Bournemouth University
School of Design, Engineering and Computing

Individual Project undertaken in fulfilment of the requirements of the
BSc (Honours) Degree in Computing

Managing Wireless Hotspots

By
Kevin Martin

Project Supervisor
Steve Welsh

Date of Submission
2004

Abstract

This report presents a critical review of the various authentication systems available for wireless networks. It also demonstrates the process for implementing a wireless Hotspot and also a useful software package to help in administering wireless Hotspots. In particular focus throughout this report are networks that provide Internet access to the general public, be it free or at a charge.

There are various issues that can occur when providing wireless Internet access, which are also explored, and the report highlights methods that can be used to solve these issues. Research is also presented on the current technology available to integrate wireless networks, what this technology provides and how wireless networks work.

The developed software package has been aimed at those administering and installing Hotspots at different locations. The package allows any administrator to gain access and manage the Hotspot remotely over the Internet. It has also been developed to integrate with the widely used, very popular and open source gateway and authentication package NoCatAuth.

The developed software, which is called CoNAT, has also been made available for free download at: http://www.kevmartin.com.

Acknowledgements

Thank you...

Steve Welsh, of Bournemouth University for supervising this project and also for his general advice on the Linux operating system.

The entire Intel UK ADC department for their support during my work placement at Intel UK and for helping to provide the inspiration for this project whilst attending the spring 2003 Intel Developer Forum in Berlin.

Ashton Waverley Computers, Bournemouth for the supply of various networking components during this project.

The authors of the NoCatAuth software and those on the NoCatAuth mailing list for their useful comments.

My Parents and friends for the great support offered by them during the very stressful parts of this project.

Contents

1 Introduction

1.1 Background

Over recent year's mobile technologies, such as Mobile Phones, PDA's and Laptop computers have become common place. These technologies have allowed their users to become more mobile and work anytime, anywhere – At home, a conference, on the train or even an aeroplane.

With the rapid increase in these technologies, so has a person's need and method of communicating changed. Today someone is able to be away from their office, but still: be contactable by mobile phone, have access to voice and electronic mail, and communicate with their friends and colleagues via messaging services.

As these methods have re-defined peoples lives, the need to provide all of this without being 'plugged into the wall', but being and staying wireless (needing only to 'plug in' to charge batteries) has come about. Many options are available, but the most widely sort after and needed is wireless Internet access which has been cumbersome, hard to come by and expensive.

There are currently many ways mobile users can connect to the Internet while away, including:

- Wi-Fi (Wireless Fidelity) – by far the most successful technology, and is often the term referred to the most when describing wireless networks.
- GPRS (General Packet Radio Service) – Provides a WWAN (Wireless Wide Area Network) via the GSM (Global Mobile System) communications standard. A theoretical maximum receive speed of 171.2 Kbps is possible. It is also a successful standard, due to its use in Mobile phones.
- UMTS (Universal Mobile Telecommunications Standard) – The so called '3rd Generation' or 3G system. Allowing for much higher speed communications using advanced mobile phone systems. Data transfer speeds in excess of 384Kbps are possible.
- Bluetooth – A PLAN (Personal Local Area Network) commonly used to connect laptop computers and mobile phones. With a range of around 10 metres, its use is commonly for communicating between laptops or PDA's and GPRS or UMTS mobile phones.
- Satellite – Using microwave signals to transfer large amounts of data into space and back, over vast distance commonly used for ship-to-shore based communications. Aircraft maker Boeing has been working on providing wireless Internet access on its planes, via a Satellite link.
- Wired Ethernet Link – Running at speeds ranging from 10 Mbps to 1000 Mbps.

- Dialup – Available almost anywhere, either over fixed analogue / digital telephone lines or via GSM, ranging in speeds from 9.6 Kbps (GSM) to 128 Kbps (ISDN (Integrated Services Digital Network)).

1.2 The Problem

Both Wi-Fi and Ethernet provide high-speed access to computer networks. On their own and in a fixed environment (e.g. the workplace, the home), they work without issue. However, when a user goes to a different location, they still want to work as if they were connected to their normal network. VPN (Virtual Private Network) software provides the user this access to their network (by creating an encrypted virtual tunnel across the Internet), but in order for this to function, their connection must be setup and connected to the Internet.

Ethernet connections are often the easiest to setup, typically by simply plugging a network cable into a socket and waiting for the Ethernet card to pick up a DHCP (Dynamic Host Control Protocol) address. Wi-Fi can take much more work, and involves the user having to scan for available wireless networks. Once a suitable wireless network is found and selected, the wireless connection then attempts to obtain a DHCP address.

These processes are often time consuming and can cause issues, namely with:-

1.2.1 Security

A user could connect to a network which they should not have access to or may contain information intended only for the owners of the network. Other issues might occur, such as an unscrupulous user gaining access to the network and sending ‘spam’ emails to people, pertaining to be from the company owning the network.

Communications will often be un-encrypted, allowing for those with the correct equipment to tap in and view information being sent. VPN solutions remove this aspect through the use of encryption.

1.2.2 Billing

There is no easy way to record a particular users connections and the time they have been connected to a network. It is possible to record information such as the MAC (Medium Access Control) address of the connecting network card, its IP (Internet Protocol) address, etc, but this information does not help in billing users for connecting to the system, it can only be used for statistical purposes.

1.2.3 Ease of use

Current solutions to provide logging and security on a network are often mundane and cumbersome, requiring complex networking skills to use effectively.

1.3 Project Aims and Objectives

- Investigate the benefits of wireless network access.
- Research into current network access gateways (like NoCatAuth Gateway, Aptilo's Captive Portal, etc).
- Research possible billing methods (like credit card, scratch cards, free access, account based, etc).
- Implement a wireless hotspot (using an Access Point and a simple Linux / Windows Portal/Gateway).
- Create new portal software or improve a current system such as NoCatAuth.

1.4 Project Constraints

1.4.1 Resources

- Hardware issues occurring throughout the development may restrict the projects progress.
- Hardware limitations may affect the testing and speed of development.
- Software issues may include a lack of functionality with the software that any development will interface with.
- Problems with Word Processing and Presentation software.
- Development tools, such as text editors and compilers.

1.4.2 Time

- The project must be completed and handed in by its deadline of 30th April 2004.
- The project will be completed whilst working on other university projects, which may lead to unforeseen delays in the progress of the project.
- The project and any software solution created will need to be demonstrated after the hand in date.

1.4.3 Knowledge

- Networking: My knowledge of certain areas of computer networking may be limited, and time, during research and implementation will need to be included to further this.
- Programming Language: Depending on the programming language being used, I may need to refresh my knowledge and consult reference material for advice.
- OS (Operating System): It is likely that any implementation will take place using a server grade OS such as Linux. Whilst I have a vast knowledge of Windows based systems, my knowledge of Linux based systems is limited, and this will need to be furthered throughout the project.

2 Research

2.1 Ethernet – IEEE 802.3

2.1.1 Background

Bob Metcalfe researched into computer networking whilst at Xerox PARC (Palo Alto Research Centre) in the 1970's. Metcalfe had read the studies of Norman Abramson who earlier devised a way of connecting University of Hawaii users to the main computer in Honolulu via radio.

Later, together with David Boggs, Metcalfe went about developing a way of linking Xerox's PCs (Personal Computers) together, creating the first LAN in 1976. Tanenbaum writes that Boggs and Metcalfe decided to call it "Ethernet after "luminiferous ether", which electromagnetic radiation was once thought to propagate." (*Computer Networks, 2003, PG. 66*)

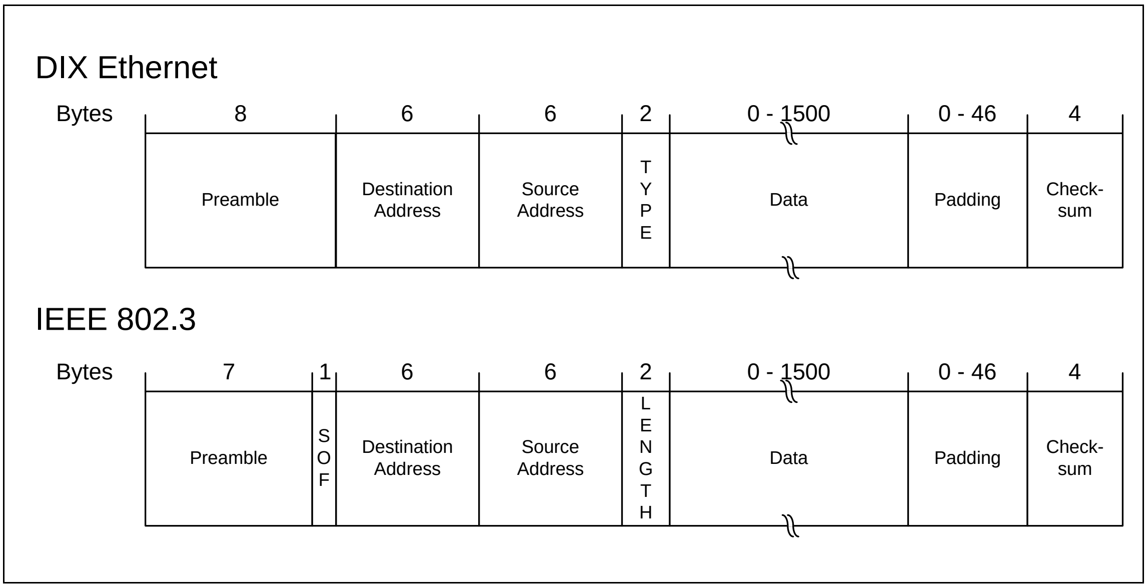
Running at 2.94 Mbps, using a thick coaxial cable (the ether) up to 2.5 Km long and supporting up to 256 machines via transceivers screwed onto the cable – a 'Multidrop Cable'. Avoiding transmission collision was to become the major benefit of Ethernet.

Boggs and Metcalfe proposed CSMA/CD (Carrier Sense Multiple Access with Collision Detection); if a machine wants to transmit, it would first check that the shared transmission medium is not in use (the "Carrier Sense" part) backing off until it is not. If a transmission is sent but it meets another transmission along the medium (i.e. transmissions are sent at the same time or propagation delays in detecting the other transmission), a jamming signal is sent to all machines (the "Collision Detection" part). Senders would then back off for a random period of time before retrying. If further collisions occurred the random period of time would increase, until transmission succeeded. Correctly switched Ethernets do not suffer from collisions as the shared transmission medium with a dedicated segment for each machine.

"The Xerox Ethernet was so successful that DEC, Intel and Xerox drew up a standard in 1978 for a 10-Mbps Ethernet, called the **DIX standard**. With two minor changes this became the IEEE 802.3 standard in 1983."

(Computer Networks, 2003, PG. 67)

2.1.2 Technology



DIX Ethernet

Bytes	8	6	6	2	0 - 1500	0 - 46	4
	Preamble	Destination Address	Source Address	TYPE	Data	Padding	Check-sum

IEEE 802.3

Bytes	7	1	6	6	2	0 - 1500	0 - 46	4
	Preamble	SOF	Destination Address	Source Address	LENGTH	Data	Padding	Check-sum

Figure 2.1 DIX and IEEE 802.3 Frame formats (Computer Networks, 2003, PG. 276)

Ethernet uses frames to transmit data as shown in Figure 2.1, containing:

- Preamble of 7 or 8 bytes, for each byte the bit pattern ‘10101010’, which as described by Tanenbaum “Manchester encoding of this pattern produces a 10 MHz square wave for 6.4µsec to allow the receiver’s clock to synchronize with the sender’s.” (*Computer Networks, 2003, PG. 275*)
- Source and Destination 48-bit MAC (Medium Access Control) addresses.
- Length, the length of the data section within the frame.
- Data
- Padding, if the data is less than 46 bytes, then this is padded out up to 46 bytes.
- Checksum, a CRC (Cyclic Redundancy Check) on the data within the frame.

10 Mbps, Ethernet (IEEE 802.3)
100 Mbps, Fast Ethernet (IEEE 802.3u)
1000 Mbps, Gigabit Ethernet (IEEE 802.3ab)
10000 Mbps, 10 Gbps Ethernet (10 GbE) (IEEE 802.3ae)

Figure 2.2 Current Ethernet Speeds

As a networks speed goes up so its minimum frame length should increase or the cable length decrease to compensate. Gigabit Ethernet introduced CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance), which works slightly differently from CSMA/CD by the sender telling other machines not to transmit whilst it is transmitting.

2.2 Wi-Fi – IEEE 802.11

2.2.1 Background

Research into Wireless LAN's started in 1992, by looking at US military technology, which used the unregulated 2.4 GHz medical radio frequency. Initially its uptake was by those in the Healthcare and Educational sectors, whereby users (e.g. doctors and teachers) were often moving around (e.g. between hospital wards / school classrooms).

To use a Wi-Fi based device is like using any other device on an Ethernet network, except that they are not connected through a physical cable and therefore can be easily moved. Tanenbaum notes that "Almost all wireless networks hook up to the wired network at some point to provide access to files, databases and the Internet" (*Computer Networks, 2003, PG. 22*)

2.2.2 Technology

Wi-fi devices typically use CSMA/CA within DCF (Digital Coordination Function) which uses no central control (like Ethernet) which forces devices to compete for air time. Another option is PCF (Point Coordination Function) which uses a central control (the "base station") removes any chance of collision. All Wi-fi devices must support DCF, with PCF as a valuable, but optional extra.

Wi-fi devices also have 48-bit MAC addresses and appear as fixed Ethernet devices on a network. Both Wi-Fi and Ethernet devices use the same MAC address pool, therefore removing any possible MAC address clashing.

Devices managing the infrastructure of a Wi-Fi network (typically an AP (Access Point)) know that a Wi-Fi device's MAC address can be physically moved around the network. Devices on the Wi-Fi network make sure that the required Ethernet frames are directed to the current location of the device.

Wi-Fi can be classed as another link layer, capable of using IEEE 802.2/LLC (Logical Link Control) encapsulation. The original IEEE 802.11 specification includes the 802.11 MAC layer and two other layers: the FHSS (Frequency Hopping Spread-spectrum) physical layer and the DSSS (Direct-sequence spread-spectrum) link layer. IEEE 802.11b introduced HR-DSSS (High Rate DSSS) and a layer for IR (Infrared) was also included, but never widely developed.

Some Wi-Fi devices work in the 5 GHz band, and can be more resilient to other radio signals, such as cordless phones and microwaves, which often work in the 2.4 GHz band. These 5 GHz devices operate using a different technique, called OFDM (Orthogonal Frequency Divisional Multiplexing).

Wireless devices typically have a range of around 100m outdoors, and around 30m indoors depending on obstacles, though further ranges are possible with modification.

IEEE Standard	Maximum Supported Speeds	Frequency Band	Technologies
802.11	1 Mbps, 2Mbps	2.4 GHz	FHSS, DSSS
Original wireless standard in 1997.			
802.11a	Up to 54 Mbps	5 GHz	OFDM
Second wireless standard, standardised in 1999 but products based on 802.11a not released until late 2000.			
802.11b	5.5 Mbps, 11 Mbps	2.4 GHz	HR-DSSS, DSSS
Third standard, standardised shortly after 802.11a, but products carrying the standard released second. Currently the most common standard in operation.			
802.11g	Up to 54 Mbps	2.4 GHz	HR-DSSS, DSSS
Currently the newest standard, standardised in June 2003, fast becoming the most commonly supported standard and also supports 802.11b devices.			

Figure 2.3 Comparison of the various IEEE 802.11 Standards

2.2.3 Infrastructure

The two main components of a wireless network are:

An AP (Access Point): The point of entry to a wired infrastructure for Wi-Fi, each access point is capable of supporting many clients at any one time. Broadcasting a customisable networks name – the ESSID (Extended Service Set Identifier), allowing wireless clients to identify and associate with a network. All hardware Access Points provide master BSS (Basic Service Set) used for bridging wired and wireless networks.

AP's can be set up to support more than one AP on a network, this is known as ESS (Extended Service Set). AP's using ESS allow a wireless client to 'roam' (move) between AP's. Some AP's can also be set to work as BSS clients, providing a wireless bridge between two wired networks.

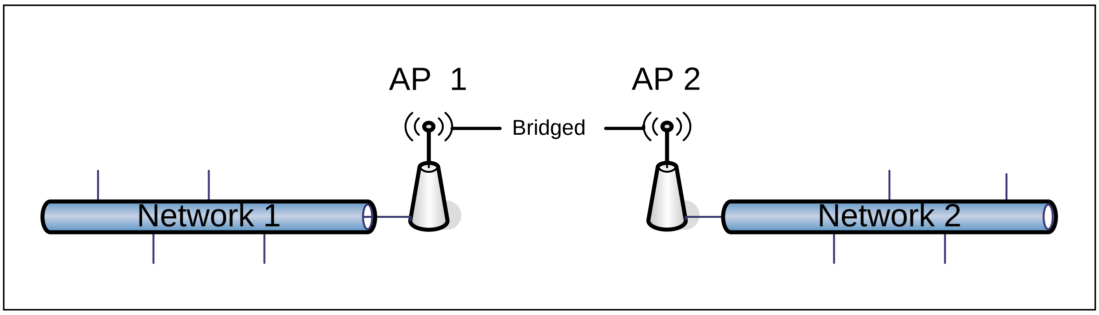


Figure 2.4 – Bridging two wired networks, wirelessly.

The 2.4 Ghz spectrum provides 11 operating channels, each using 22 MHz of signal bandwidth, "so adjacent radios will need to be separated by at least five channels to see zero overlap." (*Building Wireless Community*

Networks, 2003, Pg. 16). If channels overlap, this can cause many re-transmissions of data.

Channels 1, 6 and 11 have no overlap.

Neither do 2 and 7, 3 and 8, 4 and 9, or 5 and 10.

Figure 2.5 – Example channel separation (Building Wireless Community Networks, 2003, Pg. 16)

A Wireless Adaptor: The connection from a client to the wireless network, often a PCMCIA (Personal Computer Memory Card International Association) card, USB (Universal Serial Bus) adaptor or Mini-PCi (Peripheral Component Interconnect) card. These are known as BSS Clients, and can be set to talk to other BSS clients as well as BSS masters.

Some BSS clients are capable of operating as BSS masters with the correct configuration and software. These are ideal for small networks with only a couple of wireless clients, and one acting as the Internet Gateway. These BSS masters are often known as HostAP's.

Wireless devices can be placed in one of two radio modes:

Ad-hoc (Peer-to-Peer): Using IBSS (Independent Basic Service Set) based wireless adaptors, clients can talk to each other without the need for an AP, subject to them having the same ESSID and WEP (Wired Equivalent Policy) settings. This is useful when setting up temporary or moving networks. It is also possible for one of the clients to act as a gateway providing Internet or other network access. Flickenger makes it clear that "In this mode, no hardware AP is required" (*Building Wireless Community Networks, 2003, Pg.* 21)

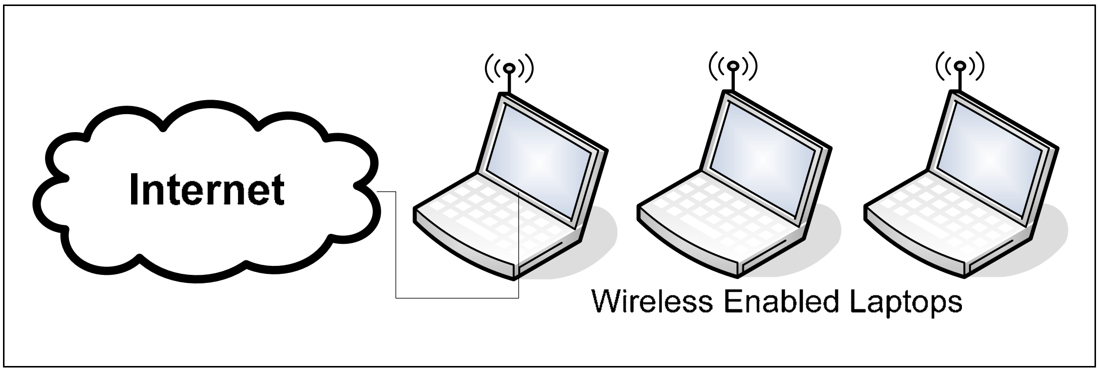


Figure 2.6 – Ad Hoc (Peer-to-Peer) wireless network, with a host providing a gateway to the Internet.

Infrastructure: Using BSS masters, clients are still able to talk to each other but any data is relayed via the AP. This is the better solution for larger networks and is considered to be more stable. When configured for Infrastructure, Wireless adaptors are unable to talk to each other without going through the AP.

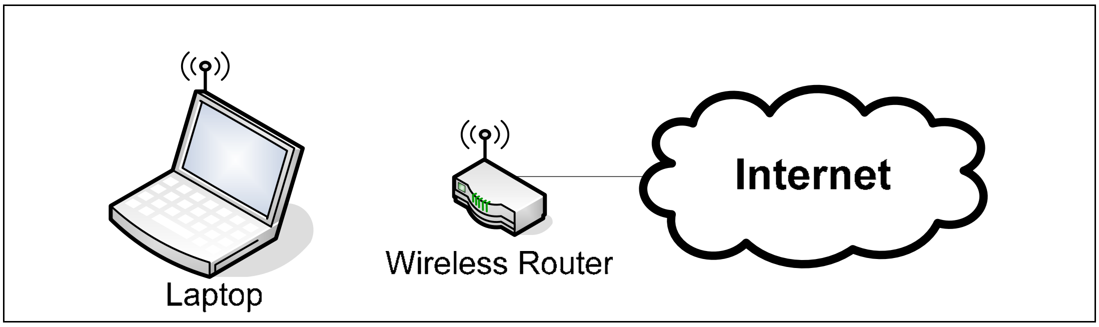


Figure 2.7 – Infrastructure Mode with wireless router comprising AP (wireless) and Switch (Ethernet)

"In this operating mode, one station (the BSS master, usually a piece of hardware, called an access point, or AP) provides wireless-to-Ethernet bridging" (*Building Wireless Community Networks, 2003, Pg. 21*)

2.3 The Hotspot Concept

In recent years, the concept of Wi-Fi based 'Hotspots' has been introduced. In his paper, "Hotspot in a Box", Ziakas, et al defines a Hotspot as being "a Wireless Local Area Network using the IEEE 802.11 standard to offer a geographical coverage area of up to a few hundred meters or less" (*"Hotspot in a Box", 2003, Pg. 4*). Hotspots are normally positioned in places where there is a constant flow of people. Typical take up is by those in the service industries such as: Conference Centres, Railway Stations, Airports and Hotels. Most users of hotspots tend to be business travellers, who may have to wait several hours at an airport for a flight or might be staying a few nights at a hotel.

More often than not, users of hotspots are most likely to VPN into their own network, and hotspot providers have striven to provide their products as secure and reliable. Typical hotspot providers include the multinational telecommunication operators like: Vodafone, BT and T-Mobile; and localised providers, which have established networks of hotspots in different countries. Each of these networks often have many AP's and are connected to multi-megabyte Internet connections.

Larger operators often use a shared authentication system, allowing for subscription options, so that, for example, a business user would be able to use their account in multiple places.

Hotspots can also be operated by smaller providers, such as shops, bars and cafes. These smaller operators may only have a couple of access points and a broadband Internet connection. They may offer the hotspot as a unique selling point of their business, as these can be created at little cost (Initial infrastructure outlay + monthly broadband subscription).

An example of this is a café owner, who might already have broadband internet access, and wants to extend access to the cafes customers. The easiest way to do this is to purchase a wireless enabled router (integrated router and access point), supporting their broadband connection. The café owner configures the router to offer internet access by broadcasting the SSID and providing its details to the customers when they visit.

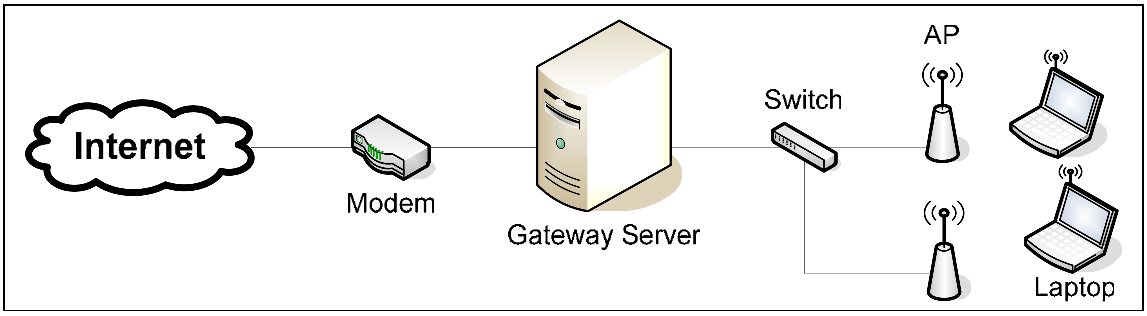


Figure 2.8 – Typical Coffee Shop 'Hotspot' setup

2.4 Controlling Wireless Access

2.4.1 WEP

WEP stands for Wired Equivalent Privacy, and is described in Vines book as "An optional IEEE 802.11 function that offers frame transmission privacy similar to that of a wired network" (*Wireless Security Essentials, 2003, Pg. 270*). WEP generates encryption keys that the source and destination can use to encrypt or decrypt frames being sent. WEP encryption can be set at varying levels, including: 40-bit, 56-bit, 126-bit and 256-bit. It is possible for WEP keys to be automatically assigned to any client on connection, therefore producing a secured connection automatically.

If WEP is used on a wireless network, and a client connects without a WEP key or with an invalid key, no network access is possible until this is rectified. It is possible, with the aid of specialised software, to calculate the WEP key in use on a wireless network and as such gain unauthorised access. WEP is often incorrectly described by some industry professionals as Wireless Encryption Protocol.

2.4.2 IEEE 802.1x

IEEE 802.1x provides Port Based Network Access Control and was drafted in 2001 to improve security in IEEE 802.11b networks. Vines also notes that "It [IEEE 802.1x] provides port-level authentication for any wired or wireless Ethernet client system" (*Wireless Security Essentials, 2003, Pg. 67*)

IEEE 802.1x is ideal for large numbers of connections. An example is a university operating a large campus wide network, which would not be happy that just anyone could connect and gain access to key information. IEEE 802.1x is able to enforce policies on access to systems, meaning that those who should be allowed access (students and faculty in the example), can gain access and those from outside, visitors, are not permitted access.

All this is controlled with credentials such as MAC addresses, static IP addresses and username/password. IEEE 802.1x is also susceptible to session hijacking methods like DOS (Denial of Service) and man in the middle attacks when used in wireless networking.

2.4.3 Combining WEP and IEEE 802.1x

WEP and IEEE 802.1x are regarded by many as the fixes for authentication and security within wireless networks. When combining WEP and IEEE 802.1x, it is possible to dynamically allocate WEP keys per session, and re-allocate these keys every so often (but more often than it takes to guess a key), thus defeating any purpose in guessing the keys values.

2.4.4 Secure Authentication Systems

Ideally on a wireless network, before full network access is allowed, the client must first authenticate. On first connecting, the client should only be able to communicate with the authentication system in use. To gain full access it must utilise services provided by the authentication system (the authenticator). This would involve transmitting credentials (Username, Password, MAC address, etc) to the authenticator, where these are checked with a RADIUS (Remote Authentication Dial-In User Service) server or existing user database. If the required credentials match, then the authenticator notifies the client of its success and that it has full network access otherwise the client is asked for further credentials.

Problems can occur, for instance when a new user joins a network and their details have not been added to the service already. Authentication servers provide an ideal tool when dealing with a static number of users. Issues can also arise when a user needs to authenticate but is not aware of this and as such "they should be notified when roaming is to occur and be forced to input authentication information or manually acknowledge this." (*Matsunaga, et al. 2003. Pg. 116*).

Mansunaga, et al. goes on further to detail that "Most public wireless LAN systems use web-based authentication schemes, and users can get IP-level network access before showing their identity and credentials." (*Matsunaga, et al. 2003. Pg. 118*).

3 Current Authentication Solutions

These are some of the most popular of software and hardware solutions available in controlling wireless authentication and use. This type of software is often classed as Captive Portals.

3.1 NoCatAuth Gateway and Authentication Server

NoCatAuth is widely used by community wireless networks as it is open source. Mike Kershaw writes in his NoCatAuth based hotspot guide that "NoCat builds a captive portal by assigning incoming users an IP address using DHCP and restricting network access until the user has validated" (*Linux-Powered Wireless Hot Spots, 2003*). NoCatAuth can support several levels of user such as: guest, paying customer and administrator. Kershaw also details in his guide, how NoCatAuth redirects traffic "By rewriting the destination of all port 80 traffic in the firewall, any Web page the user attempts to visit before validating can be rerouted to the portal login page".

Only a few AP's have the ability to control access to a network (be it MAC address, WEP, etc), and each make or model of AP typically has a different way of configuring this. NoCatAuth solves this using the IPTables firewall package included with Linux to control network access dynamically.

NoCatAuth has two main components; the Gateway and the Authentication server. When operating as the Gateway, NoCatAuth dynamically rewrites the firewall rules as users connect and as users disconnect. NoCatAuth works best in this mode when no other firewall rules are in place on the system.

As Authentication server, controls the storage and retrieval of user accounts and passwords, comparing credentials when they are sent to it and returning the success of comparing. Details can be stored in a file, SQL (Structured Query Language) database table, RADIUS server or via a Domain Logon service.

Typically when using NoCatAuth one Gateway is used per an AP, with many Gateways on a network and a single Authentication server for the whole network. Indeed, the Authentication server does not need to even be on the same network, and could be an internet source.

It is also possible to install both NoCatAuth's main components on the same machine, although this is not recommended and Kershaw also details that "it's more secure and easier for multiple gateways to use a single authentication server if you use separate machines for the authentication server and gateway." (*Linux-Powered Wireless Hot Spots, 2003*)

NoCatAuth provides good IP port security, allowing the network administrator to block IP ports and domains.

3.2 Aptilo Networks PWLAN Solutions

The PWLAN (Public Wireless LAN) solution is described by the Aptilo website as "a complete system for managing and operating professional public wireless LANs" (*Aptilio PWLAN Solutions Overview, 2004*). The system comprises:

- Aptilo Service Management Platform – the core component, dealing with authentication of users, service control, access management, configuration and monitoring. Works in partnership with current network infrastructure. Typically hosted at the operators network operations centre, although Aptilo are able to offer a managed service, whereby they host the equipment.
- Aptilo's Access Gateway – Communicating with the Service Management Platform over a VPN Internet connection, supporting multiple subnets, access point monitoring, and dynamic sessions for users. "Depending on the business model and integration level, the Access Gateway can either be centrally, regionally or locally placed in the network, catering for several separate sites." (*Aptilo PWLAN Solutions Access Gateway, 2004*)

A key feature of Aptio's solution is that it is able to work with other vendors gateways, meaning that costs can be lowered quite considerably if some infrastructure exists already. Other key features include the ability to block access to certain sites or only allow access to certain sites. It is also possible to maintain different levels of user accounts, so certain users can access anything without being charged whilst placing limits on others.

Configuration is web based, and can take place remotely if desired. Users are also able to setup an account on visiting an Aptilo based network, if enabled by its operator.

3.3 Birdstep IP Zone

IP Zone is a commercial product, providing comprehensive network management. Consisting of four components, which are:

1. The Birdstep IP Zone Access controller – Providing the front end authentication services and is typically installed locally to the network, acting also as the gateway machine, picking up users requests and redirecting these to the authentication pages.

2. The Birdstep IP Zone back end – Providing the authentication services, which can also be localised or distributed across the Internet. This back end can also interface with other operator's authentication systems allowing roaming between operator's hotspots.

3. The Birdstep IP Zone Nomadic Portal – generates all portal pages on the system, integrating with the Access controller and back end to provide

both device specific and localised information, such as available services like printers.

4. The Birdstep IP Zone Billing Gateway – integrates with external billing and accounting solutions like SMS (Short Messaging Service) and credit card payments. One Billing Gateway can serve many back end servers.

The complete set of components can be installed on one server, or distributed over many, providing a scalable and reliable solution. The architecture provides a solution that can offer centralized, fully distributed, or a combination of both centralized and distributed deployment, providing further flexibility and scaling to the network owner.

"Birdstep IP Zone Server supports terminal access over wireless infrastructures such as WLAN (IEEE 802.11b/a when available), Bluetooth HiperLAN2 and IrDA." (*Birdstep IP Zone Server, Pg. 1*)

IP Zone can be quickly and easily rolled out, running on any x86 based platform. It is also able to utilise any existing billing solution in place helping in a speedier rollout. Like other solutions, it can support many different types of authentication such as username/password, credit card, SMS and voucher payments.

3.4 IPUnplugged Internet Access Control

IPUnplugged IAC (Internet Access Control) provides a flexible architecture for wireless internet access as well as supporting multiple methods of authentication. At the heart of IAC is the roaming gateway, which can either be placed local to the network or distributed so that it can be accessed over the Internet.

"The Roaming Gateway is placed in the traffic path and performs authorization and accounting of the users. A user that has not yet logged in is redirected to the Roaming Server or any other third party portal that is performing the authentication." (*ipUnplugged Internet Access Control, 2001*)

IAC utilises many different authentication methods, including the most popular username / password, and others such as Credit Card billing and Mobile IP authentication. IAC focuses on security, encrypting all user data, protecting unauthorised access. This is done without putting any requirements on the client - all the client requires is a web browser.

4 Billing mechanisms

The following is a list of possible billing mechanisms and is derived from the research by Ziakas, et al. (All examples assume that the user wants to browse the internet and has already established the connection to the Hotspot using the correct ESSID):

4.1 Free Access

The user is given unlimited access, simple and cost effective requiring only an Access Point, Router and Internet Connection.

4.2 Free Access with User Agreement

Users must accept an agreement / disclaimer before Internet access is allowed. This system is preferable for those wanting to offer internet access without being liable for any misuse or failures of the system.

4.3 Free Timed Access

Users can use the internet for a fixed period of time (e.g. 20 minutes) before being disconnected. The user may have to accept an agreement / disclaimer before Internet access is allowed. Likely to be used in locations where users need access for a short limited time, e.g. Airports.

4.4 One-off Payment

The user is redirected to a Payment page, where they provide payment information and wait for processing. On successful processing of the payment, the user is then able to use the internet for the length of time they have paid for. Once the time has expired, they are redirected back to the payment page.

4.5 Subscription

A user subscribes to a service, for which they can access Hotspot’s operated by the service. On connecting to a Hotspot within their subscription package, they are redirected to a login page, where they enter their subscription credentials. On successfully authenticating the user, they are able to access the Internet.

Operators may use subscriptions such as:

- Monthly subscription with an included usage allowance or credit, subsequent use is charged.
- Monthly subscription with unlimited usage allowance.
- No monthly subscription, but is charged as they use.
- No chargeable subscription, but must register (perhaps receiving promotions through email).

4.6 Voucher Payment

A Scratch card or “virtual voucher” is purchased and on connecting to the hotspot, the user is redirected to a web page whereby they provide the vouchers details. If the voucher is valid, then access is provided for the time associated with it and the voucher is then made unusable.

4.7 Smart Client Authentication

Subscription based with “the payment and authentication is processed by a broker that has agreements with a number of different WISPs therefore allowing users access to a multitude of different locations.” (“Hotspot in a Box”, 2003, Pg. 8). This broker then works as a “Virtual Hotspot Operator”.

4.8 SMS (Text Messaging) Payment

Payments are taken via text messaging for pre determined access times, and can be taken in one of two ways:-

1. Reverse Text Charging – The user receives a chargeable message, with payment being debited from their account. (Authentication that the user owns the phone would need to take place first)

2. Text Charging – The user sends a message to a premium number, within a few seconds the user then receives a free message with a useable access code.

An agreement with a SMS provider would be needed who in turn would take a percentage of around 10% of any charges.

4.9 SIM Authentication

“the Hotspot authenticates the user using the SIM user information from his mobile phone and the bill is charged to the user’s mobile phone. This billing method requires linking into the GSM/GPRS networks for the SIM authentication.” (*“Hotspot in a Box”, 2003, Pg. 8*).

Complex implementation and requires agreements with the various telecommunication networks. This does provide a secure and efficient way to authenticate users.

5 Implementation

5.1 A Simple Hotspot

The following is a hotspot that has been set up to provide Internet access and is derived from the step-by-step by Schuyler (*Recipe for a Linux 802.11b Home Network, 2001*), with the major differences being that this hotspot uses IPTables, the replacement to IPchains and also the use of an AP rather than a wireless card:

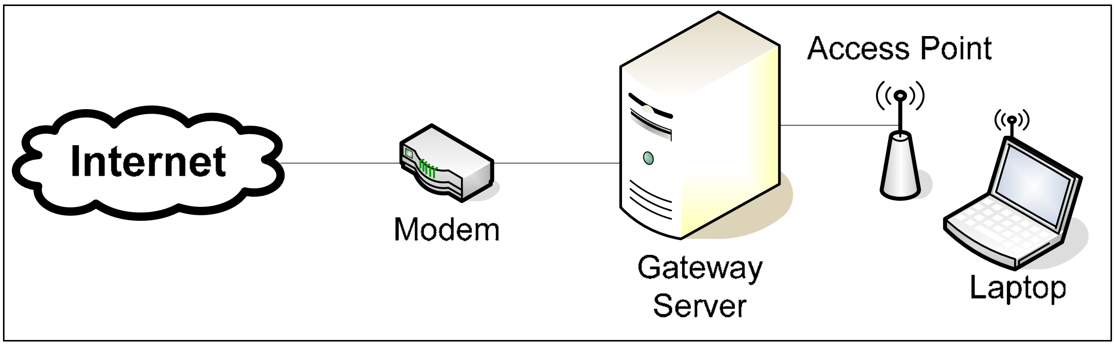


Figure 5.1 – Simple Hotspot Setup

This particular hotspot configuration consists of a Cable Modem operating at 600 Kbps, a CAT5 Crossover cable from this to a 10 Mbps Ethernet card ("External NIC") in a Intel Pentium III based desktop PC running Red Hat Linux 9.2 ("The Server"). A 100 Mbps Ethernet card ("Internal NIC") is also present and is connected to the Ethernet socket on the Intel 802.11b Access Point ("The AP") via CAT5 Crossover cable.

The Server is running several services, but the following are required to correctly run the hotspot:-

- DHCPD (Dynamic Host Control Protocol Daemon) – IP addressing on the Internal NIC.
- IPTables – providing firewall, masquerade and forwarding between the Internal and External NIC's.
- BIND (Berkeley Internet Name Daemon) – top level domain name caching.

Configuration Scripts for these modules are included in the appendices. DHCPD and BIND's configuration scripts are automatically used on service execution. The configuration script for implementing IPTables rules is written in BASH (Bourne Again SHell) and is executed directly:

```
Sh ./firewall.sh
```

The AP is configured with the ESSID of "hotspot" and WEP has been disabled. It is also configured to be assigned an IP address by the DHCPD, and with the geographic location of the UK. The particular AP in use is configurable through any web browser, telnet or RS232.

5.2 Writing an Authentication Solution

An Authentication Server is very complex, often comprising of more than one application making use of several other services. Some services are similar to those documented in the Hotspot above and can also include services for accounting and controlling access.

Whilst the mechanisms to only allow selected clients access via the use of firewall rules and web servers is relatively simple, problems arise when wanting to control the time or bandwidth clients are allowed on the network as well as what occurs when a user un-expectantly disconnects or does not respond when out of wireless range.

Authentication solutions work using the principles that once a client has been authenticated, they are allowed network access, at the same time the clients details are added to a list which is periodically checked (for example every 60 seconds) to see if they are still on the network. If a client is not found then rules are applied based on the clients details, blocking access and forcing them to re-authenticate in the future.

Any authentication solution that could be written would not offer all the options that the solutions reviewed could offer, and to do so would take considerable effort and time. It is far better to consider developing additions to current solutions, such as to improve the ease with which a gateway can be installed and put into use.

5.3 Adding NoCatAuth to the Hotspot

NoCatAuth was installed on the server using the comprehensive Instructions located on Wi-FiPlanets Website. Slight variances to the configuration detailed on Wi-FiPlanets website were made and are available commented in the configuration scripts section of the appendices.

Installing NoCatAuth is very simple, but it can take some time to configure. It is suggested that to aid in the configuration, a simple hotspot is instigated first to confirm that the wireless network and internet access can be achieved. If this is done NoCatAuth's configuration should be relatively quick and any problems would only be in NoCatAuth's configuration.

Both the Gateway and the Authentication Server have been installed on the same machines (Although this is not recommended by NoCatAuth's authors, it was not possible to locate two machines). This also required two set's of the Perl library files to be installed. Since this hotspot is only being used for development purposes and is not to be used by the public then these issues do not cause a large problem.

5.4 Management of NoCatAuth – CoNAT

The NoCatAuth software is very popular, particularly in the open-source world where it has also been developed. It provides many useful features, however it can be very difficult to administer. Unlike the other products covered earlier, NoCatAuth does not offer any web based management, only a web based status page, which is meant to only be viewed on the server or from the network.

To configure NoCatAuth an administrator must have permission to modify the "nocat.conf" file within the directory that NoCatAuth is installed in. An allowed administrator can access this file in one of three ways:

1. Sharing the file through Samba or FTP (File Transfer Protocol) therefore making it accessible on any other computer.

2. Accessing the file by being logged into and sat at the server

3. Remotely connecting to the server through telnet or another similar method.

Each method requires either extra load on the server and the network, as well as inherent security risks through allowing file shares or running a telnet service. In the case of logging into the server, this would involve a visit to the server to configure it.

I have therefore decided to create a web front end for the management of NoCAT, allowing a network administrator to control and configure a NoCatAuth installation. CoNAT is written in the PHP (Hyper text Processor) scripting language, and interfaces with the Host Operating System, the NoCatAuth software and a MySQL Database.

Currently NoCatAuth uses plain text configuration files, which can make configuration difficult particularly for those new to the system. Within the configuration file, certain areas need to be disabled and others enabled to make elements work correctly. This is particularly evident when configuring options for the user information source when NoCatAuth is operating as an Authentication Server. Many sources are available such as RADIUS and Samba, however only one can be used at any time.

CoNAT also allows network administrators to manage a NoCatAuth install without being located on the network NoCatAuth is running on. This would allow an administrator to manage several installations across a network, reducing the need to visit a server to configure it.

CoNAT does not offer configuration of other services that could be running on the same machine, such as DHCP and DNS (Domain Name Service) which are often used by NoCAT. Software such as Webmin allows for configuration of many services within the Linux environment. CoNAT is meant to complement any other control software that may be in use.

5.4.1 How CoNAT Works

CoNAT works by accessing the “nocat.conf” configuration file installed in the same directory as NoCAT. The information in these files is then parsed by CoNAT and presented on screen as required. The user is then able to select and change each value as they want. The parser automatically by-passes and leaves intact any comments in the configuration files.

The network administrator is able to specify locations for both the gateway and the authentication server. If the administrator does not know the locations of these, CoNAT will try and detect the locations. CoNAT reads in the “nocat.log” file for both the gateway and the authentication server, to gather information on whether NoCatAuth is currently running, the date and time NoCatAuth was last started on the machine and what port it is running on. This information is presented on the status page, CoNAT also determines the Web Server being used through the PHP Server object value “SERVER_SOFTWARE”.

CoNAT also allows the administrator to start, stop and restart NoCatAuth as necessary, so that once changes are made to the configuration, these changes can be put in use straight away. This uses the “nocat.rc” BASH script which can be configured to start, stop or restart the gateway, the authentication server or both. The script can be executed in the following way:

```
Sudo /etc/rc.d/init.d/nocat.rc restart
```

The BASH script can only be executed as root. As it is not good policy to make the Apache group root, then the SuDo (Superuser Do) command should be used, to allow this to happen the “sudoers” file has the following line added to the bottom of it.

```
Apache    ALL=NOPASSWD:  /etc/rc.d/init.d/nocat.rc
```

Another feature of CoNAT allows the administrator to view and update user account details, if the user details for the NoCatAuth installation are in a MySQL database. CoNAT also encapsulates the NoCatAuth status page and presents this as a way of viewing currently connected clients and for checking MAC addresses.

CoNAT requires an administrator definable username and password, which is stored in a configuration file (“settings.conf”) along with the locations of NoCAT. The password is stored as a MD5 (Message Digest 5) hash in this file for security puposes. It is recommended that the username and password do not match any others in use on the server, to improve the security of the system.

On first logging into the CoNAT software with the username “user” and the password “password”, the user is presented with the CoNAT settings screen so that they may configure CoNAT for correct running.

CoNAT uses the PHP session information to store whether the user is logged into the system or not. If they are logged in, then all options are available. Otherwise the user is only able to view the help system and status of NoCAT.

When using CoNAT, it checks the values in its settings file, and only presents options based on what it finds within that file and also by checking that certain files used by NoCatAuth exist. Therefore, if the settings file contains locations for both the gateway and the authentication server, but only the “nocat.conf” file exists in the gateway’s folder then only the gateways configuration information is available, the user is not able to update the Authentication Servers configuration or update user accounts.

CoNAT also includes a comprehensive help system providing instructions on what each option within the interface does. When editing configuration settings, Javascript functions control what boxes should contain, as some values should only have numbers in, others IP addresses and some email addresses. When entering port numbers, email addresses, URL’s (Universal Resource Location) and field information, the user is able to manually edit these or use the provided wizard. When updating port numbers, information on the current port can be provided. When updating URL’s, the user is able to view the URL.

This level of control is included to aid the user in making the correct choices, and also for making sure that rules for the structure of the configuration file are strictly followed.

5.4.2 CoNAT’s Requirements

CoNAT has been built to run from any Linux box supporting PHP and a working implementation of NoCatAuth Gateway, Authentication Server or both. To fully utilise the software, the Linux box should also be running MySQL database and a SSL (Secure Socket Layer) based web server such as Apache. Client Requirements are very minimal, only requiring a JavaScript (and if hosted securely SSL) enabled browser.

6 Testing Strategies and Methods

6.1 Simple Hotspot

The simple Hotspot implementation was tested via a wireless enabled Centrino laptop (as specified in the Appendices). Key tests included connecting to the hotspot, disconnecting from the hotspot and also placing the laptop into hibernate power saving mode.

When NoCatAuth was installed further tests were conducted with NoCatAuth working as an Open Gateway. Tests were made to make sure a connection could be maintained, that the browser was redirected upon first launch and that hibernating did not cause any issue.

The hotspot has also been tested using some of the testing methods outlined in the paper “Hotspot in a Box” by Ziakas.

6.2 CoNAT

CoNAT has been extensively tested with the following combinations:

- Only the Gateway is installed.
- Only the AuthServer is installed.
- Both the Gateway and AuthServer installed.
- Incorrectly specified Gateway or/and AuthServer locations.

Tests were applied relevant to the installed version and component of NoCatAuth when and where appropriate. The NoCatAuth Authentication Server was only installed for development of CoNAT and was only ever used to confirm that the application’s configuration could be changed and that it could be stopped, started and restarted.

CoNAT was tested running on a Red Hat Linux 9 desktop with Apache 2.0.40, PHP 4.2.2 and NoCatAuth 0.82 installed.

CoNAT’s interface has been tested on two different browsers: Microsoft Internet Explorer 6.0 running on Windows XP Professional; and Mozilla 1.2.1 running on the Red Hat desktop. Both browsers have been configured with CSS (Cascading Style Sheets) and JavaScript enabled.

Testing of CoNAT also took place both within the NoCatAuth protected network (IP address range 192.168.1.x) and externally from the Internet.

A test plan was created, of which the results for each test are included in the appendices. In addition to these test results, example use cases are also included; these were worked through to further test the operation of CoNAT during its operation.

7 Evaluation

Products such as NoCatAuth provide a reliable and easy way to maintain a Hotspot. When using Hotspots, users are able to access their own networks resources as if they were actually within their networks. Billing options within Hotspots are comprehensive, and often customised to the location and/or owner of them.

CoNat provides an efficient and cost effective solution to managing NoCatAuth. It has been extensively tested and appears able to cater for any configuration currently possible with NoCatAuth. It integrates powerful technologies like PHP and JavaScript to aid in the configuration of NoCatAuth. CoNAT allows any Linux user with a basic knowledge of system administration to administer NoCatAuth from anywhere they wish.

CoNAT is reasonably secure storing the settings away from the web servers folders, thus meaning that someone without username / password access is unable to download the settings file. It can easily be made more secure by installing it on a Secure Web Server and applying for a SSL certificate for it. CoNAT also provides the ability for future improvements and upgrades as NoCatAuth improves over time.

CoNAT currently only supports a MySQL database as other data sources appear to have many more settings and rely heavily on further system configuration. NoCatAuth includes all the necessary database schema's as standard, and CoNAT's support for MySQL is based around this.

CoNAT will also stop or restart all copies of NoCatAuth it finds running. It is possible that the user may only want to stop the Gateway, when running both the Gateway and the Authentication Server, controlling both could have an adverse effect on the system and its performance.

Further improvements to CoNAT could include the ability to use Linux Login names and passwords to login, allowing different network Administrators access to the system. Other features could also include the ability to manage better the messages and pages displayed to users on connecting to the Hotspot, providing email alerts when problems occur, the ability to view the log file for NoCatAuth and most importantly control which NoCatAuth installation is started, stopped or restarted.

At present, it is also possible to log in using the username and password specified in the settings file from several different computers at the same time, this can cause a problem when editing settings. A control to limit the number of connected users would be needed.

Due to extensive research and reading upon using Linux and PHP prior to implementation only some small issues occurred:

- Correctly configuring the DHCPD settings file so that clients IP addresses could be re-leased more frequently, as the IP address space only allowed for 253 clients (255 total addresses less 1 for the AP and 1 for the server).
- Configuring the DNS server to provide correct top level domain name caching.
- Configuring CoNAT's code so that it would not save CoNAT specific settings to NoCatAuth's configuration files.
- Producing HTML (Hyper Text Markup Language) code that was valid on both Linux and Windows based machines, particularly at issue where Linux's processing of Form's.
- Correctly configuring and executing SuDo to run the nocat.rc script to start, stop and restart NoCatAuth.

Writing elements such as the installation script for CoNAT, pop-up entry boxes and MySQL implementation was made easier because I have had previous experience in these area's through coding simple scripts in Linux and using pop-up's and MySQL in previous university assignments.

Through developing and investigating the core technologies discussed my knowledge and understanding of Linux, Wireless networking and PHP has improved dramatically. I have also learnt many new technologies and systems such as NoCatAuth, IPTables and DNS systems.

I have also improved my time keeping through the use of the gannt chart (see appendices) and my ability to research key areas to find and retrieve information useful to the development of a software application. I feel that CoNAT provides a valuable tool in the management of NoCatAuth based Hotspots, and as such should prove useful for myself in the future. I have also emailed the Authors and users of the NoCatAuth software and offered CoNAT for download, within the first 24 hours of this email, 30 copies of the software have been downloaded.

As it became apparent that developing further billing solutions due to the many existing platforms and products, it was decided to rename the project to better reflect what has been achieved, a useful management system for NoCatAuth.

Bibliography

Books

Flickenger, R, 2003. Building Wireless Community Networks. 2nd Edition. Sebastopol, CA, USA: O'Reilly.

Gast, M. S., 2002. 802.11 Wireless Networks – The Definitive Guide. Sebastopol, CA, USA: O'Reilly.

Tanenbaum, A. S., 2003. Computer Networks. 4th Edition. New Jersey, USA: Prentice Hall.

Vines, R. D., 2002. Wireless Security Essentials: Defending Mobile Systems from Data Piracy. USA: Wiley.

Papers

Ziakas, D., et al, May 2003. "Hotspot in a Box": Creating Publicly available WLAN's. Rev 2.0, Intel Corp. UK.

Matsunaga, Y., et al, September 19th 2003. Secure Authentication System for Public WLAN Roaming. WMASH '03, 113 - 121.

Balachandran, A., et al, September 19th 2003. Wireless Hotspots: Current Challenges and Future Directions. WMASH '03, 1 - 9.

Graham J. W., November 20th – 23rd 2002. Authenticating Public Access Networking. SIGUCCS '02, 247 - 248.

Zhang, J., et al, April 2002. Virtual Operator based AAA in Wireless LAN Hot Spots with Ad-hoc Networking Support. Mobile Computing and Communications Review, Vol. 6, Number 3, 10 - 21.

Godber, A., et al, September 28th 2002. Secure Wireless Gateway. WiSe '02, 41 - 46.

Electronic

Kershaw, M., 2003, Linux-Powered Wireless Hot Spots, Linux journal, Available from: http://www.linuxjournal.com/article.php?sid=6887
[25/4/2004]

Aptilo Networks, 2004, Aptilo PWLAN Solutions Overview, Aptilo, Available from: http://www.aptilo.com/pages/offerings_1_1.html
[25/4/2004]

Aptilo Networks, 2004, Aptilo PWLAN Solutions Service Management Platform, Aptilo, Available from: http://www.aptilo.com/pages/offerings_1_2.html
[25/4/2004]

Aptilo Networks, 2004, Aptilo PWLAN Solutions Access Gateway, Aptilo, Available from: http://www.aptilo.com/pages/offerings_1_3.html
[25/4/2004]

Birdstep Technology, 2001, Birdstep IP Zone Server, Birdstep, Available from: http://www.birdstep.com/collaterals/br_ipzs.pdf
[25/4/2004]

ipUnplugged, 2004, ipUnplugged Internet Access Control, ipUnplugged, Available from:
http://www.ipunplugged.com/internetaccesscontrol.asp?mi=2.2
[25/4/2004]

NoCatAuth, 2003, NoCatAuth Source Code, NoCat, Available from: http://nocat.net/download/NoCatAuth/
[25/4/2004]

Gunter, M., 2003, NoCatAuth Gateway Server Configuration, Wi-fi Planet, Available from: http://www.wi-fiplanet.com/tutorials/article.php/3111111
[25/4/2004]

Gunter, M., 2003, NoCatAuth Gateway Server Configuration, Wi-fi Planet, Available from: http://www.wi-fiplanet.com/tutorials/article.php/3111111
[25/4/2004]

Gunter, M., 2003, NoCatAuth Authentication Server Configuration, Wi-fi Planet,
Available from: http://www.wi-fiplanet.com/tutorials/article.php/3286631
[25/4/2004]

Schuyler, E., 2001, Recipe for a Linux 802.11b Home Network, O'Reilly, Available from:
http://www.oreillynet.com/pub/a/wireless/2001/03/06/recipe.html
[25/4/2004]

Norrish, J., 2001, DNS Howto – A Resolving, caching, name server, Linux.com, Available from:
http://www.linux.com/howtos/DNS-HOWTO-3.shtml [25/4/2004]

Norrish, J., 2001, Linux IP Masquerade HOWTO - Configuring IP Forwarding Policies, Linux.org, Available from:
http://www.linux.org/docs/ldp/howto/IP-Masquerade-HOWTO/firewall-examples.html [25/4/2004]

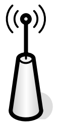
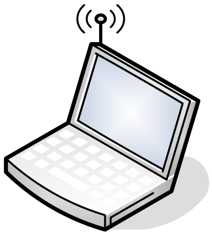
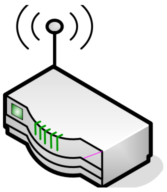
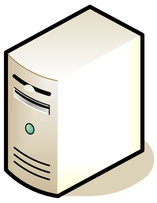
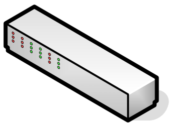
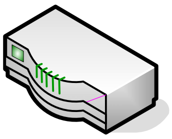
Saunders, J., 2004, Seamless Roaming Between Wireless Networks, Available from: http://www.jlsnet.co.uk/index.php?tab=3&page=projects_netswap
[20/7/2005]

Glossary of Terms

AP	Access Point
BASH	Bourne Again SHell
BIND	Berkeley Internet Name Daemon
BSS	Basic Service Set
CAT5	Category 5
CRC	Cyclic Redundancy Check
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CSS	Cascading Style Sheets
DCF	Digital Coordination Function
DHCP	Dynamic Host Control Protocol
DHCPD	DHCP Daemon
DIX	DEC (Digital Equipment Corp.), Intel and Xerox
DNS	Domain Name Service
DOS	Denial of Service
DSSS	Direct-sequence spread-spectrum
ESS	Extended Service Set
ESSID	Extended Service Set Identifier
FHSS	Frequency Hopping Spread-spectrum
FTP	File Transfer Protocol
GbE	Gigabit Ethernet
Gbps	Gigabits per second
GPRS	General Packet Radio Service
GSM	Global Mobile System
HR-DSSS	High Rate DSSS
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
HTTPD	Hyper Text Transfer Protocol Daemon
IBSS	Independent Basic Service Set
IEEE	Institute of Electrical & Electronics Engineer Standards Committee
IP	Internet Protocol
IR	Infrared
ISDN	Integrated Services Digital Network
Kbps	Kilobits per second
LAN	Local Area Network
LLC	Logical Link Control
MAC	Medium Access Control
MAN	Metropolitan Area Networks
Mbps	Megabits per second
MD5	Message Digest 5
NAT	Network Address Translation
NIC	Network Interface Card

OFDM	Orthogonal Frequency Divisional Multiplexing
OS	Operating System
PARC	Palo Alto Research Centre
PC	Personal Computer
PCF	Point Coordination Function
PCi	Peripheral Component Interconnect
PCMCIA	Personal Computer Memory Card International Association
PHP	Hypertext Pre-processor
PLAN	Personal Local Area Network
RADIUS	Remote Authentication Dial-In User Service.
SIM	Subscriber Identity Module
SSL	Secure Socket Layer
SuDo	Super User DO
TCP	Transmission Control Protocol
UMTS	Universal Mobile Telecommunications Standard
URL	Universal Resource Location
USB	Universal Serial Bus
VPN	Virtual Private Networking
WAN	Wide Area Network
WEP	Wired Equivalent Policy
Wi-Fi	Wireless Fidelity
WWAN	Wireless Wide Area Network

Glossary of Symbols

	Access Point
	Wireless enabled Laptop
	Wireless Router
	Server
	Switch
	Modem

Appendix A – System Specifications

Networking Equipment

- Intel PRO/Wireless 2011B LAN Access Point – IEEE 802.11b
- Various length CAT 5 network cables (Crossed and straight through)

PC Specification

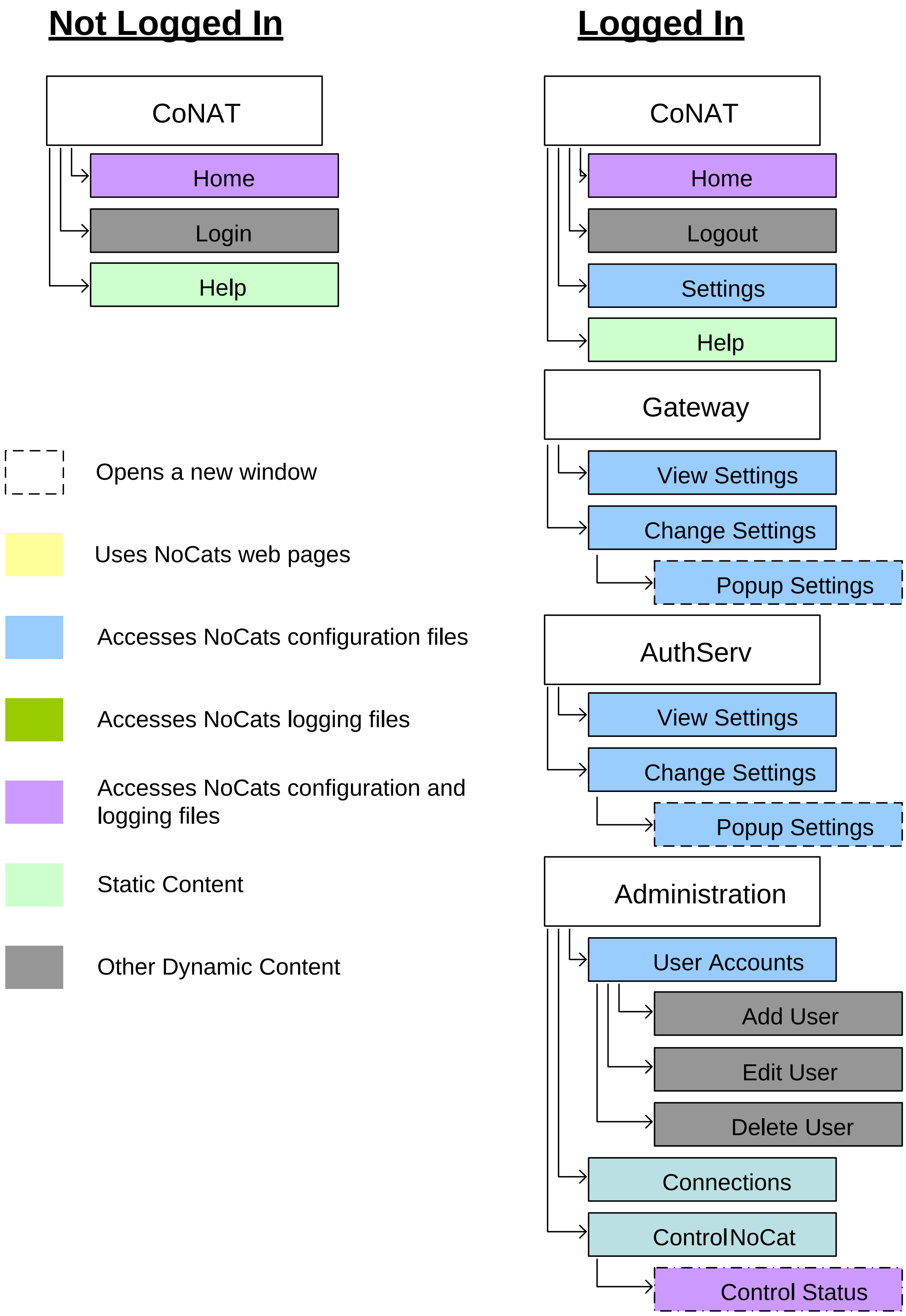
- Intel Pentium III 800 Mhz with 256KB Cache CPU
- Intel VC820 Mainboard
- 128 MB RDRAM (2 x 64 MB Modules)
- 10 GB HDD
- 1.44 MB Floppy Drive
- 48 x CD Drive
- 2 x CDRW Drive
- 64 MB ATI Video Card
- Realtek based PCI Network Card running @ 100 MBps
- Realtek based PCI Network Card running @ 10 MBps

Laptop Specification

- Sony Vaio TR1MP with Centrino Mobile Technology (CMT)
- Intel Pentium M 900 MHz with 1MB Cache ULV CPU
- 768 MB DDR-266 Memory (1 x 256 MB and 1 x 512 MB Modules)
- 30 GB 1.8” HDD
- 16 x DVD / CDRW Combo drive
- Intel 855GM Graphics
- Intel 2100 Wireless MiniPCi Card (802.11b)
- Intel 100 Mbps Network Card
- Integrated Bluetooth module
- Windows XP Professional with Internet Explorer 6.0
- Intel ProSet Networking Software.
- MSN Messenger

Appendix B – Design Documentation

CoNAT Layout



Appendix C – Testing

General hotspot Tests

Association to AP:	**ESSID:**	Hotspot
	ESSID Broadcast:	Yes
	Approx. Distance	**Speed**
General Network Coverage:	5 metres	11 Mbps
	20 metres	11 Mbps
Channel:	11	
IP Address:	192.168.1.	
DNS Address:	192.168.1.1	
Default Gateway:	192.168.1.1	
Subnet mask:	255.255.255.0	
Access to internet: http://www.google.com/	Yes	
Access to secure web service: https://www.hotmail.com/	Yes	
Connect to MSN Messenger:	Yes	

NoCatAuth based tests

Connecting to the Gateway:	OK
Authenticating with Gateway:	OK
Access to internet: http://www.google.com/	OK, had to authenticate first
Access to secure web service: https://www.hotmail.com/	OK, was able to check emails
Connect to MSN Messenger:	Yes
Hibernate laptop, wait 2 minutes and re-try accessing internet:	OK

CoNAT tests

Welcome page is displayed on first visiting CoNAT site:	Yes
Successful Login	Yes
Failed Login – Invalid Password:	Error message referring to incorrect password is shown.
Failed Login – Invalid username:	Error message referring to incorrect username is shown.
Failed Login – Missing username / password:	Error message referring to missing username / password is shown.
Failed Login – Both username and password missing:	Error message referring to missing username and password is shown.
Logout:	OK, returns to welcome screen
Help – Logged out:	Relates to Logged out only

Help – Logged in:	Relates to Logged in only
Settings Displayed:	On first starting and when clicking
Locations can be inserted:	Yes, and Insert detected works
Username change:	OK – root
Password change:	OK – password123
Gateway Settings viewed:	Yes
Gateway Settings refreshed:	Yes, no update
Gateway Settings displayed for editing:	Yes
Settings can be updated:	Yes, changed Verbosity to 5
Detected Settings can be inserted:	Yes
Homepage can be visited:	Yes – http://nocat.net
Pop-ups displayed and work as expected:	Yes
Only Include or Exclude Ports can be set:	Yes, message box appears when both are set
AuthServ Settings viewed:	Yes
AuthServ Settings refreshed:	Yes, no update
AuthServ Settings displayed for editing:	Yes
Settings can be updated:	Yes, changed Verbosity to 4
Detected Settings can be inserted:	Yes
Homepage can be visited:	Yes, http://nocat.net
Pop-ups displayed and work as expected:	Yes
DataSource change dynamically changes screen:	Yes, screen changes size
Currently applied DataSource appears on page load:	Yes
User Accounts displayed if using DBI and MySQL:	Yes – MySQL running locally
Warning message displayed if not DBI / MySQL:	Yes, no valid data source
Account can be added:	Yes – Kevin added
Account can be edited:	Yes – Kevin changed to kevin123
Account can be deleted:	Yes – kevin123 removed
NoCatAuth Connections page correctly displayed:	Yes
MAC Address link can be followed:	Yes, only when wireless client connected
NoCatAuth can be restarted:	Yes
NoCatAuth can be stopped:	Yes
NoCatAuth can be started:	Yes, only once stopped

Use Cases

Basic Configuration:

1) Log into CoNAT with username: user and password: password
2) Select the settings option
3) Configure Gateway and AuthServer locations with the "Insert Detected" option.
4) Save Changes
5) Select View Gateway Settings
6) Select View AuthServer Settings
7) Log out of CoNAT

Advanced Configuration - Gateway:

1) Log into CoNAT with username: user and password: password
2) Select Edit Gateway Settings
3) Set the Verbosity to: 5
4) Change the Setting GatewayName to: test
5) Change the Setting HomePage to : http://www.google.com
6) Save the changes
7) Select Control NoCat
8) Click the Re-start button
9) Log out of CoNAT

Advanced Configuration – Auth Server:

1) Log into CoNAT with username: user and password: password
2) Select Edit AuthServer Settings
3) Set the Verbosity to: 5
4) Change the Setting HomePage to : http://www.google.com
5) Change the LoginGreeting to : Welcome to test hotspot
6) Save the changes
7) Select Control NoCat
8) Click the Re-start button
9) Log out of CoNAT

Add user:

1) Log into CoNAT with username: user and password: password
2) Select User Accounts
3) Click Add a user
4) Add a new user with the credentials:
   a. Username: testing
   b. Password: password
   c. Name: Testing
5) Save the changes
6) Log out of CoNAT

Appendix D – Configuration Files - simple Hotspot

DHCPD.conf

```
ddns-update-style ad-hoc;

subnet 192.168.1.0 netmask 255.255.255.0 {
      option subnet-mask 255.255.255.0;
      range dynamic-bootp 192.168.1.2 192.168.1.255;
      option routers 192.168.1.1;
      default-lease-time 86400;
      max-lease-time 86400;
      option domain-name-servers 192.168.1.1;

      option netbios-name-servers 192.168.1.1;
      option netbios-dd-server 192.168.1.1;
      option netbios-node-type 8;
      option netbios-scope "";
}
```

named.conf

```
## named.conf - configuration for bind
#
# Generated automatically by redhat-config-bind, alchemist et al.
# Any changes not supported by redhat-config-bind should be put
# in /etc/named.custom
#
controls {
   inet 127.0.0.1 allow { localhost; } keys { rndckey; };
};

include "/etc/named.custom";

include "/etc/rndc.key";

zone  "0.0.127.in-addr.arpa" {
      type master;
      file  "0.0.127.in-addr.arpa.zone";
};

zone  "localhost" {
      type master;
      file  "localhost.zone";
};
```

firewall.sh

```
#!/bin/sh
#
# Kevin Martin
# 14/1/2004
#
# Firewall NAT Script
#
# Loosely based on the Linux Online IP Policy
# http://www.linux.org/docs/ldp/howto/IP-Masquerade-HOWTO/firewall-examples.html
#
# Default settings

LOCAL_NET=192.168.1.0/24
LOCAL_IF=eth1
EXTERNAL_IF=eth0

IPTABLES=/sbin/iptables
LSMOD=/sbin/lsmod
DEPMOD=/sbin/depmod
MODPROBE=/sbin/modprobe
AWK=/bin/awk
SED=/bin/sed
IFCONFIG=/sbin/ifconfig
GREP=/bin/grep

# Begin Execution of script
echo "   External Interface:  $EXTERNAL_IF"
echo "   Internal Interface:  $LOCAL_IF"
echo "   "

echo "   Determining IP Information"
echo "   "
```

```
EXTERNAL_IP="`$IFCONFIG $EXTERNAL_IF | $AWK \
 /$EXTERNAL_IF/'{next}//{split($0,a,":");split(a[2],a," ");print a[1];exit}'`"

LOCAL_IP="`$IFCONFIG $LOCAL_IF | $AWK \
 /$LOCAL_IF/'{next}//{split($0,a,":");split(a[2],a," ");print a[1];exit}'`"

echo "   External IP Address: $EXTERNAL_IP"
echo "   Internal IP Addresses: $LOCAL_NET"
echo "   Internal IP Address: $LOCAL_IP"
echo "   "

UNIVERSE=0.0.0.0/0
echo "   Universe: $UNIVERSE"
echo "   "

echo "   Loading Modules"

$DEPMOD -a

if [ -z "` $LSMOD | $GREP ip_tables | $AWK {'print $1'} `" ]; then
      $MODPROBE ip_tables
      echo -e "    Loaded : ip_tables"
else
      echo -e "    Already Loaded : ip_tables"
fi
if [ -z "` $LSMOD | $GREP ip_conntrack | $AWK {'print $1'} `" ]; then
      $MODPROBE ip_conntrack
      echo -e "    Loaded : ip_conntrack"
else
      echo -e "    Already Loaded : ip_conntrack"
fi
if [ -z "` $LSMOD | $GREP ip_conntrack_ftp | $AWK {'print $1'} `" ]; then
      $MODPROBE ip_conntrack_ftp
      echo -e "    Loaded : ip_conntrack_ftp"
else
      echo -e "    Already Loaded : ip_conntrack_ftp"
fi
if [ -z "` $LSMOD | $GREP iptable_nat | $AWK {'print $1'} `" ]; then
      $MODPROBE iptable_nat
      echo -e "    Loaded : iptable_nat"
else
      echo -e "    Already Loaded : iptable_nat"
fi
if [ -z "` $LSMOD | $GREP ip_nat_ftp | $AWK {'print $1'} `" ]; then
      $MODPROBE ip_nat_ftp
      echo -e "    Loaded : ip_nat_ftp"
else
      echo -e "    Already Loaded : ip_nat_ftp"
fi
if [ -z "` $LSMOD | $GREP ip_nat_snmp_basic | $AWK {'print $1'} `" ]; then
      $MODPROBE ip_nat_snmp_basic
      echo -e "    Loaded : ip_nat_snmp_basic"
else
      echo -e "    Already Loaded : ip_nat_snmp_basic"
fi
if [ -z "` $LSMOD | $GREP iptable_mangle | $AWK {'print $1'} `" ]; then
      $MODPROBE iptable_mangle
      echo -e "    Loaded : iptable_mangle"
else
      echo -e "    Already Loaded : iptable_mangle"
fi
if [ -z "` $LSMOD | $GREP ip_nat_irc | $AWK {'print $1'} `" ]; then
      $MODPROBE ip_nat_irc
      echo -e "    Loaded : ip_nat_irc"
else
      echo -e "    Already Loaded : ip_nat_irc"
fi
```

```
echo "   "
echo "   Turning on Forwarding"
echo "1" > /proc/sys/net/ipv4/ip_forward
echo "   "

echo "   Enabling DynamicAddr"
echo "1" > /proc/sys/net/ipv4/ip_dynaddr
echo "   "

echo "   Clearing any existing rules and Applying Default Policy"
$IPTABLES -P FORWARD DROP
$IPTABLES -F FORWARD
$IPTABLES -P INPUT DROP
$IPTABLES -F INPUT
$IPTABLES -P OUTPUT DROP
$IPTABLES -F OUTPUT
$IPTABLES -F -t nat


if [ -n "`$IPTABLES -L | $GREP drop-and-log-it`" ]; then
  $IPTABLES -F drop-and-log-it
fi

$IPTABLES -F
$IPTABLES -X
$IPTABLES -Z
echo "   "

echo "   Creating a DROP Chain"
$IPTABLES -N drop-and-log-it
$IPTABLES -A drop-and-log-it -j LOG --log-level info
$IPTABLES -A drop-and-log-it -j REJECT
echo "   "
echo "   Processing INPUT Rules"
# Allow all Loopback activity
$IPTABLES -A INPUT -i lo -s $UNIVERSE -d $UNIVERSE -j ACCEPT

# Allow all Local to see server
$IPTABLES -A INPUT -i $LOCAL_IF -s $LOCAL_NET -d $UNIVERSE -j ACCEPT
$IPTABLES -A INPUT -i $LOCAL_IF -s $LOCAL_NET -d $LOCAL_IP -j ACCEPT

# Deny External Interface claiming to be local
$IPTABLES -A INPUT -i $EXTERNAL_IF -s $LOCAL_NET -d $UNIVERSE -j drop-and-log-it

# Allow related traffic back in
$IPTABLES -A INPUT -i $EXTERNAL_IF -s $UNIVERSE -d $EXTERNAL_IP -m state --state \
 ESTABLISHED,RELATED -j ACCEPT

# Catch all other traffic
$IPTABLES -A INPUT -s $UNIVERSE -d $UNIVERSE -j drop-and-log-it

echo "   Processing OUTPUT Rules"

# Allow all Loopback activity
$IPTABLES -A OUTPUT -o lo -s $UNIVERSE -d $UNIVERSE -j ACCEPT

# Allow all Local to see all local
$IPTABLES -A OUTPUT -o $LOCAL_IF -s $EXTERNAL_IP -d $LOCAL_NET -j DROP
$IPTABLES -A OUTPUT -o $LOCAL_IF -s $LOCAL_IP -d $LOCAL_NET -j ACCEPT

# Stop anything coming into the network from external
$IPTABLES -A OUTPUT -o $EXTERNAL_IF -s $UNIVERSE -d $LOCAL_NET -j drop-and-log-it

# Allow Anything else on External Interface
$IPTABLES -A OUTPUT -o $EXTERNAL_IF -s $EXTERNAL_IP -d $UNIVERSE -j ACCEPT
```

```
# Otherwise drop everything else
$IPTABLES -A OUTPUT -s $UNIVERSE -d $UNIVERSE -j drop-and-log-it

echo "   Processing FORWARD Rules"

# Allow related traffic forward
$IPTABLES -A FORWARD -i $EXTERNAL_IF -o $LOCAL_IF -m state --state ESTABLISHED,RELATED \
 -j ACCEPT
$IPTABLES -A FORWARD -i $LOCAL_IF -o $EXTERNAL_IF -j ACCEPT

# Otherwise stop all other
$IPTABLES -A FORWARD -j drop-and-log-it

echo "   Setting PREROUTING options."

$IPTABLES -t nat -A POSTROUTING -o $EXTERNAL_IF -s $LOCAL_NET -j SNAT --to $EXTERNAL_IP

echo "   Complete"
```

Appendix E – Configuration Files – NoCat

install

```
#!/bin/sh

function top {
      clear
      echo "CoNAT INSTALL"
      echo "-------------"
      echo ""
}

nocat_file="nocat.rc"
sudoers_file="sudoers"
settings_file="settings.txt"
WHOAMINOW=`whoami`

if [ $WHOAMINOW != "root" ]; then
      top
      echo  "You are NOT root. Please login as root to install CoNAT."
      exit 0
fi

top
echo "Enter the installation location for main CoNAT Files (including end '/'): "
echo -n "[Default is /usr/local/conat/] "
read c_path

if [ "$c_path" == "" ]; then
      c_path="/usr/local/conat/"
fi
```

```
if [ -e $c_path ]; then
      top
      echo "Path already exists, Enter 'y' to continue"
      read -n1 ans
      if [ "$ans" != "y" ]; then
            clear
            exit 0;
      fi
fi

top
echo "Enter your web server's 'htdocs' location (including end '/'): "
echo -n "[Default is /var/www/html/] "
read h_path

if [ "$h_path" == "" ]; then
      h_path="/var/www/html/"
fi

while [ ! -e $h_path ]
do
      top
      echo "Path $h_path not found."
      echo "Enter your web server's 'htdocs' location (including end '/'): "
      echo -n "[Default is /var/www/html/] "
      read h_path

      if [ "$h_path" == "" ]; then
            h_path="/var/www/html/"
      fi
done

top
echo "Enter your web server's user name : "
echo -n "[Default is 'apache'] "
read w_username

if [ "$w_username" == "" ]; then
      w_username="apache"
fi

top
echo "Enter the name of the folder you want to create in $h_path to store conat's files: "
echo -n "[Default is 'conat'] "
read conat

if [ "$conat" == "" ]; then
      conat="conat"
fi

if [ -e $h_path$conat ]; then
      top
      echo "Folder $h_path$conat already exists, Enter 'y' to continue"
      read -n1 ans
      if [ "$ans" != "y" ]; then
            clear
            exit 0;
      fi
fi

top
echo "Enter the location of sudoers file : "
echo -n "[Default is '/etc/'] "
read sudoers_path

if [ "$sudoers_path" == "" ]; then
      sudoers_path="/etc/"
fi
```

```
while [ ! -e $sudoers_path$sudoers_file ]
do
      top
      echo "Sudoers File not found."
      echo "Enter the location of sudoers file (including end '/'): "
      echo -n "[Default is '/etc/'] "
      read sudoers_path

      if [ "$sudoers_path" == "" ]; then
            sudoers_path="/etc/"
      fi
done

top
echo "Install will create the following directories:"
echo ""
echo " $h_path$conat/"
echo " $c_path"
echo ""
echo "Install will also edit:"
echo ""
echo " $sudoers_path$sudoers_file"
echo ""
echo "Enter 'y' to continue"
read -n1 ans
if [ "$ans" != "y" ]; then
      clear
      exit 0;
fi

mkdir -p $c_path
cp -f ./files/*.* $c_path
```

```
mkdir -p $h_path$conat
mkdir -p $h_path$conat/images

cp -f ./images/*.* $h_path$conat/images/
cp -f ./html/*.* $h_path$conat/

sed "s|SED_PATH|$c_path|g" $h_path$conat/index.php > $h_path$conat/index.new
cp -f $h_path$conat/index.new $h_path$conat/index.php

chmod 700 $c_path$nocat_file
chmod 777 $c_path$settings_file
echo "$w_username ALL=NOPASSWD: $c_path$nocat_file" > temp
cat $sudoers_path$sudoers_file temp > temp2
cat temp2 > $sudoers_path$sudoers_file
rm -f temp temp2 $h_path$conat/index.new

top
echo "Install Complete"
echo ""
echo " Please open your browser, and browse to http://127.0.0.1/$conat"
echo ""
echo " Login with the username : user"
echo "        and the password : password"
read -n1 ans
clear
```

settings.txt

```
user_name   user
password    5f4dcc3b5aa765d61d8327deb882cf99
nocat_loc
auth_loc
first_time  yes
```

nocat.rc

```
#!/bin/sh

# Simple init script for starting
# the gateway service at boot time.
#
# Either add a call to it in /etc/rc.d/rc.local,
# or copy it to /etc/rc.d/init.d and symlink it
# to your runlevel.
#
# Edit the following line if you installed the
# nocat software somewhere else.
#
NCG=/usr/local/nocat/gw
NCA=/usr/local/nocat/authserv

case "$1" in
  start)
      case "$2" in
            gateway)
                  export PERL5LIB=$NCG/lib:$PERL5LIB
                  export NOCAT=$NCG/nocat.conf
                  echo "Starting NoCat gateway..."
                  rm -f $NCG/nocat.log
                  touch $NCG/nocat.log
                  $NCG/bin/gateway
                  ;;
            authserv)
                  export PERL5LIB=$NCA/lib:$PERL5LIB
                  export NOCAT=$NCA/nocat.conf
                  echo "Starting NoCat authserv..."
                  rm -f $NCA/nocat.log
                  touch $NCA/nocat.log
                  $NCA/bin/gateway
                  ;;
            *)
                  export PERL5LIB=$NCA/lib:$PERL5LIB
                  export NOCAT=$NCA/nocat.conf;
                  echo "Starting NoCat authserv..."
                  rm -f $NCA/nocat.log
                  touch $NCA/nocat.log
                  $NCA/bin/gateway

                  export PERL5LIB=$NCG/lib:$PERL5LIB
                  export NOCAT=$NCG/nocat.conf
                  echo "Starting NoCat gateway..."
                  rm -f $NCG/nocat.log
                  touch $NCG/nocat.log
                  $NCG/bin/gateway
                  ;;
      esac
      ;;
  stop)
      echo "Stopping All instances of Nocat..."
      killall gateway
      ;;
  restart)
        $0 stop $2
      sleep 1
        $0 start $2
        ;;
  *)
        echo "Usage: $0 {start|stop|restart} {gateway|authserv}"
        exit 1
esac

#
# Ende
#
```

gateway.conf

```
###### gateway.conf -- NoCatAuth Gateway Configuration.
#
# Format of this file is: <Directive> <Value>, one per
#   line. Trailing and leading whitespace is ignored. Any
#   line beginning with a punctuation character is assumed to
#   be a comment.

###### General settings.
#
# See the bottom of this file for options for logging to syslog.
#
# Log verbosity -- 0 is (almost) no logging. 10 is log
#   everything. 5 is probably a safe middle road.
#
Verbosity       10

##### Gateway application settings.
#
# GatewayName -- The name of this gateway, to be optionally displayed
#   on the splash and status pages. Any short string of text will do.
#
GatewayName the NoCat Network

##
#
# GatewayMode -- Determines the mode of operation of the gateway. Possible
#   values are:
#
#   Captive - Allow authentication against an auth service. LEGACY.
#   Passive - Like Captive, but YOU MUST USE THIS if your gateway
#                   is behind a NAT. Will work anyway if not. *RECOMMENDED*.
#   Open    - Simply require a user to view a splash page and accept
#               a use agreement.
#
# If Captive or Passive Mode is set, you will need to have values set for
#   AuthServiceAddr, AuthServiceURL, and LogoutURL. You will want to leave a
#   short value for LoginTimeout (probably <600).
#
# If Open Mode is set, you will need to have values set for SplashForm,
#   HomePage, and possibly DocumentRoot (or provide an absolute path for
#   SplashForm).  Also, you will want to set a large value for LoginTimeout
#   (probably >3600).
#
# Default
# GatewayMode     Passive
GatewayMode Open

##
# GatewayLog -- Optional.  If unset, messages will go to STDERR.
#
GatewayLog  /usr/local/nocat/nocat.log

##
# LoginTimeout - Number of seconds after a client's last
#   login/renewal to terminate their connection. Probably
#   don't want to set this to less than 60 or a lot of
#   bandwidth is likely to get consumed by the client's
#   renewal attempts. Defaults to 300 seconds.
#
# For Captive Mode, you want to set this to something
#   fairly short (like 10 minutes) to prevent connection
#   spoofing.
#
LoginTimeout      600
```

```
# For Open Mode portals, you probably want to comment out
#   the preceding and set LoginTimeout to
#   something large (like 86400, for one notification
#   per day).
#
# LoginTimeout    86400

###### Open Portal settings.
#
##
# HomePage -- The authservice's notion of a default
#   redirect.
#
HomePage    http://nocat.net/

# DocumentRoot -- Where all of the application templates (including
#   SplashPage) are hiding. Can be different from Apache's DocumentRoot.
#
DocumentRoot      /usr/local/nocat/htdocs

# SplashForm -- Form displayed to users on capture.
#
SplashForm  splash.html

# StatusForm -- Page displaying status of logged in users.
#
StatusForm  status.html


###### Active/Passive Portal settings.
#
##
# TrustedGroups - A list of groups registered with the auth server
#   that a user may claim membership in order to gain Member-class
#   access through this portal. The default magic value "Any" indicates
#   that a member of *any* group is granted member-class access from
#   this gateway.
#
# TrustedGroups   NoCat NYCWireless PersonalTelco
#
TrustedGroups Any

##
# Owners - Optional.  List all local "owner" class users here, separated
#   by spaces.  Owners typically get full bandwidth, and unrestricted
#   access to all network resources.
#
# Owners rob@nocat.net schuyler@nocat.net

#
# No restrictions on this network, so not needed.
#

##
# AuthServiceAddr - Required, for captive mode. Must be set to the address of
#   your authentication service. You must use an IP address
#   if DNS resolution isn't available at gateway startup.
#
# AuthServiceAddr 208.201.239.21
#
AuthServiceAddr   auth.nocat.net

#
# Not really going to test our AuthService, so we'll point to nocat's site.
#

##
# AuthServiceURL - HTTPS URL to the login script at the authservice.
```

```
#
AuthServiceURL  https://$AuthServiceAddr/cgi-bin/login

##
# LogoutURL - HTTP URL to redirect user after logout.
#
LogoutURL   https://$AuthServiceAddr/logout.html

### Network Topology
#
# ExternalDevice - Required if and only if NoCatAuth can't figure it out
#   from looking at your routing tables and picking the interface
#   that carries the default route. Must be set to the interface
#   connected to the Internet. Usually 'eth0' or 'eth1'
#   under Linux, or maybe even 'ppp0' if you're running
#   PPP or PPPoE.
#
# ExternalDevice  eth0

##
# InternalDevice - Required if and only if you have ethernet devices
#   on your gateway besides your wireless device and your 'Net connection.
#   Must be set to the interface connected to your local network, normally
#   your wireless card. In Linux, some wireless devices are named 'wvlan0'
#   or 'wlan0' rather than 'ethX'.
#
# InternalDevice  eth1

# Wi-FiPlanet specify that:
#
# ExternalDevice  eth1
# InternalDevice  eth0
#
# However, our configuration is actually (if NoCat cannot find it out):
```

```
#
# ExternalDevice  eth0
# InternalDevice  eth1
#

##
# LocalNetwork - Required if and only if NoCatAuth can't figure out
#   the network address of your local (probably wireless) network,
#   given your InternalDevice(s). Must be set to the network
#   address and net mask of your internal network. You
#   can use the number of bits in the netmask (e.g. /16, /24, etc.)
#   or the full x.x.x.x specification.
#
# LocalNetwork    10.0.1.0/24
#

#
# If NoCat cannot find it out:
#
# LocalNetwork    192.168.1.0/24
#

##
# DNSAddr - Optional. *If* you choose not to run DNS on your internal network,
#   specify the address(es) of one or more domain name server on the Internet
#   that wireless clients can use to get out. Should be the same DNS that your
#   DHCP server hands out. If left blank, NoCatAuth will presume that you
#   want to use whatever nameservers are listed in /etc/resolv.conf.
#
# DNSAddr 111.222.333.444

#
# Wi-FiPlanet recommend commenting this out too if using Cacheing DNS (which we are)
#
```

```
##
# AllowedWebHosts - Optional.  List any domains that you would like to
#   allow web access (TCP port 80 and 443) BEFORE logging in (this is the
#   pre-'skip' stage, so be careful about what you allow.)
#
# AllowedWebHosts nocat.net

##
# RouteOnly - Required only if you DO NOT want your gateway to act as a NAT.
#   Uncomment this only if you're running a strictly routed network, and
#   don't need the gateway to enable NAT for you.
#
# RouteOnly 1

##
# IgnoreMAC - Set this if and only if the NoCat gateway isn't directly
#   connected (or bridged at Layer 2) to your internal (usually wireless)
#   network. In that event, the gateway won't be able to match clients based
#   on MAC address, and will fall back to using IPs only. This is
#   theoretically less secure, as IP addresses are usually easier to spoof
#   than MAC addresses, so don't use this unless you know what you're doing.
#
# IgnoreMAC 1

##
# MembersOnly - Optional.  Uncomment this if you want to disable public
#   access (i.e. unauthenticated 'skip' button access).  You'll also want to
#   point AuthServiceURL somewhere that doesn't include a skip button (like
#   at your own Auth server.)
#
# MembersOnly     1

##
```

```
# IncludePorts - Optional.  Specify TCP ports to allow access to when
#   public class users login.  All others will be denied.
#
#   For a list of common services and their respective port numbers, see
#   your /etc/services file. Depending on your firewall, you might even
#   be able to specify said services here, instead of using port numbers.
#
# IncludePorts    22 80 443

##
# ExcludePorts - Optional.  Specify TCP ports to denied access to when
#   public class users login.  All others will be allowed.
#
#   Note that you should use either IncludePorts or ExcludePorts, but not
#   both.  If neither is specified, access is granted to all ports to
#   public class users.
#
#   You should *always* exclude port 25, unless you want to run an portal
#   for wanton spam sending. Users should have their own way of sending
#   mail. It sucks, but that's the way it is. Comment this out *only if*
#   you're using IncludePorts instead.
#
# ExcludePorts 23 25 111
#
ExcludePorts    25

####### Syslog Options -- alter these only if you want NoCat to log to the
#        system log!
#
# Log Facility - syslog or internal.  Internal sends log messages
#    using the GatewayLog or STDERR if GatewayLog is unset.  Syslog
#    sends all messages to the system log.
#
# LogFacility     internal
```

```
##
# SyslogSocket - inet or unix.  Inet connects to an inet socket returned
#    by getsrvbyname().  Unix connects to a unix domain socket returned by
#    _PATH_LOG in syslog.ph (typically /dev/log).  Defaults to unix.
#
# SyslogSocket unix

##
# SyslogOptions - Zero or more of the words pid, ndelay, cons, nowait
#    Defaults to "cons,pid".
#
# SyslogOptions cons,pid

##
# SyslogPriority - The syslog class of message to use:  In decreasing importance,
#    the typical priorities are EMERG, ALERT, CRIT, ERR, WARNING, NOTICE, INFO,
#    and DEBUG.  Defaults to INFO.
#
# SyslogPriority INFO

##
# SyslogFacility - The facility used to log messages.  Defaults to user.
# SyslogFacility user

##
# SyslogIdent - The ident of the program that is calling syslog.  This will
#    be prepended to every log entry made by NoCat.  Defaults to NoCat.
#
# SyslogIdent NoCat

###### Other Common Gateway Options. (stuff you probably won't have to change)
#
# ResetCmd, PermitCmd, DenyCmd -- Shell commands to reset,
#   open and close the firewall. You probably don't need to
#   change these.
#
# ResetCmd  initialize.fw
# PermitCmd access.fw permit $MAC $IP $Class
# DenyCmd   access.fw deny $MAC $IP $Class

##
# GatewayPort - The TCP port to bind the gateway
#   service to. 5280 is de-facto standard for NoCatAuth.
#   Change this only if you absolutely need to.
#
# GatewayPort     5280

##
# PGPKeyPath -- The directory in which PGP keys are stored.
#   NoCat tries to find this in the pgp/ directory above
#   the bin/ parent directory. Set this only if you put it
#   somewhere that NoCat doesn't expect.
#
# PGPKeyPath      /usr/local/nocat/pgp

##
# MessageVerify -- Shell command to verify a PGP signed
#   message. The actual message is delivered to the
#   command's standard input. NoCat tries to find gpg
#   and gpgv in your path. Set these only if you need to find
#   them elsewhere.
#
# GpgvPath  /usr/bin/gpgv
#
# MessageVerify   $GpgvPath --homedir=$PGPKeyPath 2>/dev/null

##
```

```
#
# IdleTimeout -- How often to check the ARP cache, in seconds,
#   for expiration of idle clients.
#
# MaxMissedARP -- How many times a client can be missing from
#   the ARP cache before we assume they've gone away, and log them
#   out. Set to 0 to disable logout based on ARP cache expiration.
#
# MaxMissedARP    2
#
# IdleTimeout   300

### Fin!
```

authserv.conf

```
###### authserv.conf -- NoCatAuth Authentication Service Configuration.
#
# Format of this file is: <Directive> <Value>, one per
#   line. Trailing and leading whitespace is ignored. Any
#   line beginning with a punctuation character is assumed to
#   be a comment.

###### General settings.
#
# Log verbosity -- 0 is (almost) no logging. 10 is log
#   everything. 5 is probably a safe middle road.
#
Verbosity       10

##
# PGPKeyPath -- The directory in which PGP keys are stored.
#   NoCat tries to find this in the pgp/ directory above
#   the bin/ parent directory. Set this only if you put it
#   somewhere that NoCat doesn't expect.
#
# PGPKeyPath      /usr/local/nocat/pgp


###### Authservice-specific settings.
#
# HomePage -- The authservice's notion of a default
#   redirect.
#
HomePage    http://nocat.net/

# DocumentRoot -- Where all of the application templates (including
#   SplashPage) are hiding. Can be different from Apache's DocumentRoot.
#
DocumentRoot      /usr/local/nocat/htdocs

##### Authservice authentication source.
#
# DataSource -- specifies what to authenticate against.
#   Possible values are DBI, Passwd, LDAP, RADIUS, PAM, Samba, IMAP, NIS.
#
DataSource  DBI

##
# Auth service database settings.
#
# If you select DataSource DBI, then Database, DB_User, and DB_Password
#   are required.
#
# Database is a DBI-style data source specification.
#
# For postgres support:
# Database  dbi:Pg:dbname=nocat
#
# For mysql support:
Database    dbi:mysql:database=nocat
DB_User           nocat
DB_Passwd   whodatatmydo?!

# Leave these settings intact, this will use the local MySQL database, and requires no other config.

## LDAP support. Requires Net::LDAP & IO::Socket::SSL to be installed from the CPAN.
#
# If you select DataSource LDAP, all of the following settings are required:
#
#    LDAP_Host - DNS name or IP Address of LDAP directory
#    LDAP_Base - the LDAP container for searching and creating users
#    LDAP_Admin_User - the fully distinguished name of the administrative user
#     NOTE: this user must be able to create users in the container specified above
#    LDAP_Admin_PW - the admin users password
#    LDAP_Hash_Passwords - Yes or No
#       - if passwords are to be MD5 hashed before being set in the directory
#    LDAP_Search_as_Admin - Yes or No
#       - "Yes" if all operations are to be done as the admin user, "No" if
#         everything but creation should be done as anonymous
#    LDAP_Filter - Attribute name containing user's ID, email address
#     or username.
#
# This version of LDAP.pm has been updated and tested against a Novell eDirectory
# LDAP server.  The login "unique ID" - the e-mail address - is stored as an
# attribute of the user, and the "name" provided by the user is used as the
# directory object name.
#
# Please send bug reports and patches.
#
# Still with this release, the admin tools don't fully work with LDAP support at
# the moment.
#
# LDAP_Host       ldap.mydomain.com
# LDAP_Base       ou=myContainer,o=universe
# LDAP_Admin_User cn=LDAPAdmin,o=universe
# LDAP_Admin_PW         ldapAdminSecret
# LDAP_Hash_Passwords   Yes
# LDAP_Search_as_Admin  Yes
# LDAP_Filter           mail

## RADIUS support. Requires Authen::Radius to be installed from the CPAN.
#
# Right now, this support is totally experimental. Please send bug reports
# and patches. The admin tools don't fully work with RADIUS support at the moment.
```

```
#
# The RADIUS_Host may by in a number of different formats and is required:
#
#   RADIUS_Host radius.nocat.net
#   RADIUS_Host radius1.nocat.net,radius2.nocat.net,radius3.nocat.net
#   RADIUS_Host radius1.nocat.net:1645,radius2.nocat.net:1812,radius3.nocat.net
#
# The previous three examples are 1 host and multiple hosts (can be any number of
# hosts separated by a comma) and finally with ports provided after a colon.  (If
# no port number is supplied, it uses the Authen::Radius default of the radius
# service in /etc/services or 1645.  Mixing entries with and without ports is
# fine.)  These examples require a RADIUS_Secret in the format:
#
# RADIUS_Secret   sHHHH
#
# The other format is to use the RADIUS_Host with a secret after
# the hostname seperated by a * such as the examples below.  This
# allows for different secrets on different hosts.
#
# RADIUS_Host radius1.nocat.net*secret1,radius2.nocat.net*secret2,radius3.nocat.net*secret3
#
# Alternatively, ports can also be used on any number of entries.
# If the secret is not present, it uses the RADIUS_Secret.
#
# RADIUS_Host radius1.nocat.net:1645*secret1,radius2.nocat.net:1812,radius3.nocat.net*secret3
#
# RADIUS_TimeOut is optional and defaults to the Authen::Radius
# default timeout.
#
# RADIUS_TimeOut 5
#
# Finally, RADIUS_Order controls the order in which RADIUS
# servers are used.  The acceptable values are "Ordered"
# (the default) and Random (which will share the load
# among the servers.
#
# RADIUS_Order Random
#     **** End RADIUS Configuration ****


## PAM support. Requires Authen::PAM to be installed from the CPAN.
#
# Make sure you have an /etc/pam.d/nocat or nocat line(s) in your /etc/pam.conf.
# See etc/pam.conf from this distribution for an example. The PAM_Service
# directive controls which PAM service NoCat attempts to authenticate against,
# but we don't recommend changing it unless you really know what you're doing
# with PAM. It defaults to "nocat". The admin tools don't work with PAM support
# at the moment.
#
# PAM_Service     nocat

## Samba support. Requires Authen::Smb to be installed from the CPAN.
#
# Samba_PDC and Samba_Domain are required. Samba_BDC is optional.
#
# Samba_Domain    MyWorkgroup
# Samba_PDC MyPrimaryDomainController
# Samba_BDC MyBackupDomainController

## IMAP support. Requires Net::IMAP::Simple to be installed from the CPAN.
#
# IMAP_Server is required. The admin tools don't work with this auth method.
#
# IMAP_Server     imap.yourdomain.net
# (or more likely:)
# IMAP_Server     localhost

## NIS support. Requires Net::NIS to be installed from the CPAN.
```

```
#
# The admin tools don't work with this auth source, surprise.
#
# DataSource NIS

## Alternately, you can use the Passwd data source.
#
# UserFile      /usr/local/nocat/etc/passwd
# GroupUserFile       /usr/local/nocat/etc/group
# GroupAdminFile    /usr/local/nocat/etc/groupadm
#
# The format of these files is as follows:
#
# In UserFile, each line is of the form <username>:<password>, where the
#   password is an MD5 digest of the user's actual password.
#
# In GroupUserFile and GroupAuthFile, each line is of the form
#   <group>:<user1>,<user2>,<user3>,...
#
# The UserFile may be updated with the bin/admintool script included in this
# distribution.

###### Auth service user table settings.
#
# UserTable names the table containing the user ID data.
#
# UserIDField names the column containing the ID that the
#    client uses to uniquely identifying themselves, i.e. their
#    e-mail address or username.
#
# UserPasswdField stores the user's MD5-hashed password.
#
# UserAuthField is deprecated and will go away.
#
```

```
UserTable   member
UserIDField login
UserPasswdField pass
UserAuthField   status
UserStampField    created

GroupTable      network
GroupIDField    network
GroupAdminField admin

####### Auth service web application settings.
#
# MinPasswdLength -- Enforced minimum user password length.
#   Not much other checking is done on the user's p/w.
#
MinPasswdLength   6

# MessageSign -- shell command to sign an auth notification
#   with. The message to be signed is written to the
#   command's standard in, and the signed message is read
#   from standard out.
#
# GpgPath   /usr/bin/gpg
#
# MessageSign     $GpgPath --clearsign --homedir=$PGPKeyPath -o-

# LocalGateway -- If you run auth service on the same subnet
#   (or host) as the gateway you need to specify the hostname
#   of the gateway. Otherwise omit it.  (Requires Net::Netmask)
#
# LocalGateway    192.168.1.7

#
# We are using a Local Gateway, therefore:
```

```
#
LocalGateway      192.168.1.1

# Auth service template names. See the individual templates
#   for details on what each one does.
#
LoginForm   login.html
LoginOKForm login_ok.html
FatalForm   fatal.html
ExpiredForm expired.html
RenewForm   renew.html
PassiveRenewForm renew_pasv.html

RegisterForm      register.html
RegisterOKForm    register_ok.html
RegisterFields    name url description

UpdateForm  update.html
UpdateFields      url description

###### Auth service user messages. Should be self-explanatory.
#
LoginGreeting   Greetings! Welcome to the NoCat Network.
LoginMissing    Please fill in all fields!
LoginBadUser    That e-mail address is unknown. Please try again.
LoginBadPass      That e-mail and password do not match. Please try again.
LoginBadStatus  Sorry, you are not a registered co-op member.

RegisterGreeting    Welcome! Please enter the following information to register.
RegisterMissing       Name, E-mail, and password fields must be filled in.
RegisterUserExists  Sorry, that e-mail address is already taken. Are you already registered?
RegisterBadUser       The e-mail address provided appears to be invalid. Did you spell it correctly?
RegisterInvalidPass All passwords must be at least six characters long.
RegisterPassNoMatch The passwords you provided do not match. Please try again.
```

```
RegisterSuccess       Congratulations, you have successfully registered.

UpdateGreeting        Enter your E-mail and password to update your info.
UpdateBadUser         That e-mail address is unknown. Please try again.
UpdateBadPass         That e-mail and password do not match. Please try again.
UpdateInvalidPass   New passwords must be at least six characters long.
UpdatePassNoMatch   The new passwords you provided do not match. Please try again.
UpdateSuccess         Congratulations, you have successfully updated your account.

######                                                                                                          Fin.
```

Appendix F – Proposal and Gantt Chart

Proposal

Project Title:

Advanced Billing Solutions for Wireless Networks / Hotspots

Project Aims and Objectives:

- Investigate the benefits of wireless network access
- Review of current access / billing methods and how these applications work (like NoCat, Aptilo's Captive Portal, etc)
- Other possible billing methods (like credit card, scratch card's, free access, account based, etc)
- Creation of new portal software or improvements to current system such as NoCat.
- Implementation of a wireless hotspot (using Access Point and a simple Linux / Windows Portal/Gateway)
- Interface implementation (Add in a user interface to configure what billing options should)

Organisation Involved and Main Contact for workplace based project

Not Applicable as yet

University computing resources required (e.g. hardware, software):

- Access to Apache or IIS Web server software, ASP / PHP support and Oracle / MySQL software – on a windows 2000 or better system.

Deliverables (e.g. User guide, specification, software):

- Specification
- User guide and installation guide
- ASP / Perl source code for web interface
- Voucher creation / credit card validation software.

Outline of method:

A full and in-depth investigation into current wireless systems will take place this will then lead to a review of current payment options, I will then devise based on questionnaires, etc what other payment options should feature. A simple wireless hotspot will then be created and work will then start on the implementation of the web interface with particular reference to usability with the most popular payment options implemented. Finally a full evaluation of the system and a demonstration of it will take place.

What is distinctive / honours worthy about the proposed project?

My project will try to offer features never seen before, it will also have a special feature in particular that will allow users to access certain web site's without submitting payment details, for example – A customer is visiting Borders Book Stores, and wants to Check details on a particular book, he would be able to access Borders Web site Free of charge and also maybe the publishers sites if applicable. All options on the software will be administratively controlled through a web interface allowing the owners of the network to enable and disable features and access as they see fit.

It will also tie in well with my chosen final year topics of: Usability Engineering, Advanced Computer Networks, Advanced Database Systems and Internet Application Development.

The project will create a product that will be fully marketable and available cross platform. It will also be a cutting edge solution in an area of the computing industry which is now starting to become very popular. It is for these reasons that I feel it is distinctive and honours worthy.

Gantt Chart