Building a Wireless Community Network in the Netherlands

Rudi van Drunen
WirelessLeiden Foundation
Leiden, the Netherlands
rudi@wirelessleiden.nl
http://www.wirelessleiden.nl

Dirk-Willem van Gulik
WirelessLeiden Foundation
dirkx@wirelessleiden.nl

Jasper Koolhaas
WirelessLeiden Foundation
jasper@wirelessleiden.nl

Huub Schuurmans
WirelessLeiden Foundation
huub@wirelessleiden.nl

Marten Vijn
WirelessLeiden Foundation
marten@wirelessleiden.nl

Abstract

With the development of low cost hardware based on IEEE 802.11b, wireless networks are an emerging technology. Using these wireless techniques outdoors it is possible to build a community network not dependent on any provider. In the Netherlands such a network is being set up in and around Leiden. Using low cost network interfaces, home-built antennas and open source software the volunteers of the Wireless Leiden Foundation were able to lay out an infrastructure for the inhabitants of Leiden at a very low cost. All kinds of applications (for-profit and not-for-profit) are using this wireless network.

1 Introduction

Current computer networks generally rely on a permanent, fixed and largely wired infrastructure which is owned and often operated by large entities such as telecom operators. A relatively new and emerging technology is wireless Ethernet, or wireless networking using the IEEE 802.11 standard. This standard encompasses the lower layers of the OSI model for transport of data as Ethernet frames using a spread spectrum based radio link.

This opens the possibility of building a network without having the problems and the cost associated with putting some sort of physical transmission medium in the ground. Instead, antennas can be used to send and receive the data using radio waves through free air.

Because of the relative simplicity of the currently available commodity hardware that uses 802.11 technology, it is relatively easy to build a local wireless community network in a town. Using this network people can share resources with each other. Examples are sharing sound or video files with the local museums, or having data (text, images, video, audio files) provided by the local government on-line. Furthermore, the network can connect to the Internet providing a low cost way of crossing the last mile to the user at high bandwidth.

In Leiden, the Netherlands, a foundation has been established by a number of knowledgeable volunteers with the intention to build a network operated and owned by a community of users, not by big entities such as telcos or Internet Service Providers (ISPs). This essentially free network infrastructure can be used by anyone present in the service area for running his or her own application. On the client platform only an industry standard IEEE 802.11b interface and a probably small antenna is needed. Usage of the infrastructure is not ”another monthly bill”, but will be free, after a one time up-front investment in equipment.

2 Method

2.1 Introduction

The wireless community network built has to meet a number of requirements to be successful. First of all it has to be as “open” as possible to the users and to the developers. Being “open” enables anyone within the community to actively use the network and participate in the building thereof. Another constraint is that the network should be both reliable and low-cost. These constraints are met in this design using commercial off the shelf (COTS) and home built (low cost) hardware (network boards, antennas and PC hardware) components and Open Source software (such as Linux, FreeBSD and other packages well known to the Free Software community).

2.2 Technologies

A number of different technologies are available to build wireless computer networks. Most commercial solutions are proprietary to certain vendors or do not use low cost hardware.

The IEEE 802.11 standard (called WiFi in the commercial world) allows users to network their machines using radio technology. Different sub-standards have been formed specifying the bandwidth or radio frequency the networks operate on. The standard defines a number of operation modes which allow for ad-hoc, point to point and point to multi point networking. Though the standard calls for two transmission technology standards: Direct Spread Spectrum (DSS) and Frequency Hopping Spread Spectrum (FHSS), the market has by and large standardized on DSS for indoor and outdoor point to multi-point use. FHSS, which is more robust against certain types of interference and densely packed endpoints, is currently only seen on long point to point connections where interference and frequency allocations are at a premium, for example, at an aggregation point.

In this project we have chosen the 802.11b as primary standard mostly because of the availability of equipment, open source drivers and the cost of the hardware. IEEE 802.11b is a DSS radio technology supporting link speeds of 1,2,5.5 and 11 Mbit/s. The standard uses the 2.4 GHz frequency band and is initially designed for home and office use but with special measures (antennas) it is also applicable to crossing longer distances outdoors (up to a maximum of approximately 15 Km line of sight). [WirelessNet, WirelessComm]

2.3 Topology

The network we are building and operating is targeted to a coverage area of $25Km^2$, which is the complete city of Leiden (The Netherlands) and its surroundings. There are no hills in the target area. Other natural obstacles such as large forests are also absent. It is a small old town center with some moderately high buildings and some 10-15 story appartment buildings in the suburbs. These appartment flats are surrounded by small houses (max. height 3 floors).

The total number of people living in this area is approximately 160,000. In this area we want to provide outdoor coverage. For using the network indoors, a small antenna connected to the client computer has to be sufficient. The antenna should preferably have a line of sight to the nearest network access point. This is required as the maximum distance between the client and the network access point (a network node) is limited due to the national legislation implementing European (EU / ERC) regulations (restricting maximum radio frequency output power and restricting antenna gain). Combining this knowledge with the anticipated traffic and bandwidth needs, it is evident that we need multiple nodes distributed over the coverage area. The nodes themselves have to be interconnected. A plot of the radio coverage of the current set-up (the historical center of Leiden ($10Km^2$)) is shown in figure 1. To provide full outdoor coverage of the target area an estimate of 25 network nodes will be needed. The interconnection

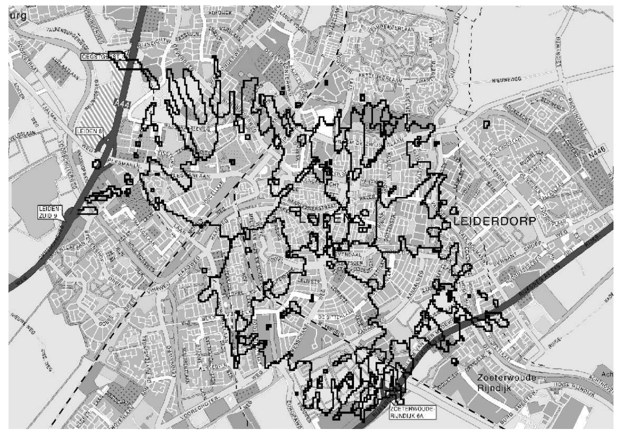

Figure 1: Current Wireless Leiden radio coverage plot. The total area shown is approx. $20Km^2$. Overlayed in black are contours of equal field strength of the areas in which the Wireless Leiden network is available when using a standard wireless network card connected to a $7dBi$ gain antenna located outdoors. (data courtesy of `www.wirelessdesign.nl`*)*

of the nodes also uses wireless links, making the network completely independent of the local (wired) infrastructure and thus very cost-effective and without significant monthly or other regularly repeating costs.

A mesh between the nodes will be formed, as each node is connected to the other nodes by at least 2 different (wireless) connections. With this approach, adding extra nodes to the network will add redundant paths and will therefore also increase the total available bandwidth. The topology as seen from the user is comparable to cellular telephony. Cells for users are created. In these cells the users share the total available bandwidth of an access point. The cells themselves are interconnected by point-to-point wireless connections. These connections form the backbone of the network.

2.4 Radio Planning

In the Netherlands there are 13 radio channels allowed in the 2.4 GHz frequency band to be used for radio local area networks. Due to the DSS technology a used channel does not occupy a single discrete frequency but is a distribution of power in a 22 MHz wide frequency interval. In figure 2 it is shown that there are 3 completely separate channels available. Combining this knowledge with the topology and the goal of providing coverage in a fairly big area poses a challenging problem [Beckmann]. This problem is solved by careful selection of the channels to be used. For example, at the cost of a small increase in noise but no decrease in bandwidth, it is possible to use 4 channels on the same site. (e.g. channels 1,5,9 and 13). Of course the channel use is very much dependent on the local situation at the radio-level. Other measures to prevent cross-interference of different radio-links include the use of (directional) antennas and the location and polarization of the antennas. In the

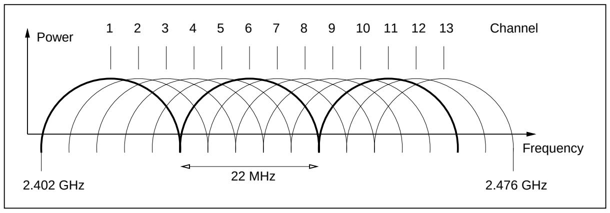

Figure 2: The available channels in the Netherlands.

Netherlands the 2.4 GHz frequency band is reserved for a number of licensed and unlicensed applications ranging from microwaves to video-links, vehicle identification systems and radio amateur (ham) use. The use of WiFi in this so-called ISM (Industrial, Scientific, Medical) band is allowed unlicensed, providing the above mentioned channel restrictions are observed and the effective output power of the antenna (EIRP) does not exceed 20 dBm (= 100 mW).

Providing coverage in the target area requires careful radio planning. Often chaotic behaviour is seen (i.e. small changes in initial conditions can turn out to be critical). Techniques such as hexagon planning, as used in the cellular telephone world, are used to optimize the coverage and spectrum (channel) efficiency. A homogeneous network is created by placing every site on a predetermined grid point such that every cell covers half of the inter cell distance. The physical cell boundaries are determined by the local situation and the antennas used. The cell boundaries are not symmetrical for up and down link due to local and receiver noise. This also needs to be taken in account to avoid areas without coverage (i.e. downlink possible, but no uplink possible). Using commercial Hata-Okumura [Ho] model based, radio-planning software we can simulate the propagation of the signal and optimize the location of the different sites. These advanced planning tools (e.g. CellCad (www.lcc.com) or PathLoss (www.pathloss.com)) are not comparable to those available in Open Source. Alternatives such as Radio Mobile (www.cplus.org/rmw) lack important data on the environment such as building characteristics. As most of the commercial tools require a large computing infrastructure and various subscriptions to maps and other GIS data, like building heights and absorbency or reflection characteristics, it is not feasible to actually run the simulations ourselves. A sponsor has access to this software (CellCad) and runs the required simulations.

Nevertheless, a site survey is always needed to actually measure the noise generated by other radio sources on-site and check the signal strength of the already running nodes. Due to the high absorbency of the radio signals by (for example) trees in the line of sight, local field measurements are essential in planning a node. A site survey is done using the Kismet [Kismet] or dstumbler [Dstumbler] software running on a Linux or FreeBSD laptop with a small panel or directional antenna and wireless network interface.

As a subproject within the community a compact automatic survey tool is currently under design. This tool will collect signal strength and noise figures from existing Wireless Leiden nodes as well as signals from access points or other devices in the 2.4 GHz band. The hardware device based on an embedded system connected to a GPS receiver can be carried through the city on various utility vehicles to effectively cover the complete area. Once completed, the software environment will be comparable to the software setup of a node with some extra tools to log the data. When available, the logging data will then be plotted as an overlay on the simulation maps and made available to the users wanting to connect to the network as a guideline for aiming their antennas.

2.5 Antennas

The standard antennas that come with wireless interface cards are designed for office use. To use this network interface in a long distance outdoor connection as we do, different (external) antennas are needed.

2.5.1 Antenna types

Good access point antennas are omni directional or sector types. An omni directional antenna has the advantage of quickly providing a large coverage area. The usage of sector antennas makes it possible to split the area around a node into several different independent parts. The main advantages of this approach are that while providing a stronger signal into the sectors, the access point node is able to handle more users (i.e. a larger total available bandwidth for that cell). A disadvantage of using sector antennas is that more wireless interfaces are needed and that interference can be a problem.

A point-to-point connection needs an antenna which directs all the radio frequency (RF) energy into one direction. The limiting factor for this antenna is mainly its size. High gain antennas reduce the chance of interference to other nodes or to the other antennas of the same node but are often quite large in size.

The key to a successful antenna setup for a given node location is in evaluating the simulation results together with the site survey details. After that, the cost factor is not to be neglected in the selection process. In some cases a home-made antenna can have a better price / performance ratio than a commercial one.

Clients need a directional antenna to connect to a node. There is no need for clients to be connected to more than one node at the same time. The main specification for the client antenna is the minimum gain that is needed to obtain a signal strong enough to make a good connection to an access point. Based on this gain, several different types of antennas are available. Other factors that determine the best antenna are the physical size, appearance and the cost of the antenna.

Simple antennas for use with access points [Omni], clients [Quad] or point-to-point connections [Yagi] can be made at home, keeping the price at the lowest possible level. Almost anything can be turned into an antenna. It has been shown that it is not difficult to make a WiFi antenna [Pringle]. The Wireless Leiden website has a large collection of antenna designs provided by users.

The layout of the Wireless Leiden network is designed on a typical client antenna gain of 7 dBi. Antennas with this gain are cheap and / or easy to construct.

2.5.2 Lightning protection

Lightning protection is only applied at the building level, like connection of the antenna pole to the existing lightning protection system. This means that there are no special surge protectors used in the antenna cables. The chance that lightning strikes in the Netherlands is so small compared to the cost of the arrestors and the problems associated with them (like insertion loss and the protection level they provide) that the risk of getting the (low cost) equipment damaged by lightning is acceptable.

2.6 IP space Planning

The network uses TCP/IP as the transport layer. Therefore, every active element in the network should have an address. Using a private IP version 4 range, enough addresses will be available. Using IP version 6 will be a future enhancement and is not yet fully implemented. So, as the IP network grows there is a need for not only planning the radio frequency space, but also planning the IP space. The IP range is assigned on the basis of the different postal (zip) code regions in the coverage area, combined with the population density and average income per head. Client machines are provided with the correct network information for the area they are in by a local DHCP server running on the (first) node (the access point) they are connecting to. The information provided by the DHCP server is the IP address to use, the netmask, the nameserver addresses and the default gateway.

Any two nodes in the backbone that are connected to one another by a point-to-point link use a /30 IP range (i.e. 4 IP addresses). This ensures that `traceroute` shows the logical network topology (no physical hops are missing because all packets must rise to the IP level, none are routed by MAC address). IP space that is used for numbering an interlink from node A to node B is assigned from a separate IP range; it should belong to neither geographical area A or B, as it is part of the backbone.

2.6.1 Internal routing

Having the IP space mapped to the physical map, a number of areas are created that contain different subnetworks. Between these areas there are different network links. This approach creates multiple routes from a source to a destination. It is no longer possible to manually manage the routing tables in all nodes that have more than one interconnect. Using the Open Shortest Path First (OSPF) routing protocol minimizes the routing configuration of a node. As soon as the appropriate interlinks are made (or broken) on the IP level, the OSPF processes start to exchange information about their knowledge of the network topology. This also allows us to add or remove nodes from the network without any routing reconfiguration.

Using the hop count and cost factors on various routes, the system will select the route with the lowest total cost. If any link fails, it is detected by the OSPF daemons and all routing tables in the network will be updated to reflect the new situation. Currently, routing is done solely on hop count. All cost factors are the same.

By using OSPF routing and different interconnects on one node, the reliability of the network on the IP level is greatly enhanced at a negligible cost of processor overhead and network bandwidth.

For the implementation of the OSPF protocol the Zebra [Zebra, Routing] package is used. Zebra is an Open Source package which provides a number of different routing protocols by various daemons, all controlled by a master daemon. Currently OSPFv2 routing is used throughout the backbone network because the protocol is fit to the topology and does not impose a large overhead.

2.6.2 External routing

Using a private IP space on the network will introduce new issues on routing when connecting the network to the Internet using multiple ISP connections. Some nodes have an external default gateway that is injected into OSPF. These routes are propagated through the network so that each node has one or more routes to the Internet.

Currently there is no smart algorithm to decide which one to use (or to use a particular route to the Internet for a particular user). Several options are being investigated: tunnels (e.g. PPTP), source routing and the use of proxy servers.

Also, discussions with RIPE are planned to address the possibility of obtaining a block of IPv4 numbers together with an AS number. This will ease some of the ISP connection problems.

2.7 Site allocation

Once a site is designated by using the planning procedure, it might be a tedious task to get permission from the different building-owners to have the antennas and equipment installed. Because of the not-for-profit and volunteer-driven nature of this project, it might be difficult to explain to the building owner that we are not able to pay the same amount of money as a cellular phone provider. A cellular provider will easily pay up to Euro 10,000 each month for a location. We, as Wireless Leiden affiliated volunteers, have organized ourselves in an official charitable trust (foundation) with a statute, and this foundation has managed to generate some positive publicity. Being an official foundation with some media exposure and a good story – a free, fast community network – has been helpful in gaining rooftop access free of charge. Having access to people knowledgeable on building and safety issues ensures a setup according to all local building regulations, like measures against lightning strikes.

Locations that are important to the community, like schools or the Town Hall, are target locations to set up nodes. Also cooperating commercial enterprises that want to provide services on the network or want to make use of the network for private communications (e.g. between different branch offices) are quite willing to invest in the equipment and time to set up and maintain a node.

2.8 Setting up a Node

A typical network node setup consists of a number of antennas and a computer system. We use 2 or more directional antennas to connect to other backbone nodes and one omni-directional antenna for local client access (See figure 3). A PC or other system provides the routing and access-point functionality. A typical node setup can be found in figure 4. The home-built and partly commercial antennas are connected to a computing platform that can, but does not need to be connected to the local network at the site using wired Ethernet. Setting up a network node requires some real hard hardware work to get the antennas lined up and affixed to the building. As we are using directional and polarized antennas to prevent interference, the alignment of the antennas is fairly critical. Once the antennas are set up, connection to the node

Figure 3: Antennas as mounted on a building (2 directional antennas and one omnidirectional antenna).

machine is done using low loss coaxial cable. As keeping cable losses to a minimum is important, the length of the cable should be minimized as each meter of antenna cable introduces a loss of approximately 0.25 dB. The node machine should be as close as possible to the antennas. When using a small embedded system (see section 2.9) the node computer can even be mounted outside on the antenna pole. In this case, special measures have to be taken to weatherproof the setup (e.g. like preventing condensation). This setup can use Power over Ethernet (PoE) [PowerF] [PowerS]. A single UTP network cable (where the length is not an issue provided that it is less than 100 meters) connects the antenna setup to the power supply (indoors) and the on-site local wired Ethernet if needed.

Before and after the installation of the node hardware a number of basic alignment, throughput, and reliability tests have to be run. These tests are used to be sure that the links in the set-up will operate as expected. The expectation values for these tests are derived from the site survey and simulation results as described in section 2.4. The alignment test comprises of monitoring the signal strength and noise figure when a backbone link is established. By physically varying the antenna direction the signal as seen by the driver software on the wireless interface should be maximized, while the reported noise is minimized. Throughput and reliability is tested by transferring large chunks of data from one node to its neighbor and observing the packet loss and transfer rate.

Often, once the node is set up it is quite difficult to physically access the machine and the antennas because they mostly are located at remote locations and in buildings with complex access procedures. A typical test for the reliability is to copy some video data (often comprised of large files) through the node. This test differs from the test mentioned above where a single wireless link is tested, here the complete (routing) functionality of a node is tested. A test protocol is used to assure the repeatability of the test procedures.

An external hardware watchdog device is used to ensure a reboot of the system when some part of the software crashes. These watchdog devices are home built and very simple. The watchdog watches the serial port of the node computer for a “hello” string. This string has to be sent every minute (using a small program or even a script). If this string is not received within 90 seconds of the previous string, the watchdog circuit will cut the power to the setup and will turn it on after a minute. Using either journaling file systems or RAM based file systems minimizes problems caused by a `fsck` operation after the setup has been powercycled.

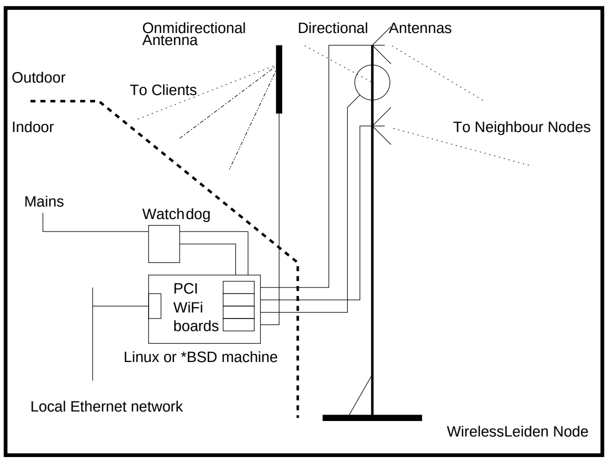

Figure 4: A typical WirelessLeiden Node setup.

The software environment on the machines is a standard Open Source free UNIX (currently FreeBSD) distribution stripped down to fit in a minimal hardware configuration. The (kernel) device drivers that are used to control the wireless network cards and some network operations and management utilities are added (ref. section 2.10).

Using a standard off-the-shelf Open Source operating system enables us to implement a node quite fast while at the same time keeping the flexibility of changing things on all levels when the network grows and / or the technology changes. Another aspect is the large and diverse knowledge-base available within the development and engineering group. Also, the Open Source development model guarantees a fast turn around time in fixing bugs or evaluating features. Last but not least, in a not-for-profit organization working from donations the initial cost of the software is of major importance.

2.9 Hardware Platform

Each node needs a piece of hardware that handles the wireless signal and provides IP routing. Currently Intel Pentium I class PC machines with a PCI bus are used. The wireless interface cards are standard cards using the Intersil PRISM series chip set and a PCI bus, such as the Linksys WMP11 or Compaq WL200. The PCs used are often found as surplus machines or are donated. Due to the different hardware configurations of the machines it is difficult to standardize the hardware and improve the reliability. To overcome these problems different options are evaluated. Commercial wireless nodes (e.g. the Nokia rooftop systems) are not an option as they are often not “open” and far too expensive. Often they use proprietary adaptations from the 802.11 standard and are not accessible to the developers in the community to develop the hard- and software further.

Embedded systems based on i486 or better processors with at least one Ethernet interface and at least 2 slots for connecting wireless network cards are a good alternative. When using an Intel i486 or better processor most of all software developments can be reused.

Embedded boards are available in various different flavors. At this moment we are testing the Soekris Engineering embedded boards. These boards use an i486 compatible processor (AMD Elan SC 520) running at 133 MHz, 64 Mbytes of RAM, a Compact Flash slot, a mini PCI slot and two PCMCIA slots. Figure 5 shows a block diagram of this board when used as wireless network node. For the PCMCIA and Mini PCI solution wireless cards based on the PRISM 2 reference design (e.g. SENAO) are used. This adds software compatibility with the PRISM 2.5 based PCI cards that are used in the PC design. The Soekris board is a commercial product and will run either Linux or a flavor of BSD without problems, freeing the developments in the node software from the hardware developments. Other industrial Intel based boards often use various bus standards (e.g. PC 104), which add additional costs in connecting (standard) cards for wireless networking. Self-designing a board level solution is not an option due to lack of resources and money. At a price higher than the price of the donated PC machines these embedded systems will add reliability and standardization to the node base. The intention is to migrate the core network nodes from PC’s to embedded systems.

2.10 Software Platform

The main choice of Operating system for a Wireless Leiden Node is FreeBSD. In the development of the Wireless Leiden node a number of problems were encountered which have resulted in the current FreeBSD installation.

Initial Linux implementations of the node functionality resulted in problems with the performance and stability of the systems. Partly they were due to the older hardware used: the PCI controller not allowing to assign

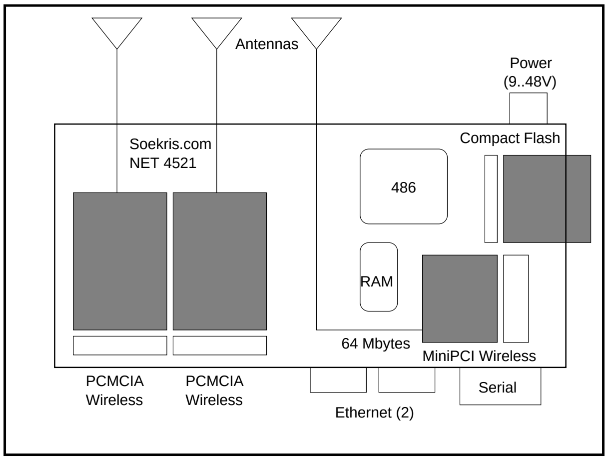

Figure 5: A Soekris based node.

different interrupts to the wireless cards and the interrupt handling in the Linux kernel. Other problems were connected with using multiple wireless cards with the PRISM hostap driver [HostAP]. We could have used other drivers (e.g. wlan-ng), but the hostap driver was the only one implementing an user software (i.e. not firmware) driven access point functionality. Another major issue was the number of different Linux distributions around and the rapid change thereof.

Using FreeBSD solved some of the Linux-related problems, like interrupt handling on heavily loaded systems, while adding features such as having one build tree, a steady release and distribution scheme. Furthermore, a single stable PRISM 2 support stack and access to a more mature configuration system makes the system more suitable for mass deployment. Another benefit of using FreeBSD is the inclusion of several wireless network card firmware cutoff checks which will restrict the functionality accessible from userland when a too old version of the firmware is detected.

For the Compaq WL200 card we did have to fix a specific routing bug which was introduced in between FreeBSD 4.6 and 4.7 as part of the cardbus / newbus project. This has since been picked up by CURRENT and will be in the next release of FreeBSD. Prior to FreeBSD 5.0 release some issues where also found and reported with regard to the `wicontrol` and `ifconfig` settings not being propagated properly. These have since been fixed.

It should be noted that each of the different wireless stacks on Linux has at least one or two elements which are vastly superior to the rather dated PRISM 2 support in FreeBSD; but unfortunately each stack also exhibited showstoppers ranging from erroneous detection of cards to kernel panics. It is not unlikely that a maturing or crystallizing, of one of the wireless options of Linux will make Linux a viable choice in the near future. This, combined for example with a good AODV routing protocol implementation (see section 3.4), may again cause us to switch platforms. Another, unplanned, benefit of the BSD stack is the rather mature diskless boot system which is part of the standard distribution; which is an excellent starting point for automated installers and systems operating from a read-only storage device.

As FreeBSD has just a single version number and release path; the entire 'build' procedure of the master machine from which all nodes are automatically built entails less than 5 kbytes of script; just short of one page. The 'definition' of a node, including kernel configuration, routing, antenna frequencies etc.; i.e. all that needs to be defined given a specific version of FreeBSD, currently runs at around 12 kbytes of data.

Tools such as 'mergemaster' available in FreeBSD and the rather elaborate documentation make upgrades in the field, even across versions, reliable and predictable. Likewise extensive use is made of the 'ports' collection to manage the versioning and updating of "third party" packages such as the ISC-Bind, Zebra and a few others.

2.11 Version control

Deploying a large number of network nodes consistently and reliably, depends for a large part on the version control of the different configurations and their configuration files. In the Wireless Leiden project Subversion is used together with a system called Genesis.

Subversion [svn] is a new breed of CTM; akin to SCCS or CVS - but more suited to Open Source code management as it does not require extensive Unix account management and can be configured to use HTTP as its communication protocol. Currently, Subversion is hosted on a remote machine and is connected to the Internet through a DAV module and an Apache 2.0 web server. Access controls are intentionally light and commit access is virtually for the asking. Genesis is a web based homebuilt configuration file generator written in PERL [Perl5].

Due to the remote installation of both the Subversion and Genesis installation and configuration of a Node machine relies on a live Internet connection. However, it should be noted that both systems could be moved to a more 'local' environment close by or could be replicated. So far this has not been necessary.

2.12 Building the Nodes

Apart from doing the real hard hardware setup of a node, the software configuration on the system has to be set up. To automate the software installation and configuration, the Wireless Leiden Node Factory was developed. A non trivial solution was needed because node configurations differ more than just by their IP address, due to different hardware configurations and platforms being used.

2.12.1 The Node Factory setup

The node factory consists of a central machine which contains the distribution(s) to be installed on the node along with a diskless boot environment. The latter is used during the initial boot as a staging ground for the actual install. A new machine is connected to the central machine, booted from floppy using 'etherboot' which subsequently launches into a diskless FreeBSD install. All hardware will be detected, a reformat of the hard disk is done and the node software is installed and configured.

The install master server is a machine with two network cards and a single wireless network card. At present a Pentium II class machine with 256 Mb memory and 4 Gb harddisk is used.

The first network card connects to the Internet through a DSL connection. This is needed to fetch source, binaries and configuration files. The second network card is connected to the machine to be installed and configured. The wireless network card is used for the automated tests during the installation.

As the first network interface on the server provides Internet connectivity, the second nic offers a NFS, an etherboot and a PXEboot environment to the client to be installed. To enable NFS standard FreeBSD rc.conf and export settings are used; etherboot floppies are created using the default version in /usr/ports and the PXEboot environment relies on the DHCP daemon using a standard configuration from the handbook.

In order to allow a newly installed node to access the Internet routing is enabled between the nics by a `gateway_enable` setting in rc.conf.

The software is retrieved from the Net by downloading a FreeBSD image. After installing this on the server software was upgraded from 4.7 to the CURRENT branch of FreeBSD and frozen. The CURRENT branch was selected over the STABLE branch at that time to make sure the latest wireless drivers and patches thereof were included. However, testing on basic functionality was needed to validate the distribution for production use.

Using the `make world` mechanism with a different DESTDIR, a complete separate FreeBSD tree is built. Added to this tree at the root is a special diskless install kernel and, in the default /boot/kernel location, a stripped down run time kernel. Furthermore, two additional scripts are added in the root which will control the actual install. An extra rc.local script is added which will run as the final step during the first diskless boot. The latter causes the disk to be partitioned, formatted and the install to start.

The next step is making a bootfloppy for each of the network cards with the right version of etherboot. On Soekris embedded systems the native PXE boot environment is used instead.

2.12.2 Building a Node system

Preparation for building a new nodes means collecting a Pentium (or better) CPU some RAM, two or three wireless lan cards and an Ethernet card. A 500 Mb harddisk (or more) will do fine. The BIOS of the PC is configured to boot from floppy disk, or in the case of the Soekris, to boot from the network using PXE. Besides the BIOS on the Soekris machines, we have not tried any BIOS-es or Ethernet cards which supported PXE directly but have relied on a few different etherboot floppies instead.

In the first stage the new machine boots from floppy disk to initiate etherboot. DHCP will assign an IP address and provide the parameters which allow the node to mount the NFS file system. After loading the kernel, a script prepares the harddisk by making it bootable, defines its partitions and formats file systems. This is followed by copying all the files from the server to its own harddisk. After a reboot the new machine has it own system in order to run.

After the reboot installation of some packages such as a dhcp-server, a nameserver, snmp etc. takes place. The installation is managed by a set of homebuilt scripts (see section 7). The reason to do so after a reboot rather than during the diskless boot is that during the initial diskless boot we do not want to rely on having sufficient memory available. The diskless system uses a memory based /var and /tmp overlay and therefore consumes quite some memory. The scripts are invoked by a state engine at the end of the boot procedure.

At the end of the procedure the new machine will show a list of hostnames. After entering one at the command line the machine will lookup its configuration on the Internet in the remote Genesis. The old files will be backed up and the new ones that define the specific node will be put in the right places.

Then approximately 15 minutes after the first (floppy) boot, the new machine is in operational state and ready to be deployed and located at site.

2.13 Design rationale

Using this Node Factory setup for building a network node it is very easy to setup and deploy wireless nodes, while ensuring a maximum of maintainability. The low-level configuration of each node is nearly identical and by using version control and the configuration database it is very easy to track down changes and keep these documented. A last important point is that using Open Source software the complete system will end up to be very cheap.

2.14 Security

Security is currently not applied on the network infrastructure level. Of course, all network nodes have appropriate security to secure the boxes themselves, but as an infrastructure provider, we have the rule of "security is the responsibility of the user". On the radio level we use a combination of narrow beams (directional antennas) to interconnect nodes together with Wired Equivalent Privacy (WEP) or even a WEP infrastructure with dynamic keying. As WEP provides no actual security [Borisov], the user of the infrastructure must be aware of the *in*security of the transported data, and use security on a higher level, for example by using IPsec tunnels over the existing infrastructure. Raising awareness of these problems and their solutions by the users is important. Right now the projects website addresses this in detail.

The nature of the transport layer adds an extra possibility of Denial of Service (DoS) attacks by "jamming" a connection on the radio level. This can happen when another device using the same frequency is operating in close vicinity of the node. Because we are operating concurrently with other users in the same frequency space this can be a problem. Adding redundant paths together with the appropriate routing protocols is the way to overcome the problem for the user.

2.15 Running a Network Node

The network node is highly self-contained and does not need frequent maintenance visits. All software and configuration maintenance, upgrading etc. is done from the network itself using the configurations in the repository. Software level reconfiguring of the node can be done on the fly to cope with the changing network architecture.

However, some on-site hardware maintenance may be needed. Typical issues to be considered are taking care of the fans in the power supply and coping with hardware failures such as hard disk problems. Furthermore, some hardware maintenance on the antennas is needed. Regular inspections of the state of the antennas and mounting hardware are required to ensure safe and reliable operation. Local building safety regulations may also require regular on-site inspections of the set-up.

The Network node can be connected to the users' (home) network in order to give the local user wired access to it. Therefore, on the Ethernet port DHCP is enabled.

Traffic and other operational data on the node is gathered using RRDB and RRDtool [RRDB] and sent to the central repository at regular intervals. Some examples of graphs used can be found in figure 6. For simple "health" management of the nodes the free version of Big Brother [BB] is used. Big Brother generates a number of "status lamps" on the network map available on

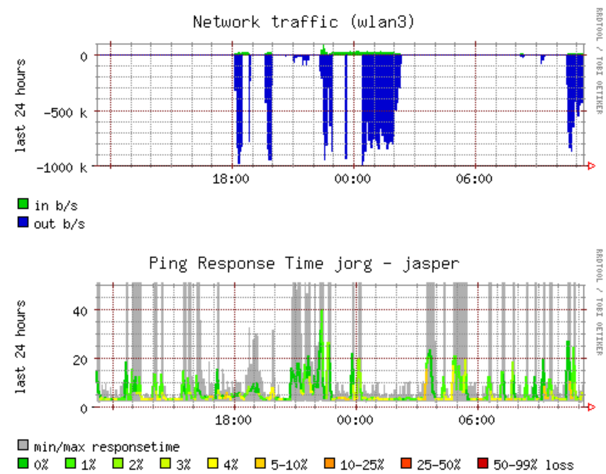

Figure 6: A link traffic and `ping` *latency graph made by RRDtool.*

the website. The combination of the graphs created by RRDtool together with the *ok-notok* information from Big Brother has shown to be a valuable tool to monitor the network. If the network and its traffic grows, other tools may be needed.

3 Results

Within approximately 1.5 years the network has grown form a single point-to-point connection to a completely usable network. Logging MAC address data shows that the network is used by approximately 300 different machines. On a day-to-day basis it is currently estimated that approximately 100 users are connected. With the introduction of (free, sponsored) Internet connectivity on the network this will without doubt grow. Although the learning curve was steep, the fully operational network is now gaining more acceptance as more applications become available.

3.1 Applications

During the first year of this project we have succeeded in building about 12 network nodes in the Leiden area. These nodes are interlinked with their neighbors by 802.11b links and have access points to accommodate users to connect wireless to the network. Applications currently in use are various VPN connections of enterprises giving their employees access to the company network, schools with different locations connected to each other, a video server and some gaming applications.

As of March 2003 a large Internet provider is sponsoring the Wireless Leiden Foundation with a number of high speed (8 Mbit/s) DSL connections. Without raising big routing and peering problems the first phase of integrating Internet connectivity to the Wireless Leiden network a number of proxy servers [Squid] and captive portals [NoCat] are being deployed to allow the clients to surf the Internet using http and https only. In the meantime, a more definitive network design that allows multiple ISPs to use the infrastructure is being developed.

Due to the achieved outdoor coverage of the network, applications that require mobile use of the network are possible. These applications are currently being developed by users of the network. We are investigating how these applications will use the network and what additional features they might need (e.g. roaming functionality and / or the use of MobileIP [MobileIP]).

3.2 Problems

Problems that are encountered during the start-up phase of the network can be divided into two groups. There are technical and non-technical problems to be solved.

Technical problems are seen on every level of the network stack. Starting with the physical level, it is difficult to plan the network using the different constraints like noise, limited channels available, natural and other obstacles. Due to this issue the configuration of the network is constantly evolving. Changing of antenna directions can be difficult once a node is set up.

On the higher levels we have encountered problems with the network drivers for the wireless boards we are using. These problems can seriously affect the throughput and reliability of the network. Here the use of open source software and the open source development model pays off. Problems can be solved within the group, or with help and information gained from the Internet. Also having a choice between different solutions to the problem helps.

On the IP level, having many point-to-point links and a number of applications results in quite a big puzzle to get everything configured correctly. Using a central configuration repository and the use of routing protocols helps, but some problems still remain to be solved. For example when connecting to the Internet (with a number of providers), routing and numbering issues needs to be solved. Also the latency in the network might pose a problem in some near real-time applications or applications that use streaming technology (e.g. live video).

Furthermore, the reliability of the hardware can be a problem. Because of cost issues, complete (used) PC machines are in use as network nodes. These machines are not as reliable as dedicated routers running on embedded systems without moving parts (like fans or harddisks). Minimizing the points of failure, testing, validating and proper maintenance helps a lot here. Ultimately, using embedded systems will ease this problem.

Non-technical problems include gaining acceptance within the town and the community. Without a broad acceptance and support of various organizations it is not possible to build this kind of a network in a not-for-profit fashion. Access to rooftops of high buildings etc. is essential. However, we do not believe that a for-profit organization has any chance of succeeding at all. The upfront costs of building a network without volunteers are very large. Possible revenue streams will be insufficient to recoup those large costs.

The other difficult task is to manage a large group of volunteers that are actually doing the work. With the growth of the network, also new people are joining to do the design, management and the building of the network and the nodes. With every new engineer another degree of (intellectual) freedom is added to the system. Managing this enormously large and diverse task force in an open manner can be difficult.

Also, we see that on the non-technical level the more religious ideas between e.g. the different operating systems or engineering solutions on a specific topic are a problema to cope with (like *BSD vs. Linux). Having a heterogeneous set-up with different operating systems will gain a broader acceptance within the group and probably builds a more robust network, but will be much more difficult to manage.

3.3 Experiences

Right now, the network is in full operation and the first applications are successfully being deployed. Due to the rather small group of technically skilled people the main focus of Wireless Leiden has been to set up nodes and provide coverage in the historical center of Leiden. In addition we have concentrated on knowledgeable individuals and professional organizations as first users, because they require less IT-assistance from the volunteers. As this is accomplished, the next step is extending the network to the suburbs and connecting the actual private users to the network.

Doing research in this environment is very attractive: the big advantage is that a testbed (i.e. an actual running network) is already deployed, so actual field testing of new technologies is relatively easy.

3.4 Future work

For the coming year a target is set to build and install at least one new network node every month to extend the coverage and increase reliability and bandwidth. The second target is to set up a structure to effectively help the individual users to connect to the network. A third target is getting more applications running.

New technologies are being tested to cope with the expected growth of the network. Upgrading the backbone links between network nodes to use 802.11a or 802.11g technology (max. 54 Mbit/s) [WCompare, IEEE] is in test.

Also in evaluation is the use of more complex and advanced technologies using mesh or ad-hoc networking [Hu, Maltz, Royer]. Imagine a network where all clients are also a node and every client is connected to the two or three closest other clients (close is defined here as the distance at which one is able to create a *good* (signal to noise ratio) connection at that particular moment in time), not to a central node in the neighborhood. The noise floor would remain at the same level as the network gets more dense while the number of possible parallel paths from A to B also grows with the number of clients. More clients lead to shorter paths which will result in less RF power needed to obtain the same connection quality. Each new participant brings along his or her own piece of network, bandwidth and noise-reduction.

Using these techniques, advanced routing architectures auch as AODV may be used. The Ad hoc On Demand Distance Vector (AODV) routing protocol [AODV, aodv-ietf] is an "on demand" routing algorithm, only creating routes when they are actually needed. AODV is loop-free and scales to large numbers of mobile nodes.

Another point of evaluation is the use of applications that require Quality of Service (QoS) facilities in the network such as IP telephony and wide band video streaming.

4 Related work

In different parts of the world wireless communities are developing. In the USA there are a number of leading initiatives [nyc, Seattlewireless, FreeNetworks], but also in Europe and Australia communities have been formed. The main difference between the Wireless Leiden network and a number of other wireless communities is that Wireless Leiden definitely not has a hobby-network kind of style and set-up. Due to the professionals affiliated with the Wireless Leiden Foundation and the partnerships with major (local) groups of potential users, hardware vendors, the university and content providers it is far beyond the concept of a number of people sharing their DSL connection using wireless technologies. The other main difference is that due to the acceptance by the community and therefore the possibility to set up nodes on a large number of non-individually owned buildings, all connections between the network nodes can be wireless. No wired connection is needed. Here the European (or Dutch) mentality of cooperation to achieve the best result may be an advantage.

Last but not least, as the infrastructure is based on open standards and Open Source software, it is freeing us from vendor lock-ins and is achieving the broadest possible range of applications while providing sustainability of the complete system.

5 Conclusion

With the use of relatively low cost technologies and Open Source software it is possible to build a wireless network which is used by the community. Technical problems do exist, but by usage of Open Source software they can easily be solved. The process of building a network using a loosely-coupled group of volunteers is not easy but bringing organization in the group when the network becomes more complex helps a lot. Having the back-up of the community and local enterprises also helps to gain momentum and visibility of the project, which in return speeds up the development and growth of the network.

6 Acknowledgments

The authors wish to thank all people affiliated with the Wireless Leiden foundation, The city of Leiden and the sponsors of the project. Without the dedication, professionalism and enthousiasm of these people and organizations this project would not be able to prove its successfulness. Special thanks go to the following people: Johan de Stigter of Gandalf and Evert Verduin of WirelessDesign for their technical input. Furthermore thanks go to Caroline Beijer, Brad Knowles and Dick van Velzen for their valuable comments. Special thanks also to the FreeNIX reviewers and to our shepherd, Guido van Rooij.

7 Availability

As the Wireless Leiden Foundation is an open community, all information, software and hardware that is being developed is free to use for everyone under the Gnu Public License. A WiKi website is available to effectively share this information. Unfortunately it is in Dutch, but as engineering language is international, with some effort the important technical info can be extracted. The website can be found at:

`http://www.WirelessLeiden.nl`

Direct links to the node software can be found here: `http://www.WirelessLeiden.nl/wcl/cgi-bin/moin.cgi/InstallLayers`
The subversion tree is available under the following link: `http://wleiden.webweaving.org:8080/svn/node-config/master/`

References

[AODV] AODV: a routing protocol for ad hoc networks, `http://w3.antd.nist.gov/wctg/aodv_kernel/`

[aodv-ietf] Ad hoc On-Demand Distance Vector (AODV) Routing draft-ietf, http://http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-13.txt

[BB] Big Brother: a network monitor, http://www.bb4.com/

[Bawug] BaWUG, The Bay Area Wireless Users group, http://www.bawug.org

[Beckmann] D. Beckmann, U. Killat, *Applied Frequency Planning for Cellular Radio Networks.*, Int. Journal of Electronics and Communications, 54-4 2000 pg. 211-217, 2000.

[Borisov] Nikita Borisov, Ian Goldberg and David Wagner, *Intercepting Mobile Communications: The Insecurity of 802.11.*, 7th Annual International Conference on Mobile Computing and Networking, 2001.

[Dstumbler] dstumbler: a wardriving / netstumbling / lanjacking utility for bsd operating systems, http://www.dachb0den.com/projects/dstumbler.html

[FreeNetworks] Free Networks, information on different community networks, http://www.freenetworks.org/

[HostAP] Host AP driver for Intersil Prism2/2.5/3, http://hostap.epitest.fi/

[Hu] Y. Hu, A.Perrig, D.B.Johnson, *Ariadne, a Secure On-Demand Routing Protocol for Ad-Hoc Networks.*, MobiCom 2002,(2002).

[Ho] Hata-Okumura Model, http://wcrg.engr.ucf.edu/Web_Paper/HATA.html

[IEEE] IEEE working group for wireless LANs, http://grouper.ieee.org/groups/802/11/

[Kismet] Kismet sniffer, http://www.kismetwireless.net/

[Maltz] David A. Maltz, Josh Broch, and David B. Johnson, *Experiences Designing and Building a Multi-Hop Wireless Ad Hoc Network Testbed.*, MU School of Computer Science Technical Report CMU-CS-99-116. March 1999.

[MobileIP] IP Routing for Wireless/Mobile Hosts (mobileip), http://www.ietf.org/html.charters/mobileip-charter.html

[NoCat] The NoCat Community Wireless Network Project white paper, http://nocat.net/nocatrfc.txt

[nyc] New York City Wireless, http://nycwireless.net/

[Omni] Trevor Marshall, *802.11b WLAN Waveguide Antennas*, http://trevormarshall.com/waveguides.htm

[Perl5] Perl5 Programmers reference, http://www.metronet.com/perlinfo/doc, (1996).

[PowerF] Power over Ethernet FAQ, http://www.nycwireless.net/poe/

[PowerS] Power over Ethernet standard working group http://grouper.ieee.org/groups/802/3/af/

[Pringle] Rob Flickenger, *Antenna on the Cheap (er, Chip)*, O'Reilly weblogs, Jul 5 2001. http://www.oreillynet.com/cs/weblog/view/wlg/448

[Quad] The Dutch double Quad antenna, http://sharon.esrac.ele.tue.nl/pub/antenne/13-double-quad.jpg

[Routing] R.Malhotra, *IP Routing*, O'Reilly & Associates, Inc. (2002).

[Royer] E. Royer and C.K. Toh, *A review of current routing protocols for ad-hoc mobile wireless networks.*, IEEE Personal Communications Magazine, pp 46-55 (1999).

[RRDB] Tobi Oetiker, *RRDB and RRDtool*, http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/

[Seattlewireless] Seattlewireless, a wireless community in Seattle, http://www.seattlewireless.net

[svn] Subversion, http://www.tigris.org/

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

[WirelessComm] Rob Flickenger, *Building Wireless Community Networks*, O'Reilly & Associates, Inc. (2001).

[WCompare] Netgear *Wireless Comparison Whitepaper*, http://www.netgear.com/pdf_docs/WirelessInforev4.pdf

[WirelessNet] M. Gast, *802.11 Wireless Networks*, O'Reilly & Associates, Inc. (2002).

[Yagi] SeattleWireless, *Building Yagi antennas*, http://seattlewireless.net/index.cgi/BuildingYagiAntennas

[Zebra] The ZEBRA routing package, http://www.zebra.org/