Ethernet cable, a wireless link, or any other kind of physical network. The cloud symbol is commonly used to stand in for “The Internet”, and represents any number of intervening IP networks. Neither Alice nor Bob need to be concerned with how those networks operate, as long as the routers forward IP traffic towards the ultimate destination. If it weren’t for Internet protocols and the cooperation of everyone on the net, this kind of communication would be impossible.

Designing the physical network

It may seem odd to talk about the “physical” network when building wireless networks. After all, where is the physical part of the network? In wireless networks, the physical medium we use for communication is obviously electromagnetic energy. But in the context of this chapter, the physical network refers to the mundane topic of where to put things. How do you arrange the equipment so that you can reach your wireless clients? Whether they fill an office building or stretch across many miles, wireless networks are naturally arranged in these three logical configurations: ***point-to-point links***, ***point-to-multipoint links***, and ***multipoint-to-multipoint clouds***. While different parts of your network can take advantage of all three of these configurations, any individual link will fall into one of these topologies.

Point-to-point

Point-to-point links typically provide an Internet connection where such access isn’t otherwise available. One side of a point-to-point link will have an Internet connection, while the other uses the link to reach the Internet. For example, a university may have a fast frame relay or VSAT connection in the middle of campus, but cannot afford such a connection for an important building just off campus. If the main building has an unobstructed view of the remote site, a point-to-point connection can be used to link the two together. This can augment or even replace existing dial-up links. With proper antennas and clear line of sight, reliable point-to-point links in excess of thirty kilometers are possible.

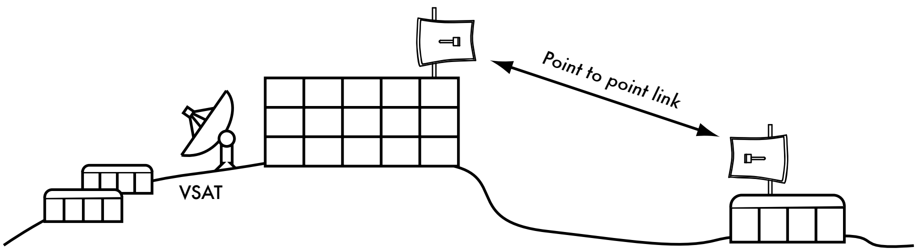

Figure 3.14: A point-to-point link allows a remote site to share a central Internet connection.

Of course, once a single point-to-point connection has been made, more can be used to extend the network even further. If the remote building in our example is at the top of a tall hill, it may be able to see other important locations that can't be seen directly from the central campus. By installing another point-to-point link at the remote site, another node can join the network and make use of the central Internet connection.

Point-to-point links don't necessarily have to involve Internet access. Suppose you have to physically drive to a remote weather monitoring station, high in the hills, in order to collect the data which it records over time. You could connect the site with a point-to-point link, allowing data collection and monitoring to happen in realtime, without the need to actually travel to the site. Wireless networks can provide enough bandwidth to carry large amounts of data (including audio and video) between any two points that have a connection to each other, even if there is no direct connection to the Internet.

Point-to-multipoint

The next most commonly encountered network layout is ***point-to-multipoint***. Whenever several nodes[2] are talking to a central point of access, this is a point-to-multipoint application. The typical example of a point-to-multipoint layout is the use of a wireless ***access point*** that provides a connection to several laptops. The laptops do not communicate with each other directly, but must be in range of the access point in order to use the network.

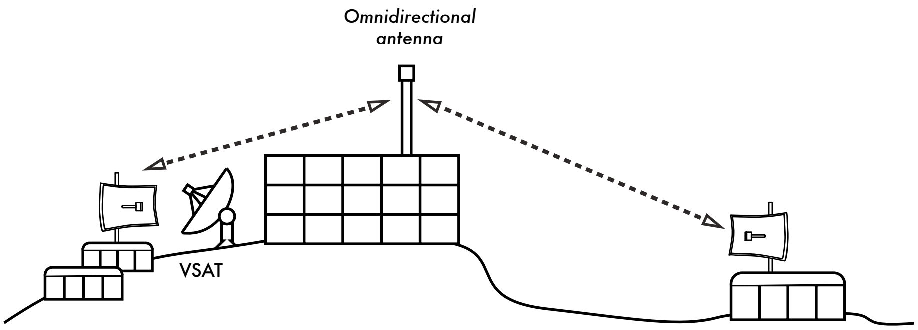

Figure 3.15: The central VSAT is now shared by multiple remote sites. All three sites can also communicate directly at speeds much faster than VSAT.

Point-to-multipoint networking can also apply to our earlier example at the university. Suppose the remote building on top of the hill is connected to the central campus with a point-to-point link. Rather than setting up several point-to-point links to distribute the Internet connection, a single antenna could be used that is visible from several remote buildings. This is a classic

2. A ***node*** is any device capable of sending and receiving data on a network. Access points, routers, computers, and laptops are all examples of nodes.

example of a wide area ***point*** (remote site on the hill) ***to multipoint*** (many buildings in the valley below) connection.

Note that there are a number of performance issues with using point-to-multipoint over very long distance, which will be addressed later in this chapter. Such links are possible and useful in many circumstances, but don't make the classic mistake of installing a single high powered radio tower in the middle of town and expecting to be able to serve thousands of clients, as you would with an FM radio station. As we will see, two-way data networks behave very differently than broadcast radio.

Multipoint-to-multipoint

The third type of network layout is ***multipoint-to-multipoint***, which is also referred to as an ***ad-hoc*** or ***mesh*** network. In a multipoint-to-multipoint network, there is no central authority. Every node on the network carries the traffic of every other as needed, and all nodes communicate with each other directly.

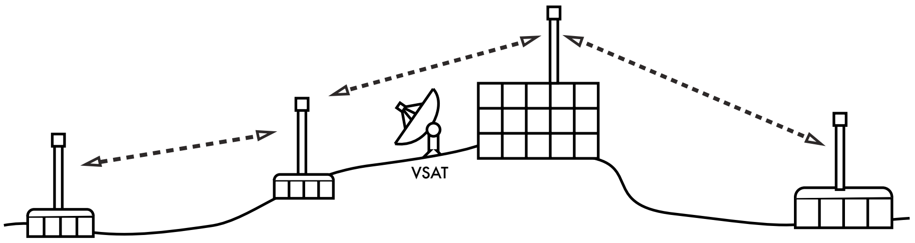

Figure 3.16: A multipoint-to-multipoint mesh. Every point can reach each other at very high speed, or use the central VSAT connection to reach the Internet.

The benefit of this network layout is that even if none of the nodes are in range of a central access point, they can still communicate with each other. Good mesh network implementations are self-healing, which means that they automatically detect routing problems and fix them as needed. Extending a mesh network is as simple as adding more nodes. If one of the nodes in the "cloud" happens to be an Internet gateway, then that connection can be shared among all of the clients.

Two big disadvantages to this topology are increased complexity and lower performance. Security in such a network is also a concern, since every participant potentially carries the traffic of every other. Multipoint-to-multipoint networks tend to be difficult to troubleshoot, due to the large number of changing variables as nodes join and leave the network. Multipoint-to-multipoint clouds typically have reduce capacity compared to point-to-point or point-to-multipoint networks, due to the additional overhead of managing the network routing and increased contention in the radio spectrum.

Nevertheless, mesh networks are useful in many circumstances. We will see an example of how to build a multipoint-to-multipoint mesh network using a routing protocol called OLSR later in this chapter.

Use the technology that fits

All of these network designs can be used to complement each other in a large network, and can obviously make use of traditional wired networking techniques whenever possible. It is a common practice, for example, to use a long distance wireless link to provide Internet access to a remote location, and then set up an access point on the remote side to provide local wireless access. One of the clients of this access point may also act as a mesh node, allowing the network to spread organically between laptop users who all ultimately use the original point-to-point link to access the Internet.

Now that we have a clear idea of how wireless networks are typically arranged, we can begin to understand how communication is possible over such networks.

802.11 wireless networks

Before packets can be forwarded and routed to the Internet, layers one (the physical) and two (the data link) need to be connected. Without link local connectivity, network nodes cannot talk to each other and route packets.

To provide physical connectivity, wireless network devices must operate in the same part of the radio spectrum. As we saw in **Chapter 2**, this means that 802.11a radios will talk to 802.11a radios at around 5 GHz, and 802.11b/g radios will talk to other 802.11b/g radios at around 2.4 GHz. But an 802.11a device cannot interoperate with an 802.11b/g device, since they use completely different parts of the electromagnetic spectrum.

More specifically, wireless cards must agree on a common channel. If one 802.11b radio card is set to channel 2 while another is set to channel 11, then the radios cannot communicate with each other.

When two wireless cards are configured to use the same protocol on the same radio channel, then they are ready to negotiate data link layer connectivity. Each 802.11a/b/g device can operate in one of four possible modes:

1. ***Master mode*** (also called ***AP*** or ***infrastructure mode***) is used to create a service that looks like a traditional access point. The wireless card creates a network with a specified name (called the ***SSID***) and channel, and offers network services on it. While in master mode, wireless cards manage all communications related to the network (authenticating wire-

less clients, handling channel contention, repeating packets, etc.) Wireless cards in master mode can only communicate with cards that are associated with it in managed mode.

2. ***Managed mode*** is sometimes also referred to as ***client*** mode. Wireless cards in managed mode will join a network created by a master, and will automatically change their channel to match it. They then present any necessary credentials to the master, and if those credentials are accepted, they are said to be ***associated*** with the master. Managed mode cards do not communicate with each other directly, and will only communicate with an associated master.

3. ***Ad-hoc mode*** creates a multipoint-to-multipoint network where there is no single master node or AP. In ad-hoc mode, each wireless card communicates directly with its neighbors. Nodes must be in range of each other to communicate, and must agree on a network name and channel.

4. ***Monitor mode*** is used by some tools (such as **Kismet**, see **Chapter 6**) to passively listen to all radio traffic on a given channel. When in monitor mode, wireless cards transmit no data. This is useful for analyzing problems on a wireless link or observing spectrum usage in the local area. Monitor mode is not used for normal communications.

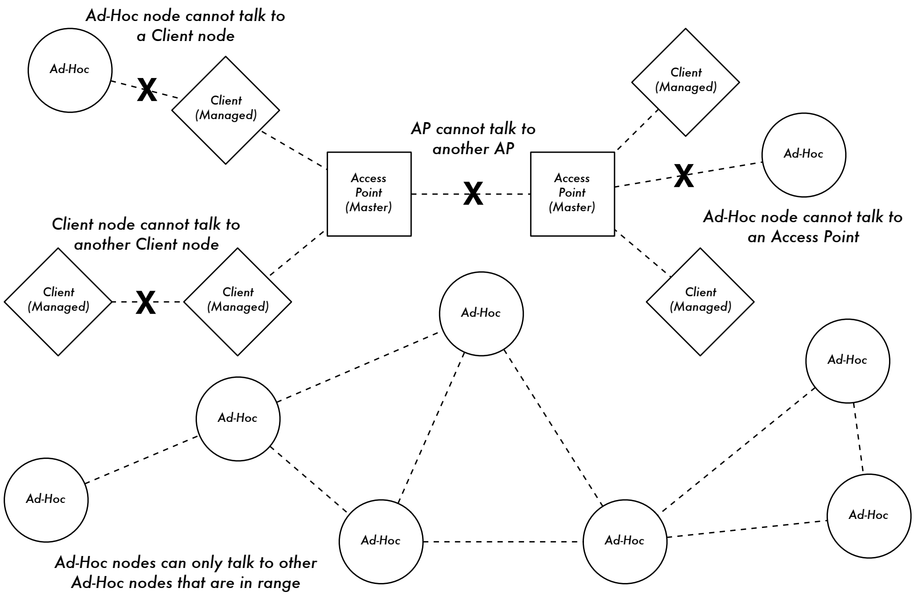
Ad-Hoc node cannot talk to a Client node
Ad-Hoc
X
Client (Managed)
AP cannot talk to another AP
Access Point (Master)
X
Access Point (Master)
Client (Managed)
X
Ad-Hoc
Ad-Hoc node cannot talk to an Access Point
Client node cannot talk to another Client node
Client (Managed)
X
Client (Managed)
Client (Managed)
Ad-Hoc
Ad-Hoc
Ad-Hoc
Ad-Hoc
Ad-Hoc
Ad-Hoc
Ad-Hoc
Ad-Hoc nodes can only talk to other Ad-Hoc nodes that are in range

Figure 3.17: APs, Clients, and Ad-Hoc nodes.

When implementing a point-to-point or point-to-multipoint link, one radio will typically operate in master mode, while the other(s) operate in managed mode. In a multipoint-to-multipoint mesh, the radios all operate in ad-hoc mode so that they can communicate with each other directly.

It is important to keep these modes in mind when designing your network layout. Remember that managed mode clients cannot communicate with each other directly, so it is likely that you will want to run a high repeater site in master or ad-hoc mode. As we will see later in this chapter, ad-hoc is more flexible but has a number of performance issues as compared to using the master / managed modes.

Mesh networking with OLSR

Most WiFi networks operate in infrastructure mode - they consist of an access point somewhere (with a radio operating in master mode), attached to a DSL line or other large scale wired network. In such a ***hotspot*** the access point usually acts as a master station that is distributing Internet access to its clients, which operate in managed mode. This topology is similar to a mobile phone (GSM) service. Mobile phones connect to a base station - without the presence of such a base station mobiles can't communicate with each other. If you make a joke call to a friend that is sitting on the other side of the table, your phone sends data to the base station of your provider that may be a mile away - the base station then sends data back to the phone of your friend.

WiFi cards in managed mode can't communicate directly, either. Clients - for example, two laptops on the same table - have to use the access point as a relay. Any traffic between clients connected to an access point has to be sent twice. If client A and C communicate, client A sends data to the access point B, and then the access point will retransmit the data to client C. A single transmission may have a speed of 600 kByte/sec (thats about the maximum speed you could achieve with 802.11b) in our example - thus, because the data has to be repeated by the access point before it reaches its target, the effective speed between both clients will be only 300 kByte/sec.

In ad-hoc mode there is no hierarchical master-client relationship. Nodes can communicate directly as long as they are within the range of their wireless interfaces. Thus, in our example both computers could achieve full speed when operating ad-hoc, under ideal circumstances.

The disadvantage to ad-hoc mode is that clients do not repeat traffic destined for other clients. In the access point example, if two clients A and C can't directly “see” each other with their wireless interfaces, they still can communicate as long as the AP is in the wireless range of both clients.

Ad-hoc nodes do not repeat by default, but they can effectively do the same if ***routing*** is applied. Mesh networks are based on the strategy that every mesh-enabled node acts as a relay to extend coverage of the wireless network. The more nodes, the better the radio coverage and range of the mesh cloud.

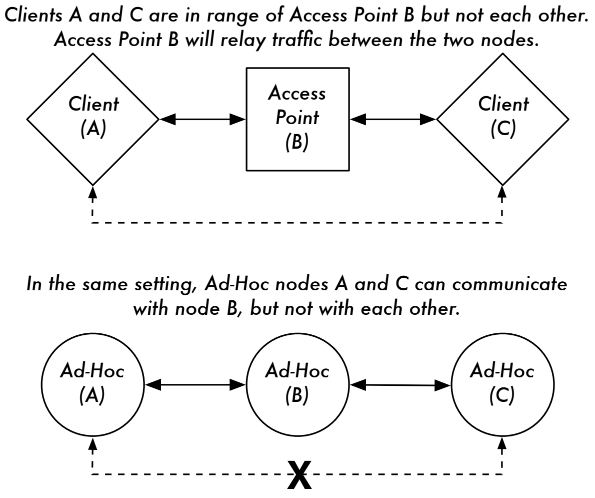
Clients A and C are in range of Access Point B but not each other.
Access Point B will relay traffic between the two nodes.
Client (A)
Access Point (B)
Client (C)
In the same setting, Ad-Hoc nodes A and C can communicate with node B, but not with each other.
Ad-Hoc (A)
Ad-Hoc (B)
Ad-Hoc (C)
X

Figure 3.18: Access point B will relay traffic between clients A and C. In Ad-Hoc mode, node B will not relay traffic between A and C by default.

There is one big tradeoff that must be mentioned at this point. If the device only uses one radio interface, the available bandwidth is significantly reduced every time traffic is repeated by intermediate nodes on the way from A to B. Also, there will be interference in transmission due to nodes sharing the same channel. Thus, cheap ad-hoc mesh networks can provide good radio coverage on the last mile(s) of a community wireless network at the cost of speed-- especially if the density of nodes and transmit power is high.

If an ad-hoc network consists of only a few nodes that are up and running at all time, don't move and always have stable radio links - a long list of ifs - it is possible to write individual routing tables for all nodes by hand.

Unfortunately, those conditions are rarely met in the real world. Nodes can fail, WiFi enabled devices roam around, and interference can make radio links unusable at any time. And no one wants to update several routing tables by hand if one node is added to the network. By using routing protocols that automatically maintain individual routing tables in all nodes involved, we can avoid these issues. Popular routing protocols from the wired world (such as OSPF) do not work well in such an environment because they are not designed to deal with lossy links or rapidly changing topology.

Mesh routing with olsrd

The Optimized Link State Routing Daemon - olsrd - from *olsr.org* is a routing application developed for routing in wireless networks. We will concentrate on this routing software for several reasons. It is a open-source project that supports Mac OS X, Windows 98, 2000, XP, Linux, FreeBSD, OpenBSD and

NetBSD. Olsrd is available for access points that run Linux like the Linksys WRT54G, Asus Wl500g, AccessCube or Pocket PCs running Familiar Linux, and ships standard on Metrix kits running Pyramid. Olsrd can handle multiple interfaces and is extensible with plug-ins. It supports IPv6 and it is actively developed and used by community networks all over the world.

Note that there are several implementations of Optimized Link State Routing, which began as an IETF-draft written at INRIA France. The implementation from *olsr.org* started as a master thesis of Andreas Toennesen at UniK University. Based on practical experience of the free networking community, the routing daemon was modified. Olsrd now differs significantly from the original draft because it includes a mechanism called Link Quality Extension that measures the packet loss between nodes and calculates routes according to this information. This extension breaks compatibility to routing daemons that follow the INRIA draft. The olsrd available from *olsr.org* can be configured to behave according to the IETF draft that lacks this feature - but there is no reason to disable Link Quality Extension unless compliance with other implementations is required.

Theory

After olsrd is running for a while, a node knows about the existence of every other node in the mesh cloud and which nodes may be used to route traffic to them. Each node maintains a routing table covering the whole mesh cloud. This approach to mesh routing is called ***proactive routing***. In contrast, ***reactive routing*** algorithms seek routes only when it is necessary to send data to a specific node.

There are pros and cons to proactive routing, and there are many other ideas about how to do mesh routing that may be worth mentioning. The biggest advantage of proactive routing is that you know who is out there and you don't have to wait until a route is found. Higher protocol traffic overhead and more CPU load are among the disadvantages. In Berlin, the Freifunk community is operating a mesh cloud where olsrd has to manage more than 100 interfaces. The average CPU load caused by olsrd on a Linksys WRT54G running at 200 MHz is about 30% in the Berlin mesh. There is clearly a limit to what extent a proactive protocol can scale - depending on how many interfaces are involved and how often the routing tables are updated. Maintaining routes in a mesh cloud with static nodes takes less effort than a mesh with nodes that are constantly in motion, since the routing table has to be updated less often.

Mechanism

A node running olsrd is constantly broadcasting 'Hello' messages at a given interval so neighbors can detect it's presence. Every node computes a statistic how many 'Hellos' have been lost or received from each neighbor -

thereby gaining information about the topology and link quality of nodes in the neighborhood. The gained topology information is broadcasted as topology control messages (TC messages) and forwarded by neighbors that olsrd has chosen to be multipoint relays.

The concept of multipoint relays is a new idea in proactive routing that came up with the OLSR draft. If every node rebroadcasts topology information that it has received, unnecessary overhead can be generated. Such transmissions are redundant if a node has many neighbors. Thus, an olsrd node decides which neighbors are favorable multipoint relays that should forward its topology control messages. Note that multipoint relays are only chosen for the purpose of forwarding TC messages. Payload is routed considering all available nodes.

Two other message types exist in OLSR that announce information: whether a node offers a gateway to other networks (HNA messages) or has multiple interfaces (MID messages). There is not much to say about what this messages do apart from the fact that they exist. HNA messages make olsrd very convenient when connecting to the Internet with a mobile device. When a mesh node roams around it will detect gateways into other networks and always choose the gateway that it has the best route to. However, olsrd is by no means bullet proof. If a node announces that it is an Internet gateway - which it isn't because it never was or it is just offline at the moment - the other nodes will nevertheless trust this information. The pseudo-gateway is a black hole. To overcome this problem, a dynamic gateway plugin was written. The plugin will automatically detect at the gateway if it is actually connected and whether the link is still up. If not, olsrd ceases to send false HNA messages. It is highly recommended to build and use this plugin instead of statically enabling HNA messages.

Practice

Olsrd implements IP-based routing in a userland application - installation is pretty easy. Installation packages are available for OpenWRT, AccessCube, Mac OS X, Debian GNU/Linux and Windows. OLSR is a standard part of Metrix Pyramid. If you have to compile from source, please read the documentation that is shipped with the source package. If everything is configured properly all you have to do is start the olsr program.

First of all, it must be ensured that every node has a unique statically assigned IP-Address for each interface used for the mesh. It is not recommended (nor practicable) to use DHCP in an IP-based mesh network. A DHCP request will not be answered by a DHCP server if the node requesting DHCP needs a multihop link to connect to it, and applying dhcp relay throughout a mesh is likely impractical. This problem could be solved by using IPv6, since there is plenty of space available to generate a unique IP from the MAC address of each card involved (as suggested in "IPv6 State-less Address Autoconfiguration in large mobile ad hoc networks" by K. Weniger and M. Zitterbart, 2002).

A wiki-page where every interested person can choose an individual IPv4 address for each interface the olsr daemon is running on may serve the purpose quite well. There is just not an easy way to automate the process if IPv4 is used.

The broadcast address should be 255.255.255.255 on mesh interfaces in general as a convention. There is no reason to enter the broadcast address explicitly, since olsrd can be configured to override the broadcast addresses with this default. It just has to be ensured that settings are the same everywhere. Olsrd can do this on its own. When a default olsrd configuration file is issued, this feature should be enabled to avoid confusion of the kind "why can't the other nodes see my machine?!?"

Now configure the wireless interface. Here is an example command how to configure a WiFi card with the name wlan0 using Linux:

```
iwconfig wlan0 essid olsr.org mode ad-hoc channel 10 rts 250 frag 256
```

Verify that the wireless part of the WiFi card has been configured so it has an ad-hoc connection to other mesh nodes within direct (single hop) range. Make sure the interface joins the same wireless channel, uses the same wireless network name ESSID (Extended Service Set IDentifier) and has the same Cell-ID as all other WiFi-Cards that build the mesh. Many WiFi cards or their respective drivers do not comply with the 802.11 standard for ad-hoc networking and may fail miserably to connect to a cell. They may be unable to connect to other devices on the same table, even if they are set up with the correct channel and wireless network name. They may even confuse other cards that behave according to the standard by creating their own Cell-ID on the same channel with the same wireless network name. WiFi cards made by Intel that are shipped with Centrino Notebooks are notorious for doing this.

You can check this out with the command **`iwconfig`** when using GNU-Linux. Here is the output on my machine:

```
wlan0 IEEE 802.11b  ESSID:"olsr.org"
 Mode:Ad-Hoc  Frequency:2.457 GHz  Cell: 02:00:81:1E:48:10
 Bit Rate:2 Mb/s   Sensitivity=1/3
 Retry min limit:8   RTS thr=250 B   Fragment thr=256 B
 Encryption key:off
 Power Management:off
 Link Quality=1/70  Signal level=-92 dBm  Noise level=-100 dBm
 Rx invalid nwid:0  Rx invalid crypt:28  Rx invalid frag:0
 Tx excessive retries:98024 Invalid misc:117503 Missed beacon:0
```

It is important to set the 'Request To Send' threshold value RTS for a mesh. There will be collisions on the radio channel between the transmissions of nodes on the same wireless channel, and RTS will mitigate this. RTS/CTS adds a handshake before each packet transmission to make sure that the channel is clear. This adds overhead, but increases performance in case of hidden nodes - and hidden nodes are the default in a mesh! This parameter sets the size of the smallest packet (in bytes) for which the node sends RTS. The RTS threshold value must be smaller than the IP-Packet size and the 'Fragmentation threshold' value - here set to 256 - otherwise it will be disabled. TCP is very sensitive to collisions, so it is important to switch RTS on.

Fragmentation allows to split an IP packet in a burst of smaller fragments transmitted on the medium. This adds overhead, but in a noisy environment this reduces the error penalty and allows packets to get through interference bursts. Mesh networks are very noisy because nodes use the same channel and therefore transmissions are likely to interfere with each other. This parameter sets the maximum size before a data packet is split and sent in a burst - a value equal to the maximum IP packet size disables the mechanism, so it must be smaller than the IP packet size. Setting fragmentation threshold is recommended.

Once a valid IP-address and netmask is assigned and the wireless interface is up, the configuration file of olsrd must be altered in order that olsrd finds and uses the interfaces it is meant to work on.

For Mac OS-X and Windows there are nice GUI's for configuration and monitoring of the daemon available. Unfortunately this tempts users that lack background knowledge to do stupid things - like announcing black holes. On BSD and Linux the configuration file **`/etc/olsrd.conf`** has to be edited with a text editor.

A simple olsrd.conf

It is not practical to provide a complete configuration file here. These are some essential settings that should be checked.

```
UseHysteresis          no
TcRedundancy           2
MprCoverage            3
LinkQualityLevel       2
LinkQualityWinSize     20

LoadPlugin "olsrd_dyn_gw.so.0.3"
{
     PlParam     "Interval"   "60"
     PlParam     "Ping"       "151.1.1.1"
    PlParam     "Ping"       "194.25.2.129"
}

Interface "ath0" "wlan0" {
 Ip4Broadcast 255.255.255.255
}
```

There are many more options available in the **olsrd.conf**, but these basic options should get you started. After these steps have been done, olsrd can be started with a simple command in a terminal:

```
olsrd -d 2
```

I recommend to run it with the debugging option -d 2 when used on a workstation, especially for the first time. You can see what olsrd does and monitor how well the links to your neighbors are. On embedded devices the debug level should be 0 (off), because debugging creates a lot of CPU load.

The output should look something like this:

```
--- 19:27:45.51 --------------------------------------------- DIJKSTRA

192.168.120.1:1.00 (one-hop)
192.168.120.3:1.00 (one-hop)

--- 19:27:45.51 ------------------------------------------------ LINKS

IP address       hyst   LQ     lost   total  NLQ    ETX
192.168.120.1    0.000  1.000  0      20     1.000  1.00
192.168.120.3    0.000  1.000  0      20     1.000  1.00

--- 19:27:45.51 -------------------------------------------- NEIGHBORS

IP address       LQ     NLQ    SYM   MPR   MPRS  will
192.168.120.1    1.000  1.000  YES   NO    YES   3
192.168.120.3    1.000  1.000  YES   NO    YES   6

--- 19:27:45.51 --------------------------------------------- TOPOLOGY

Source IP addr   Dest IP addr     LQ     ILQ    ETX
192.168.120.1    192.168.120.17   1.000  1.000  1.00
192.168.120.3    192.168.120.17   1.000  1.000  1.00
```

Using OLSR on Ethernet and multiple interfaces

It is not necessary to have a wireless interface to test or use olsrd - although that is what olsrd is designed for. It may as well be used on any NIC. WiFi-interfaces don't have to operate always in ad-hoc mode to form a mesh when mesh nodes have more than one interface. For dedicated links it may be a very good option to have them running in infrastructure mode. Many WiFi cards and drivers are buggy in ad-hoc mode, but infrastructure mode works fine - because everybody expects at least this feature to work. Ad-hoc mode has not had many users so far, so the implementation of the ad-hoc mode was done sloppily by many manufacturers. With the rising popularity of mesh networks, the driver situation is improving now.

Many people use olsrd on wired and wireless interfaces - they don't think about network architecture. They just connect antennas to their WiFi cards, connect cables to their Ethernet cards, enable olsrd to run on all computers and all interfaces and fire it up. That is quite an abuse of a protocol that was designed to do wireless networking on lossy links - but - why not?

They expect olsrd to solve every networking problem. Clearly it is not necessary to send 'Hello' messages on a wired interface every two seconds - but it works. This should not be taken as a recommendation - it is just amazing what people do with such a protocol and that they have such success with it. In fact the idea of having a protocol that does everything for newbies that want to have a small to medium sized routed LAN is very appealing.

Plugins

A number of plugins are available for olsrd. Check out the *olsr.org* website for a complete list. Here a little HOWTO for the network topology visualization plugin **`olsrd_dot_draw`**.

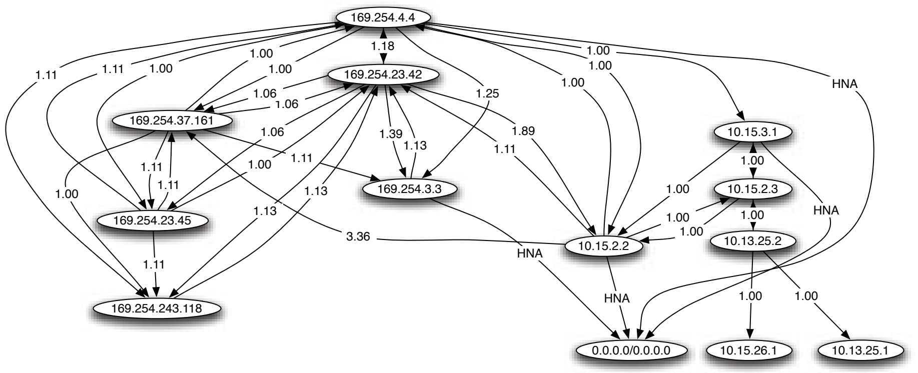

Figure 3.19: An automatically generated OLSR network topology.

Often it is very good for the understanding of a mesh network to have the ability to show the network topology graphically. **`olsrd_dot_draw`** outputs the topology in the dot file format on TCP port 2004. The graphviz tools can then be used to draw the graphs.

Installing the dot_draw Plugin

Compile the olsr plugins separately and install them. To load the plugin add the following lines to **`/etc/olsrd.conf`**. The parameter "accept" specifies which host is accepted to view the Topology Information (currently only one) and is "localhost" by default. The parameter "port" specifies the TCP port.

```
LoadPlugin "olsrd_dot_draw.so.0.3"
{
	PlParam "accept" "192.168.0.5"
	PlParam "port" "2004"
}
```

Then restart olsr and check if you get output on TCP Port 2004

```
telnet localhost 2004
```

After a while you should get some text output.

Now you can save the output graph descriptions and run the tools **`dot`** or **`neato`** form the graphviz package to get images.

Bruno Randolf has written a small perl script which continuously gets the topology information from olsrd and displays it using the graphviz and ImageMagick tools.

First install the following packages on your workstation:

- graphviz, *http://www.graphviz.org/*
- ImageMagick, *http://www.imagemagick.org/*

Download the script at: *http://meshcube.org/nylon/utils/olsr-topology-view.pl*

Now you can start the script with **`./olsr-topology-view.pl`** and view the topology updates in near-realtime.

Troubleshooting

As long as the WiFi-cards can 'see' each other directly with their radios, doing a ping will work whether olsrd is running or not. This works because the large netmasks effectively make every node link-local, so routing issues are side-stepped at the first hop. This should be checked first if things do not seem to work as expected. Most headaches people face with WiFi in Ad-Hoc mode are caused by the fact that the ad-hoc mode in drivers and cards are implemented sloppily. If it is not possible to ping nodes directly when they are in range it is most likely a card/driver issue, or your network settings are wrong.

If the machines can ping each other, but olsrd doesn't find routes, then the IP-addresses, netmask and broadcast address should be checked.

Finally, are you running a firewall? Make sure it doesn't block UDP port 698.

Estimating capacity

Wireless links can provide significantly greater ***throughput*** to users than traditional Internet connections, such as VSAT, dialup, or DSL. Throughput is also referred to as ***channel capacity***, or simply ***bandwidth*** (although this term is unrelated to radio bandwidth). It is important to understand that a wireless device's listed speed (the ***data rate***) refers to the rate at which the radios can exchange symbols, not the usable throughput you will observe. As mentioned earlier, a single 802.11g link may use 54 Mbps radios, but it will only provide up to 22 Mbps of actual throughput. The rest is overhead that the radios need in order to coordinate their signals using the 802.11g protocol.

Note that throughput is a measurement of bits over time. 22 Mbps means that in any given second, up to 22 megabits can be sent from one end of the link to the other. If users attempt to push more than 22 megabits through the link, it will take longer than one second. Since the data can't be sent immediately, it is put in a ***queue***, and transmitted as quickly as possible. This backlog of data increases the time needed for the most recently queued bits to the traverse the link. The time that it takes for data to traverse a link is called ***latency***, and high latency is commonly referred to as ***lag***. Your link will eventually send all of the queued traffic, but your users will likely complain as the lag increases.

How much throughput will your users really need? It depends on how many users you have, and how they use the wireless link. Various Internet applications require different amounts of throughput.

Application	BW / User	Notes
Text messaging / IM	< 1 kbps	As traffic is infrequent and asynchronous, IM will tolerate high latency.
Email	1 to 100 kbps	As with IM, email is asynchronous and intermittent, so it will tolerate latency. Large attachments, viruses, and spam significantly add to bandwidth usage. Note that web email services (such as Yahoo or Hotmail) should be considered as web browsing, not as email.
Web browsing	50 - 100+ kbps	Web browsers only use the network when data is requested. Communication is asynchronous, so a fair amount of lag can be tolerated. As web browsers request more data (large images, long downloads, etc.) bandwidth usage will go up significantly.

Application	BW / User	Notes
Streaming audio	96 - 160 kbps	Each user of a streaming audio service will use a constant amount of relatively large bandwidth for as long as it plays. It can tolerate some transient latency by using large buffers on the client. But extended periods of lag will cause audio “skips” or outright session failures.
Voice over IP (VoIP)	24 - 100+ kbps	As with streaming audio, VoIP commits a constant amount of bandwidth to each user for the duration of the call. But with VoIP, the bandwidth is used roughly equally in both directions. Latency on a VoIP connection is immediate and annoying to users. Lag greater than a few milliseconds is unacceptable for VoIP.
Streaming video	64 - 200+ kbps	As with streaming audio, some intermittent latency is avoided by using buffers on the client. Streaming video requires high throughput and low latency to work properly.
Peer-to-peer file-sharing applications (BitTorrent, KaZaA, Gnutella, eDonkey, etc.)	0 - infinite Mbps	While peer to peer applications will tolerate any amount of latency, they tend to use up all available throughput by transmitting data to as many clients as possible, as quickly as possible. Use of these applications will cause latency and throughput problems for all other network users unless you use careful bandwidth shaping.

To estimate the necessary throughput you will need for your network, multiply the expected number of users by the sort of application they will probably use. For example, 50 users who are chiefly browsing the web will likely consume 2.5 to 5 Mbps or more of throughput at peak times, and will tolerate some latency. On the other hand, 50 simultaneous VoIP users would require 5 Mbps or more of throughput **in both directions** with absolutely no latency. Since 802.11g wireless equipment is ***half duplex*** (that is, it only transmits or receives, never both at once) you should accordingly double the required throughput, for a total of **10 Mbps**. Your wireless links must provide that capacity every second, or conversations will lag.

Since all of your users are unlikely to use the connection at precisely the same moment, it is common practice to ***oversubscribe*** available throughput by some factor (that is, allow more users than the maximum available bandwidth can support). Oversubscribing by a factor of 2 to 5 is quite common. In all likelihood, you will oversubscribe by some amount when building your network infrastructure. By carefully monitoring throughput throughout your network, you will be able to plan when to upgrade various parts of the network, and how much additional resources will be needed.

Expect that no matter how much capacity you supply, your users will eventually find applications that will use it all. As we'll see at the end of this chapter, using bandwidth shaping techniques can help mitigate some latency problems. By using bandwidth shaping, web caching, and other techniques, you can significantly reduce latency and increase overall network throughput.

To get a feeling for the lag felt on very slow connections, the ICTP has put together a bandwidth simulator. It will simultaneously download a web page at full speed and at a reduced rate that you choose. This demonstration gives you an immediate understanding of how low throughput and high latency reduce the usefulness of the Internet as a communications tool. It is available at *http://wireless.ictp.trieste.it/simulator/*

Link planning

A basic communication system consists of two radios, each with its associated antenna, the two being separated by the path to be covered. In order to have a communication between the two, the radios require a certain minimum signal to be collected by the antennas and presented to their input socket. Determining if the link is feasible is a process called ***link budget*** calculation. Whether or not signals can be passed between the radios depends on the quality of the equipment being used and on the diminishment of the signal due to distance, called ***path loss***.

Calculating the link budget

The power available in an 802.11 system can be characterized by the following factors:

- **Transmit Power**. It is expressed in milliwatts or in dBm. Transmit Power ranges from 30mW to 200mW or more. TX power is often dependent on the transmission rate. The TX power of a given device should be specified in the literature provided by the manufacturer, but can sometimes be difficult to find. Online databases such as the one provided by SeattleWireless (*http://www.seattlewireless.net/HardwareComparison*) may help.
- **Antenna Gain**. Antennas are passive devices that create the effect of amplification by virtue of their physical shape. Antennas have the same characteristics when receiving and transmitting. So a 12 dBi antenna is simply

a 12 dBi antenna, without specifying if it is in transmission or reception mode. Parabolic antennas have a gain of 19-24 dBi, omnidirectional antennas have 5-12 dBi, sectorial antennas have roughly a 12-15 dBi gain.

- **Minimum Received Signal Level,** or simply, the sensitivity of the receiver. The minimum RSL is always expressed as a negative dBm (- dBm) and is the lowest power of signal the radio can distinguish. The minimum RSL is dependent upon rate, and as a general rule the lowest rate (1 Mbps) has the greatest sensitivity. The minimum will be typically in the range of -75 to -95 dBm. Like TX power, the RSL specifications should be provided by the manufacturer of the equipment.
- **Cable Losses**. Some of the signal's energy is lost in the cables, the connectors and other devices, going from the radios to the antennas. The loss depends on the type of cable used and on its length. Signal loss for short coaxial cables including connectors is quite low, in the range of 2-3 dB. It is better to have cables as short as possible.

When calculating the path loss, several effects must be considered. One has to take into account the ***free space loss***, ***attenuation*** and ***scattering***. Signal power is diminished by geometric spreading of the wavefront, commonly known as free space loss. Ignoring everything else, the further away the two radios, the smaller the received signal is due to free space loss. This is independent from the environment, depending only on the distance. This loss happens because the radiated signal energy expands as a function of the distance from the transmitter.

Using decibels to express the loss and using 2.45 GHz as the signal frequency, the equation for the free space loss is

$$L_{fsl} = 40 + 20*\log(r)$$

where L_{fsl} is expressed in dB and r is the distance between the transmitter and receiver, in meters.

The second contribution to the path loss is given by attenuation. This takes place as some of the signal power is absorbed when the wave passes through solid objects such as trees, walls, windows and floors of buildings. Attenuation can vary greatly depending upon the structure of the object the signal is passing through, and it is very difficult to quantify. The most convenient way to express its contribution to the total loss is by adding an "allowed loss" to the free space. For example, experience shows that trees add 10 to 20 dB of loss per tree in the direct path, while walls contribute 10 to 15 dB depending upon the construction.

Along the link path, the RF energy leaves the transmitting antenna and energy spreads out. Some of the RF energy reaches the receiving antenna directly, while some bounces off the ground. Part of the RF energy which bounces off the ground reaches the receiving antenna. Since the reflected signal has a longer way to travel, it arrives at the receiving antenna later than the direct signal. This effect is called ***multipath***, or signal dispersion. In some cases reflected signals add together and cause no problem. When they add together out of phase, the received signal is almost worthless. In some cases, the signal at the receiving antenna can be zeroed by the reflected signals. This is known as extreme fading, or ***nulling***. There is a simple technique that is used to deal with multipath, called ***antenna diversity***. It consists of adding a second antenna to the radio. Multipath is in fact a very location-specific phenomenon. If two signals add out of phase at one location, they will not add destructively at a second, nearby location. If there are two antennas, at least one of them should be able to receive a usable signal, even if the other is receiving a distorted one. In commercial devices, antenna switching diversity is used: there are multiple antennas on multiple inputs, with a single receiver. The signal is thus received through only one antenna at a time. When transmitting, the radio uses the antenna last used for reception. The distortion given by multipath degrades the ability of the receiver to recover the signal in a manner much like signal loss. A simple way of applying the effects of scattering in the calculation of the path loss is to change the exponent of the distance factor of the free space loss formula. The exponent tends to increase with the range in an environment with a lot of scattering. An exponent of 3 can be used in an outdoor environment with trees, while one of 4 can be used for an indoor environment.

When free space loss, attenuation, and scattering are combined, the path loss is:

```
L(dB) = 40 + 10*n*log(r) + L(allowed)
```

For a rough estimate of the link feasibility, one can evaluate just the free space loss. The environment can bring further signal loss, and should be considered for an exact evaluation of the link. The environment is in fact a very important factor, and should never be neglected.

To evaluate if a link is feasible, one must know the characteristics of the equipment being used and evaluate the path loss. Note that when performing this calculation, you should only add the TX power of one side of the link. If you are using different radios on either side of the link, you should calculate the path loss twice, once for each direction (using the appropriate TX power for each calculation). Adding up all the gains and subtracting all the losses gives

```
      TX Power Radio 1
    + Antenna Gain Radio 1
    - Cable Losses Radio 1
    + Antenna Gain Radio 2
    - Cable Losses Radio 2
-------------------------------
        = Total Gain
```

Subtracting the Path Loss from the Total Gain:

```
                    Total Gain
                  - Path Loss
                 ───────────────
         = Signal Level at one side of the link
```

If the resulting signal level is greater than the minimum received signal level, then the link is feasible! The received signal is powerful enough for the radios to use it. Remember that the minimum RSL is always expressed as a negative dBm, so -56 dBm is greater than -70 dBm. On a given path, the variation in path loss over a period of time can be large, so a certain margin (difference between the signal level and the minimum received signal level) should be considered. This margin is the amount of signal above the sensitivity of radio that should be received in order to ensure a stable, high quality radio link during bad weather and other atmospheric disturbances. A margin of 10 to 15 dB is fine. To give some space for attenuation and multipath in the received radio signal, a margin of 20dB should be safe enough.

Once you have calculated the link budget in one direction, repeat the calculation for the other direction. Substitute the transmit power for that of the second radio, and compare the result against the minimum received signal level of the first radio.

Example link budget calculation

As an example, we want to estimate the feasibility of a 5 km link, with one access point and one client radio. The access point is connected to an omnidirectional antenna with 10 dBi gain, while the client is connected to a sectorial antenna with 14 dBi gain. The transmitting power of the AP is 100mW (or 20 dBm) and its sensitivity is -89 dBm. The transmitting power of the client is 30mW (or 15 dBm) and its sensitivity is -82 dBm. The cables are short, with a loss of 2dB at each side.

Adding up all the gains and subtracting all the losses for the AP to client link gives:

```
  20 dBm (TX Power Radio 1)
+ 10 dBi (Antenna Gain Radio 1)
-  2 dB  (Cable Losses Radio 1)
+ 14 dBi (Antenna Gain Radio 2)
-  2 dB  (Cable Losses Radio 2)
──────────
  40 dB = Total Gain
```

The path loss for a 5 km link, considering only the free space loss is: