioMobile under Linux

Radio Mobile will also work using Wine under Ubuntu Linux. While the application runs, some button labels may run beyond the frame of the button and can be hard to read.

We were able to make Radio Mobile work with Linux using the following environment:

- IBM Thinkpad x31
- Ubuntu Breezy (v5.10), *http://www.ubuntu.com/*
- Wine version 20050725, from the Ubuntu Universe repository

There are detailed instructions for installing RadioMobile on Windows at *http://www.cplus.org/rmw/english1.html*. You should follow all of the steps except for step 1 (since it is difficult to extract a DLL from the VBRUN60SP6.EXE file under Linux). You will either need to copy the MSVBVM60.DLL file from a Windows machine that already has the Visual Basic 6 run-time environment installed, or simply Google for MSVBVM60.DLL, and download the file.

Now continue with step 2 at from the above URL, making sure to unzip the downloaded files in the same directory into which you have placed the downloaded DLL file. Note that you don't have to worry about the stuff after step 4; these are extra steps only needed for Windows users.

Finally, you can start Wine from a terminal with the command:

```
# wine RMWDLX.exe
```

You should see RadioMobile running happily in your XWindows session.

Avoiding noise

The unlicensed ISM and U-NII bands represent a very tiny piece of the known electromagnetic spectrum. Since this region can be utilized without paying license fees, many consumer devices use it for a wide range of applications. Cordless phones, analog video senders, Bluetooth, baby monitors, and even microwave ovens compete with wireless data networks for use of the very limited 2.4 GHz band. These signals, as well as other local wireless networks, can cause significant problems for long range wireless links. Here are some steps you can use to reduce reception of unwanted signals.

- **Increase antenna gain on both sides of a point-to-point link.** Antennas not only add gain to a link, but their increased directionality tends to reject noise from areas around the link. Two high gain dishes that are pointed at each other will reject noise from directions that are outside the path of the link. Using omnidirectional antennas will receive noise from all directions.

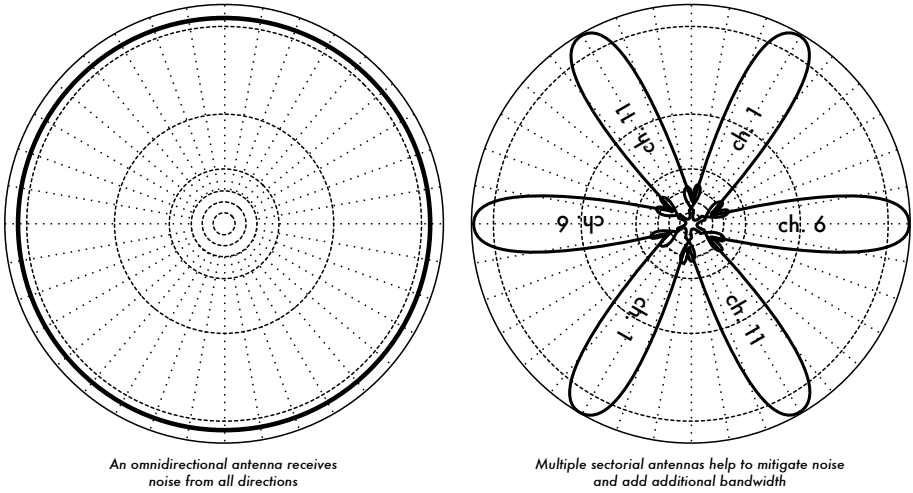

Figure 3.21: A single omnidirectional antenna vs. multiple sectorials.

- **Use sectorials instead of using an omnidirectional.** By making use of several sectorial antennas, you can reduce the overall noise received at a distribution point. By staggering the channels used on each sectorial, you can also increase the available bandwidth to your clients.

- **Don't use an amplifier.** As we will see in **Chapter 4**, amplifiers can make interference issues worse by indiscriminately amplifying all received signals, including sources of interference. Amplifiers also cause interference problems for other nearby users of the band.
- **Use the best available channel.** Remember that 802.11b/g channels are 22 MHz wide, but are only separated by 5MHz. Perform a site survey, and select a channel that is as far as possible from existing sources of interference. Remember that the wireless landscape can change at any time as people add new devices (cordless phones, other networks, etc.) If your link suddenly has trouble sending packets, you may need to perform another site survey and pick a different channel.
- **Use smaller hops and repeaters, rather than a single long distance shot.** Keep your point-to-point links as short as possible. While it may be possible to create a 12 km link that cuts across the middle of a city, you will likely have all kinds of interference problems. If you can break that link into two or three shorter hops, the link will likely be more stable. Obviously this isn't possible on long distance rural links where power and mounting structures are unavailable, but noise problems are also unlikely in those settings.
- **If possible, use 5.8 GHz, 900MHz, or another unlicensed band**. While this is only a short term solution, there is currently far more consumer equipment installed in the field that uses 2.4 GHz. Using 802.11a or a 2.4 GHz to 5.8 GHz step-up device will let you avoid this congestion altogether. If you can find it, some old 802.11 equipment uses unlicensed spectrum at 900MHz (unfortunately at much lower bit rates). Other technologies, such as Ronja (*http://ronja.twibright.com/*) use optical technology for short distance, noise-free links.
- **If all else fails, use licensed spectrum**. There are places where all available unlicensed spectrum is effectively used. In these cases, it may make sense to spend the additional money for proprietary equipment that uses a less congested band. For long distance point-to-point links that require very high throughput and maximum uptime, this is certainly an option. Of course, these features come at a much higher price tag compared to unlicensed equipment.

To identify sources of noise, you need tools that will show you what is happening in the air at 2.4 GHz. We will see some examples of these tools in **Chapter 6**.

Repeaters

The most critical component to building long distance network links is ***line of sight*** (often abbreviated as ***LOS***). Terrestrial microwave systems simply cannot tolerate large hills, trees, or other obstacles in the path of a long distance link. You must have a clear idea of the lay of the land between two points before you can determine if a link is even possible.

But even if there is a mountain between two points, remember that obstacles can sometimes be turned into assets. Mountains may block your signal, but assuming power can be provided they also make very good ***repeater*** sites.

Repeaters are nodes that are configured to rebroadcast traffic that is not destined for the node itself. In a mesh network, every node is a repeater. In a traditional infrastructure network, nodes must be configured to pass along traffic to other nodes.

A repeater can use one or more wireless devices. When using a single radio (called a ***one-arm repeater***), overall efficiency is slightly less than half of the available bandwidth, since the radio can either send or receive data, but never both at once. These devices are cheaper, simpler, and have lower power requirements. A repeater with two (or more) radio cards can operate all radios at full capacity, as long as they are each configured to use non-overlapping channels. Of course, repeaters can also supply an Ethernet connection to provide local connectivity.

Repeaters can be purchased as a complete hardware solution, or easily assembled by connecting two or more wireless nodes together with Ethernet cable. When planning to use a repeater built with 802.11 technology, remember that nodes must be configured for master, managed, or ad-hoc mode. Typically, both radios in a repeater are configured for master mode, to allow multiple clients to connect to either side of the repeater. But depending on your network layout, one or more devices may need to use ad-hoc or even client mode.

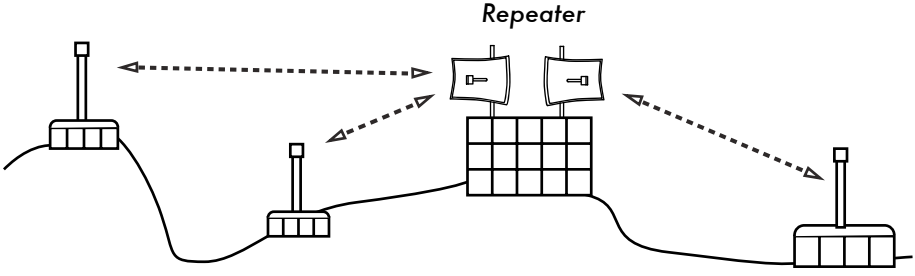

Figure 3.22: The repeater forwards packets over the air between nodes that have no direct line of sight.

Typically, repeaters are used to overcome obstacles in the path of a long distance link. For example, there may be buildings in your path, but those buildings contain people. Arrangements can often be worked out with building owners to provide bandwidth in exchange for roof rights and electricity. If the building owner isn't interested, tenants on high floors may be able to be persuaded to install equipment in a window.

If you can't go over or through an obstacle, you can often go around it. Rather than using a direct link, try a multi-hop approach to avoid the obstacle.

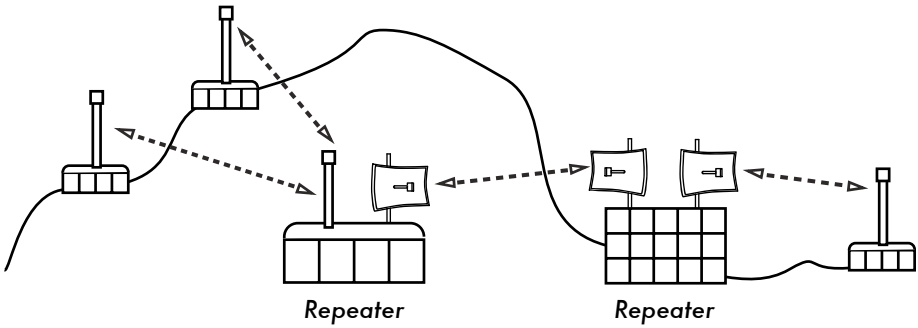

Figure 3.23: No power was available at the top of the hill, but it was circumvented by using multiple repeater sites around the base.

Finally, you may need to consider going backwards in order to go forwards. If there is a high site available in a different direction, and that site can see beyond the obstacle, a stable link can be made via an indirect route.

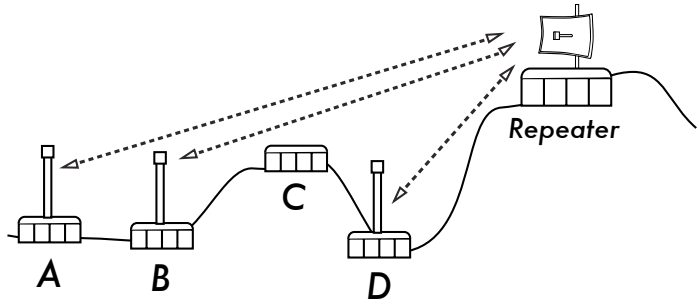

Figure 3.24: Site D could not make a clean link to site A or B, since site C is in the way and is not hosting a node. By installing a high repeater, nodes A, B, and D can communicate with each other. Note that traffic from node D actually travels further away from the rest of the network before the repeater forwards it along.

Repeaters in networks remind me of the "six degrees of separation" principle. This idea says that no matter who you are looking for, you need only contact five intermediaries before finding the person. Repeaters in high places can "see" a great deal of intermediaries, and as long as your node is in range of the repeater, you can communicate with any node the repeater can reach.

Traffic optimization

Bandwidth is measured as the amount of bits transmitted over a time interval. This means that over time, bandwidth available on any link approaches infinity. Unfortunately, for any given period of time, the bandwidth provided by any given network connection is not infinite. You can always download (or upload) as much traffic as you like; you need only wait long enough. Of course, human users are not as patient as computers, and are not willing to

wait an infinite amount of time for their information to traverse the network. For this reason, bandwidth must be managed and prioritized much like any other limited resource.

You will significantly improve response time and maximize available throughput by eliminating unwanted and redundant traffic from your network. This section describes a few common techniques for making sure that your network carries only the traffic that must traverse it. For a more thorough discussion of the complex subject of bandwidth optimization, see the free book *How to Accelerate Your Internet* (*http://bwmo.net/*).

Web caching

A web proxy server is a server on the local network that keeps copies of recently retrieved or often used web pages, or parts of pages. When the next person retrieves these pages, they are served from the local proxy server instead of from the Internet. This results in significantly faster web access in most cases, while reducing overall Internet bandwidth usage. When a proxy server is implemented, the administrator should also be aware that some pages are not cacheable-- for example, pages that are the output of server-side scripts, or other dynamically generated content.

The apparent loading of web pages is also affected. With a slow Internet link, a typical page begins to load slowly, first showing some text and then displaying the graphics one by one. In a network with a proxy server, there could be a delay when nothing seems to happen, and then the page will load almost at once. This happens because the information is sent to the computer so quickly that it spends a perceptible amount of time rendering the page. The overall time it takes to load the whole page might take only ten seconds (whereas without a proxy server, it may take 30 seconds to load the page gradually). But unless this is explained to some impatient users, they may say the proxy server has made things slower. It is usually the task of the network administrator to deal with user perception issues like these.

Proxy server products

There are a number of web proxy servers available. These are the most commonly used software packages:

- **Squid**. Open source Squid is the de facto standard at universities. It is free, reliable, easy to use and can be enhanced (for example, adding content filtering and advertisement blocking). Squid produces logs that can be analyzed using software such as Awstats, or Webalizer, both of which are open source and produce good graphical reports. In most cases, it is easier to install as part of the distribution than to download it from

http://www.squid-cache.org/ (most Linux distributions such as Debian, as well as other versions of Unix such as NetBSD and FreeBSD come with Squid). A good Squid configuration guide can be found on the Squid Users Guide Wiki at *http://www.deckle.co.za/squid-users-guide/*.

- **Microsoft Proxy server 2.0.** Not available for new installations because it has been superseded by Microsoft ISA server and is no longer supported. It is nonetheless used by some institutions, although it should perhaps not be considered for new installations.
- **Microsoft ISA server.** ISA server is a very good proxy server program, that is arguably too expensive for what it does. However, with academic discounts it may be affordable to some institutions. It produces its own graphical reports, but its log files can also be analyzed with popular analyzer software such as Sawmill (*http://www.sawmill.net/*). Administrators at a site with MS ISA Server should spend sufficient time getting the configuration right; otherwise MS ISA Server can itself be a considerable bandwidth user. For example, a default installation can easily consume more bandwidth than the site has used before, because popular pages with short expiry dates (such as news sites) are continually being refreshed. Therefore it is important to get the pre-fetching settings right, and to configure pre-fetching to take place mainly overnight. ISA Server can also be tied to content filtering products such as WebSense. For more information, see: *http://www.microsoft.com/isaserver/* and *http://www.isaserver.org/* .

Preventing users from bypassing the proxy server

While circumventing Internet censorship and restrictive information access policy may be a laudable political effort, proxies and firewalls are necessary tools in areas with extremely limited bandwidth. Without them, the stability and usability of the network are threatened by legitimate users themselves. Techniques for bypassing a proxy server can be found at *http://www.antiproxy.com/* . This site is useful for administrators to see how their network measures up against these techniques.

To enforce use of the caching proxy, you might consider simply setting up a network access policy and trusting your users. In the layout below, the administrator has to trust that his users will not bypass the proxy server.

In this case the administrator typically uses one of the following techniques:

- **Not giving out the default gateway address through DCHP.** This may work for a while, but some network-savvy users who want to bypass the proxy might find or guess the default gateway address. Once that happens, word tends to spread about how to bypass the proxy.

- **Using domain or group policies.** This is very useful for configuring the correct proxy server settings for Internet Explorer on all computers in the domain, but is not very useful for preventing the proxy from being bypassed, because it depends on a user logging on to the NT domain. A user with a Windows 95/98/ME computer can cancel his log-on and then bypass the proxy, and someone who knows a local user password on his Windows NT/2000/XP computer can log on locally and do the same.
- **Begging and fighting with users.** This approach, while common, is never an optimal situation for a network administrator.

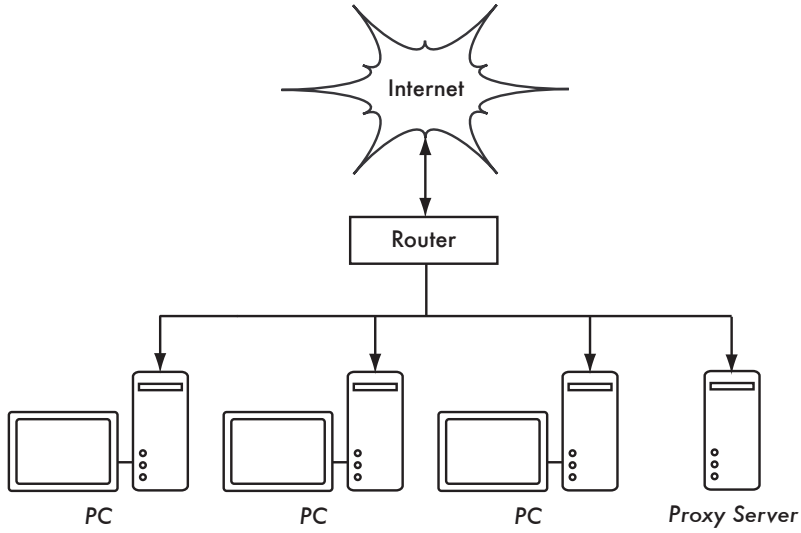

Figure 3.25: This network relies on trusted users to properly configure their PCs to use the proxy server.

The only way to ensure that proxies cannot be bypassed is by using the correct network layout, by using one of the three techniques described below.

Firewall

A more reliable way to ensure that PCs don't bypass the proxy can be implemented using the firewall. The firewall can be configured to allow only the proxy server to make HTTP requests to the Internet. All other PCs are blocked, as shown in **Figure 3.26**.

Relying on a firewall may or may not be sufficient, depending on how the firewall is configured. If it only blocks access from the campus LAN to port 80 on web servers, there will be ways for clever users to find ways around it. Additionally, they will be able to use other bandwidth hungry protocols such as BitTorrent or Kazaa.

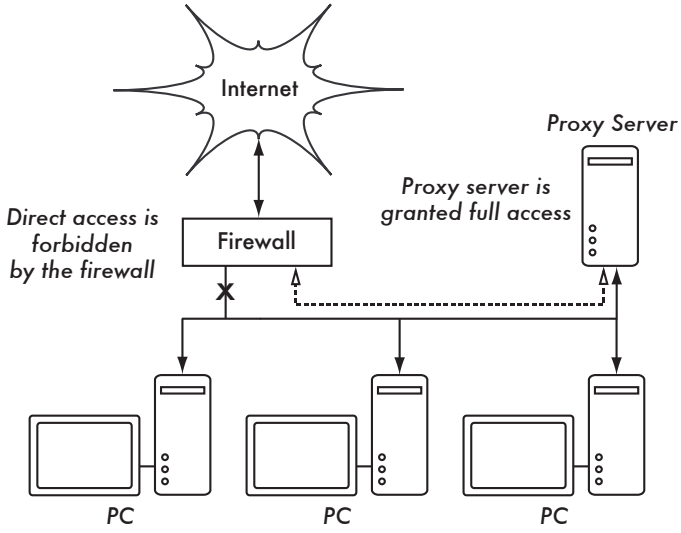

Figure 3.26: The firewall prevents PCs from accessing the Internet directly, but allows access via the proxy server.

Two network cards

Perhaps the most reliable method is to install two network cards in the proxy server and connect the campus network to the Internet as shown below. In this way, the network layout makes it physically impossible to reach the Internet without going through the proxy server.

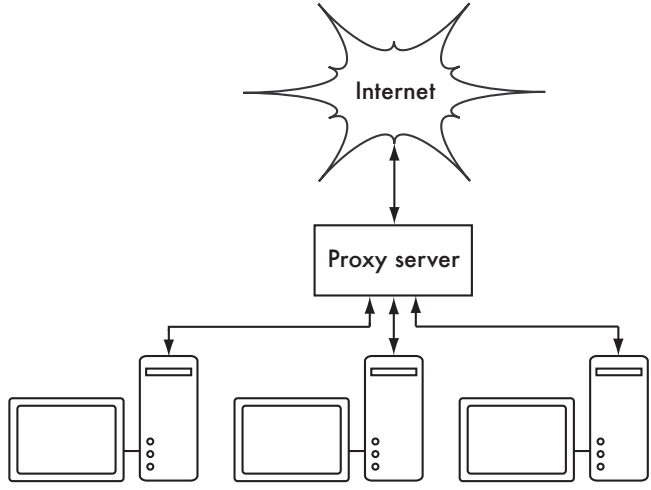

Figure 3.27: The only route to the Internet is through the proxy.

The proxy server in this diagram should not have IP forwarding enabled, unless the administrators knows exactly what they want to let through.

One big advantage to this design is that a technique known as ***transparent proxying*** can be used. Using a transparent proxy means that users' web requests are automatically forwarded to the proxy server, without any need to manually configure web browsers to use it. This effectively forces all web traffic to be cached, eliminates many chances for user error, and will even work with devices that do not support use of a manual proxy. For more details about configuring a transparent proxy with Squid, see:

- *http://www.squid-cache.org/Doc/FAQ/FAQ-17.html*
- *http://tldp.org/HOWTO/TransparentProxy.html*

Policy-based routing

One way to prevent bypassing of the proxy using Cisco equipment is with policy routing. The Cisco router transparently directs web requests to the proxy server. This technique is used at Makerere University. The advantage of this method is that, if the proxy server is down, the policy routes can be temporarily removed, allowing clients to connect directly to the Internet.

Mirroring a website

With permission of the owner or web master of a site, the whole site can be mirrored to a local server overnight, if it is not too large. This is something that might be considered for important websites that are of particular interest to the organization or that are very popular with web users. This may have some use, but it has some potential pitfalls. For example, if the site that is mirrored contains CGI scripts or other dynamic content that require interactive input from the user, this would cause problems. An example is a website that requires people to register online for a conference. If someone registers online on a mirrored server (and the mirrored script works), the organizers of the site will not have the information that the person registered.

Because mirroring a site may infringe copyright, this technique should only be used with permission of the site concerned. If the site runs ***rsync***, the site could be mirrored using rsync. This is likely the fastest and most efficient way to keep site contents synchronized. If the remote web server is not running rsync, the recommended software to use is a program called ***wget***. It is part of most versions of Unix/Linux. A Windows version can be found at *http://xoomer.virgilio.it/hherold/*, or in the free Cygwin Unix tools package (*http://www.cygwin.com/*).

A script can be set up to run every night on a local web server and do the following:

- Change directory to the web server document root: for example, **`/var/www/`** on Unix, or **`C:\Inetpub\wwwroot`** on Windows.
- Mirror the website using the command:

```
wget --cache=off -m http://www.python.org
```

The mirrored website will be in a directory **`www.python.org`**. The web server should now be configured to serve the contents of that directory as a name-based virtual host. Set up the local DNS server to fake an entry for this site. For this to work, client PCs should be configured to use the local DNS server(s) as the primary DNS. (This is advisable in any case, because a local caching DNS server speeds up web response times).

Pre-populate the cache using wget

Instead of setting up a mirrored website as described in the previous section, a better approach is to populate the proxy cache using an automated process. This method has been described by J. J. Eksteen and J. P. L. Cloete of the CSIR in Pretoria, South Africa, in a paper entitled **Enhancing International World Wide Web Access in Mozambique Through the Use of Mirroring and Caching Proxies**. In this paper (available at *http://www.isoc.org/inet97/ans97/cloet.htm*) they describe how the process works:

> "*An automatic process retrieves the site's home page and a specified number of extra pages (by recursively following HTML links on the retrieved pages) through the use of a proxy. Instead of writing the retrieved pages onto the local disk, the mirror process discards the retrieved pages. This is done in order to conserve system resources as well as to avoid possible copyright conflicts. By using the proxy as intermediary, the retrieved pages are guaranteed to be in the cache of the proxy as if a client accessed that page. When a client accesses the retrieved page, it is served from the cache and not over the congested international link. This process can be run in off-peak times in order to maximize bandwidth utilization and not to compete with other access activities.*"

The following command (scheduled to run at night once every day or week) is all that is needed (repeated for every site that needs pre-populating).

```
wget --proxy-on --cache=off --delete after -m http://www.python.org
```

These options enable the following:

- **-m**: Mirrors the entire site. wget starts at *www.python.org* and follows all hyperlinks, so it downloads all subpages.
- **--proxy-on**: Ensures that wget makes use of the proxy server. This might not be needed in set-ups where a transparent proxy is employed.
- **--cache=off**: Ensures that fresh content is retrieved from the Internet, and not from the local proxy server.
- **--delete after**: Deletes the mirrored copy. The mirrored content remains in the proxy cache if there is sufficient disk space, and the proxy server caching parameters are set up correctly.

In addition, wget has many other options; for example, to supply a password for websites that require them. When using this tool, Squid should be configured with sufficient disk space to contain all the pre-populated sites and more (for normal Squid usage involving pages other than the pre-populated ones). Fortunately, disk space is becoming ever cheaper and disk sizes are far larger than ever before. However, this technique can only be used with a few selected sites. These sites should not be too big for the process to finish before the working day starts, and an eye should be kept on disk space.

Cache hierarchies

When an organization has more than one proxy server, the proxies can share cached information among them. For example, if a web page exists in server A's cache, but not in the cache of server B, a user connected via server B might get the cached object from server A via server B. ***Inter-Cache Protocol*** (***ICP***) and ***Cache Array Routing Protocol*** (***CARP***) can share cache information. CARP is considered the better protocol. Squid supports both protocols, and MS ISA Server supports CARP. For more information, see *http://squid-docs.sourceforge.net/latest/html/c2075.html*. This sharing of cached information reduces bandwidth usage in organizations where more than one proxy is used.

Proxy specifications

On a university campus network, there should be more than one proxy server, both for performance and also for redundancy reasons. With today's cheaper and larger disks, powerful proxy servers can be built, with 50 GB or more disk space allocated to the cache. Disk performance is important, therefore the fastest SCSI disks would perform best (although an IDE based cache is better than none at all). RAID or mirroring is not recommended.

It is also recommended that a separate disk be dedicated to the cache. For example, one disk could be for the cache, and a second for the operating system and cache logging. Squid is designed to use as much RAM as it can get, because when data is retrieved from RAM it is much faster than when it

comes from the hard disk. For a campus network, RAM memory should be 1GB or more:

- Apart from the memory required for the operating system and other applications, Squid requires 10 MB of RAM for every 1 GB of disk cache. Therefore, if there is 50 GB of disk space allocated to caching, Squid will require 500 MB extra memory.
- The machine would also require 128 MB for Linux and 128 MB for Xwindows.
- Another 256 MB should be added for other applications and in order that everything can run easily. Nothing increases a machine's performance as much as installing a large amount of memory, because this reduces the need to use the hard disk. Memory is thousands of times faster than a hard disk. Modern operating systems keep frequently accessed data in memory if there is enough RAM available. But they use the page file as an extra memory area when they don't have enough RAM.

DNS caching and optimization

Caching-only DNS servers are not authoritative for any domains, but rather just cache results from queries asked of them by clients. Just like a proxy server that caches popular web pages for a certain time, DNS addresses are cached until their ***time to live*** (***TTL***) expires. This will reduce the amount of DNS traffic on your Internet connection, as the DNS cache may be able to satisfy many of the queries locally. Of course, client computers must be configured to use the caching-only name server as their DNS server. When all clients use this server as their primary DNS server, it will quickly populate a cache of IP addresses to names, so that previously requested names can quickly be resolved. DNS servers that are authoritative for a domain also act as cache name-address mappings of hosts resolved by them.

Bind (named)

Bind is the de facto standard program used for name service on the Internet. When Bind is installed and running, it will act as a caching server (no further configuration is necessary). Bind can be installed from a package such as a Debian package or an RPM. Installing from a package is usually the easiest method. In Debian, type

```
apt-get install bind9
```

In addition to running a cache, Bind can also host authoritative zones, act as a slave to authoritative zones, implement split horizon, and just about everything else that is possible with DNS.

dnsmasq

One alternative caching DNS server is ***dnsmasq***. It is available for BSD and most Linux distributions, or from *http://www.thekelleys.org.uk/dnsmasq/*. The big advantage of dnsmasq is flexibility: it easily acts as both a caching DNS proxy and an authoritative source for hosts and domains, without complicated zone file configuration. Updates can be made to zone data without even restarting the service. It can also serve as a DHCP server, and will integrate DNS service with DHCP host requests. It is very lightweight, stable, and extremely flexible. Bind is likely a better choice for very large networks (more than a couple of hundred nodes), but the simplicity and flexibility of dnsmasq makes it attractive for small to medium sized networks.

Windows NT

To install the DNS service on Windows NT4: select Control Panel ➔ Network ➔ Services ➔ Add ➔ Microsoft DNS server. Insert the Windows NT4 CD when prompted. Configuring a caching-only server in NT is described in Knowledge Base article 167234. From the article:

> "*Simply install DNS and run the Domain Name System Manager. Click on DNS in the menu, select New Server, and type in the IP address of your computer where you have installed DNS. You now have a caching-only DNS server.*"

Windows 2000

Install DNS service: Start ➔ Settings ➔ Control Panel ➔ Add/Remove Software. In Add/Remove Windows Components, select Components ➔ Networking Services ➔ Details ➔ Domain Name System (DNS). Then start the DNS MMC (Start ➔ Programs ➔ Administrative Tools ➔ DNS) From the Action menu select "Connect To Computer..." In the Select Target Computer window, enable "The following computer:" and enter the name of a DNS server you want to cache. If there is a . [dot] in the DNS manager (this appears by default), this means that the DNS server thinks it is the root DNS server of the Internet. It is certainly not. Delete the . [dot] for anything to work.

Split DNS and a mirrored server

The aim of split DNS (also known as ***split horizon***) is to present a different view of your domain to the inside and outside worlds. There is more than one way to do split DNS; but for security reasons, it's recommended that you have two separate internal and external content DNS servers (each with different databases).

Split DNS can enable clients from a campus network to resolve IP addresses for the campus domain to local RFC1918 IP addresses, while the rest of the

Internet resolves the same names to different IP addresses. This is achieved by having two zones on two different DNS servers for the same domain.

One of the zones is used by internal network clients and the other by users on the Internet. For example, in the network below the user on the Makerere campus gets *http://www.makerere.ac.ug/* resolved to 172.16.16.21, whereas a user elsewhere on the Internet gets it resolved to 195.171.16.13.

The DNS server on the campus in the above diagram has a zone file for *makerere.ac.ug* and is configured as if it is authoritative for that domain. In addition, it serves as the DNS caching server for the Makerere campus, and all computers on the campus are configured to use it as their DNS server.

The DNS records for the campus DNS server would look like this:

```
makerere.ac.ug
www CNAME      webserver.makerere.ac.ug
ftp CNAME      ftpserver.makerere.ac.ug
mail CNAME     exchange.makerere.ac.ug
mailserver     A       172.16.16.21
webserver      A       172.16.16.21
ftpserver      A       172.16.16.21
```

But there is another DNS server on the Internet that is actually authoritative for the *makerere.ac.ug* domain. The DNS records for this external zone would look like this:

```
makerere.ac.ug
www A 195.171.16.13
ftp A 195.171.16.13
mail    A 16.132.33.21
        MX mail.makerere.ac.ug
```

Split DNS is not dependent on using RFC 1918 addresses. An African ISP might, for example, host websites on behalf of a university but also mirror those same websites in Europe. Whenever clients of that ISP access the website, it gets the IP address at the African ISP, and so the traffic stays in the same country. When visitors from other countries access that website, they get the IP address of the mirrored web server in Europe. In this way, international visitors do not congest the ISP's VSAT connection when visiting the university's website. This is becoming an attractive solution, as web hosting close to the Internet backbone has become very cheap.

Internet link optimization

As mentioned earlier, network throughput of up to 22 Mbps can be achieved by using standard, unlicensed 802.11g wireless gear. This amount of bandwidth will likely be at least an order of magnitude higher than that provided by

your Internet link, and should be able to comfortably support many simultaneous Internet users.

But if your primary Internet connection is through a VSAT link, you will encounter some performance issues if you rely on default TCP/IP parameters. By optimizing your VSAT link, you can significantly improve response times when accessing Internet hosts.

TCP/IP factors over a satellite connection

A VSAT is often referred to as a ***long fat pipe network***. This term refers to factors that affect TCP/IP performance on any network that has relatively large bandwidth, but high latency. Most Internet connections in Africa and other parts of the developing world are via VSAT. Therefore, even if a university gets its connection via an ISP, this section might apply if the ISP's connection is via VSAT. The high latency in satellite networks is due to the long distance to the satellite and the constant speed of light. This distance adds about 520 ms to a packet's round-trip time (RTT), compared to a typical RTT between Europe and the USA of about 140 ms.

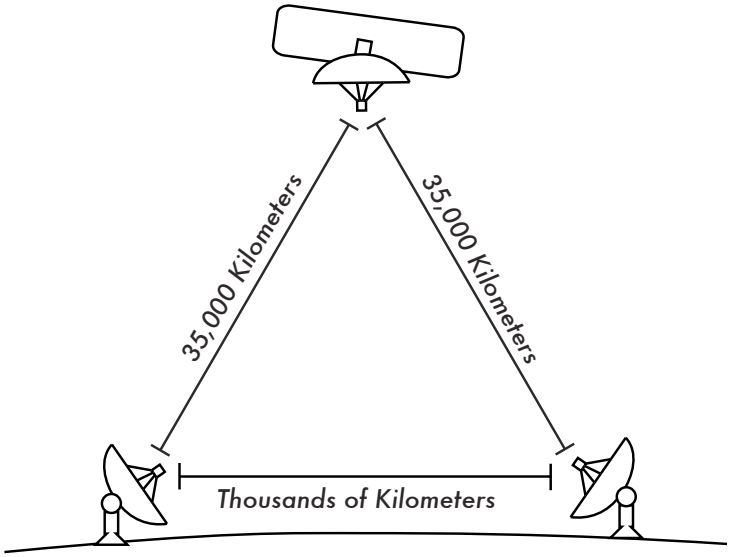

Figure 3.28: Due to the speed of light and long distances involved, a single ping packet can take more than 520 ms to be acknowledged over a VSAT link.

The factors that most significantly impact TCP/IP performance are **long RTT**, **large bandwidth delay product**, and **transmission errors**.

Generally speaking, operating systems that support modern TCP/IP implementations should be used in a satellite network. These implementations support the RFC 1323 extensions: