Virginia Tech ■ ECE/CS 4570: Wireless Networks and Mobile Systems ■ Spring 2006

Project P10 – A Wireless LAN Hot Spot

Report Due: May 3, 2006 at 4:00 p.m.

Part I – Objectives and Project Materials

Objective

The objective of this project is to understand how routing, IP firewalls, and IP masquerading (also known as network address and port translation) can be integrated to offer wireless connectivity or a "hot spot" service. Understanding of the following should result from this project.

- DHCP daemon use and configuration
- Use of iptables for basic firewalling and IP masquerading
- Configuring a computer running Linux to work as a router
- Basic web authentication using a web interface and JavaScript

Hardware

You will need to use the following hardware for this project.

- Notebook computer
- iPAQ handheld computer
- Intel Wireless Gateway
- Two 802.11b network interface cards (NICs)

Software

You will need to use the following software for this project.

- Linux on the notebook computer
- iptables on the notebook computer
- DHCPd on the notebook computer
- Apache Web Server on the notebook computer
- CGI development language (e.g., C++ or Perl) or PHP on the notebook computer
- HTML and JavaScript
- Internet Explorer under PocketPC on the iPAQ

Part II – The "Hot Spot" System

Congratulations, you have been hired by Stu's Pizza Shack to build and install a wireless hot spot for their pizzeria. The following is the network specification requested by the manager, Stu

1. The service should provide 802.11b wireless access.
2. Everything must work off of a single external IP address since Stu's Pizza Shack only one Internet connection with a single external address.
3. The service should be as simple for the customer to use as possible.
4. Provide user authentication since only paying customers should have access.

Technical Specification

For this project, you are to design and implement a "hot spot" gateway and network. A hot spot is a wireless local area network (WLAN) commonly used in small businesses and public areas to offer convenient wireless connectivity to customers and visitors. WLAN access offered at a coffee shop is an example of a hot spot service. The following are major concerns in a hot spot environment.

1. *Simplicity* – A user should be able to gain access using a standard notebook or handheld computer with an 802.11b NIC and dynamic host configuration (i.e., using DHCP).
2. *Security* – Only registered (paying or otherwise authorized) users should have access to the network. The use of one-way hashing hides passwords from prying eyes.
3. *Maintainability* – Since most restaurant employees are not your average Linux guru, ease of administration is a must.

Configuration Overview

To accomplish this task, you must use your notebook computer running Linux and your Intel Wireless Gateway to provide the basic hot spot service. For testing, you will use your Compaq iPAQ with an 802.11b NIC as the client. Figure 1 shows the network configuration for testing. Note that a "public Internet" will be provided to you during the demonstration. Also, your hot spot should be able to support additional wireless devices, as represented by the second iPAQ in Figure 1.

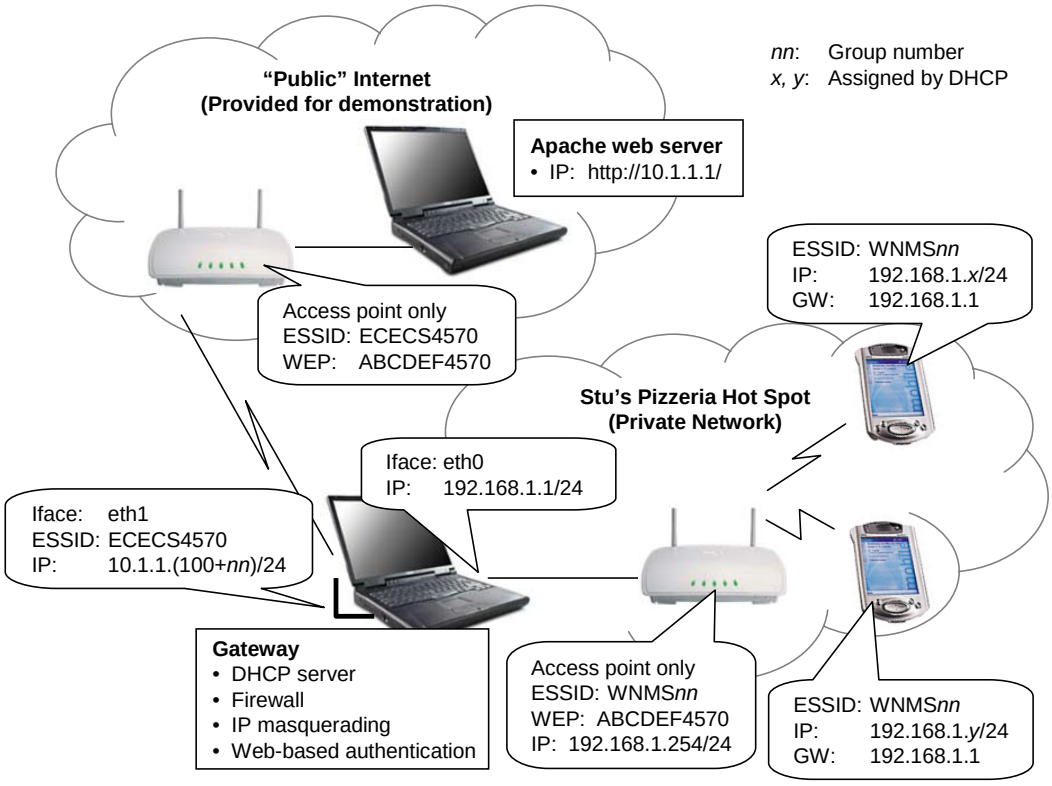

Figure 1. WLAN hot spot hardware configuration for demonstration.

The components shown in the configuration of Figure 1 are used as follows.

1. Intel Wireless Gateway
    a. The access point acts as an access point only. Disable all of the gateway functionality of the access point. In particular, the DHCP server on the access point must be disabled.
    b. Configure the access point to use the ESSID WNMS*nn*, where *nn* is your group number.
    c. Configure the access point to have an IP address of 192.168.1.254/24.
    d. The wireless clients (the iPAQs) use this as the access point to connect to the network.
2. Notebook Computer
    a. The notebook computer, running Linux, acts as the DHCP server, IP masquerading gateway, and authentication host.
    b. The DHCP server software of choice is the standard DHCP daemon, dhcpd, found with the Linux distribution on the notebook computer. (See /etc/dhcpd.conf for configuration.)
    c. The wired interface will act as the internal interface and connects to the access point with the cross-over cable. The internal interface has the address 192.168.1.1/24.
    d. The external interface, the wireless interface, has the address 10.1.1.<100+*nn*>/24, where *nn* is your group number, and will connect to the provided wireless network for the demonstration. Feel free to do your own testing using your own or the campus WLAN.
    e. Iptables will be used for firewalling and IP masquerading.
    f. The notebook provides authentication, as described below. It runs the Apache web server to aid in this function.
3. iPAQ
    a. The iPAQ serves as a test client. Your service should work for multiple simultaneous users, although the available equipment limits your testing to one client. If possible, use another wireless client, e.g., another iPAQ or a notebook, to test your system.

Authentication

The gateway (notebook computer) must perform authentication so that only recognized users may use the hot spot service. Figure 2 shows the high-level steps of the authentication process.

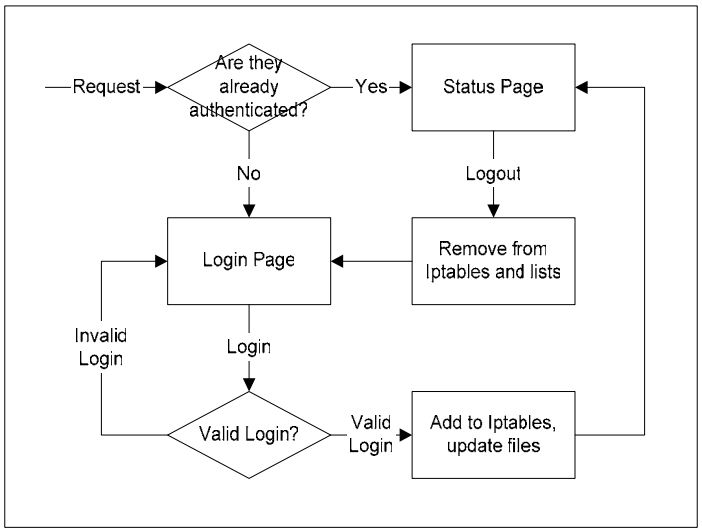

Figure 2. The authentication process.

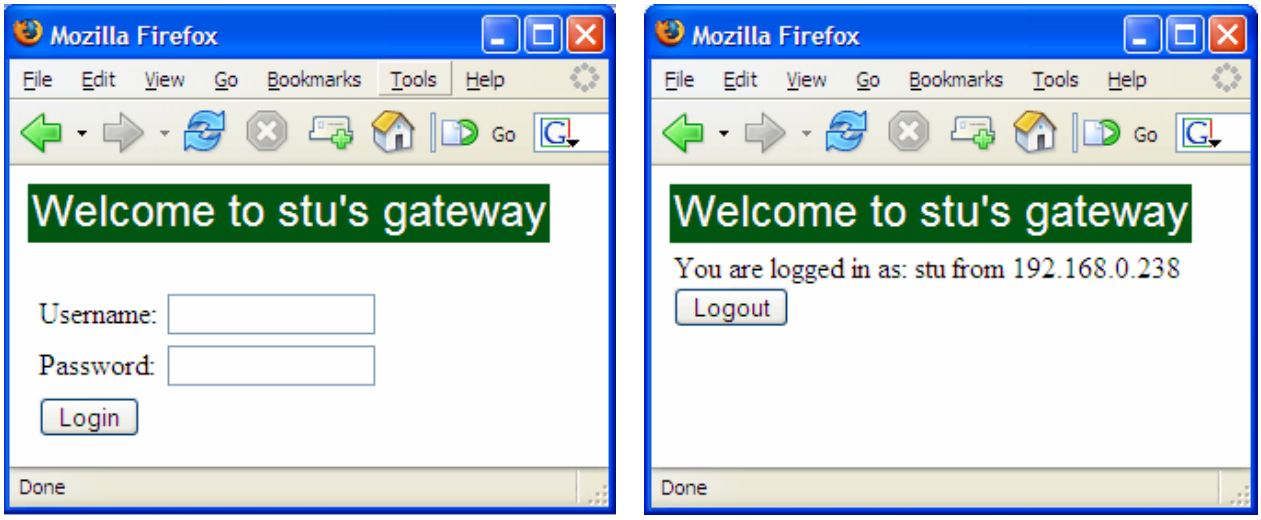

Figure 3. Example login page (left) and status page (right).

Prior to successfully authenticating:

1. All clients should be able to get an IP address from the gateway upon joining the network.
2. The client should not have access to the outside world; the only exception to this is allowing DNS requests out and DNS responses in and this feature is optional.
3. All HTTP requests should be redirected to the gateway's login page; example see Figure 3. All other requests should be blocked by the gateway.

Authentication:

1. The login page must provide the same functionality as shown in the left side of Figure 3.
2. When the user clicks the "Login" button, the client browser computes the login information and sends it to the gateway. Upon receipt, the gateway attempts to validate the credentials of the client and accepts or denies its request to login. If the credentials are not valid, the login page is, again, presented and a "Bad username/password" or similar error message should be included somewhere in the page. If the credentials are valid, the user should be directed to the status page, similar to that shown in the right side of Figure 3. If the login is successful, the status page must provide: (i) the IP address used by this client, and (ii) the username associated with this client. (Note that the right side of Figure 3 does not show the user name, but the user name is required.)

After authenticating successfully:

1. The authenticated host has full access to the outside world.
2. Anytime the host connects to the gateway, the status page (with an example shown in the right side of Figure 3) is displayed. The status page should tell the user who they are logged in as (the username) and where they are logged in from (the IP address) and should include a logout function.
3. If and when a user decides to logout, by clicking the logout button or link, the gateway removes the access rules from iptables and all instances of the user from the state files.

Technical Details

Due to the simplicity of sniffing wireless links, some protection needs to be added to the login process. This ensures that only those that pay for access get access; recent polls indicate that money is *good*, so the folks at Stu's want to get as much of it as possible. To protect passwords while they are floating around in the ether, you must employ some sort of security scheme. Secure HTTP, or https, is an option, but does not teach you much about the process of authentication and, also, requires access to a trusted certificate authority or pre-shared private keys. For this deployment, another choice is to use a one-way encoding scheme on the password to make sure that only the gateway knows what the value should be. Also, we'll be including a one-time-use, random number, called a *nonce*, to make sure we avoid any replay attacks. (This scheme has been used by various commercial services and products that do not use Secure HTTP.)

On the client side you will be using JavaScript to take a Message Digest 5 (MD5) hash of the password, concatenated with the nonce from the server. This will be encoded using Base 64 encoding as well. We call this the secret. Finally, the secret and the username are sent to the gateway for authentication.

The gateway will validate the nonce, lookup the password corresponding to your username and compute its version of the secret. If the two secrets match, the user is successfully authenticated and allowed to access the world. If they do not match, the username and/or password is invalid (or the functions are not properly implemented). In either case, once a nonce has been used, it cannot be used again.

Nonces and hashing are critical to this type of authentication method. Because of this, great care must be taken in how the nonces work. The first rule is that a nonce should only be valid for one authentication attempt. Each time the login page is loaded, a new nonce should be included. A nonce expires after a certain period of time and should be removed from the nonce file. This expiration time is set to 5 minutes, but should be easily changeable. Expired nonces should be removed from the list of valid nonces when the nonces are being evaluated, not when new ones are being added.

You will need four files for keeping track of different states.

- Password file – holds usernames and passwords
- Allow file – needed if you are using a cronjob to update iptables; the syntax is up to you
- Success file – keeps track of users that have been authenticated, holds username and IP address
- Nonce file – keeps track of active nonces; holds the IP address, nonce, and a sent timestamp

Network Configuration

The network should be configured as follows (see Figure 1). Also refer to the "Configuration Overview" section above.

1. Access Point
    i. Gateway functionality turned off
    ii. ESSID: WNMS<group number, 2 digits>
    iii. WEP: 64-bit, key: ABCDEF4570
2. Clients
    i. DHCP-assigned address range: 192.168.1.10–192.168.1.100/24
    ii. Gateway router: 192.168.1.1
3. Gateway
    i. Internal (wired) interface: 192.168.1.1/24
    ii. External (wireless) interface: 10.1.1.<100+group number>/24
    iii. Gateway router: 10.1.1.1

Firewall Configuration

Use the following information when configuring the firewall in the gateway notebook.

1. The policy for the filter:INPUT, filter:FORWARD, and filter:OUTPUT chains is DROP. All other chains have an ACCEPT policy.
2. Rules will be added to all of the filter chains, the nat:PREROUTING chain, and the nat:POSTROUTING chain. These are the only chains that should have rules added.
3. After a client is successfully authenticated, the client should have full access to the outside world through the gateway.

Shell Scripts

You will need to write the following two scripts for the gateway notebook.

1. Startup Script – The startup script should have all the components to initialize the interfaces, firewall, and services. The script should follow this order:
   - shutdown network interfaces,
   - clear the firewall,
   - setup the firewall,
   - bring up the network interfaces, and
   - start any needed services.
2. Shutdown Script – The shutdown script should set the firewall back to no firewall mode; all rules flushed, extra chains removed, and all polices set to ACCEPT. It should also shutdown the services needed for the project. It does not need to set the network interfaces back to their pre-project state.

Hints

Here are some hints related to your design, implementation, and testing.

1. Work to get DHCP and IP masquerading working before moving on to the authentication portion.
2. The web page root is /var/www/html. The HTTP daemon configuration file is /etc/httpd/conf/httpd.conf.
3. For your authentication scripts, iptables must be executed by root. Research `cron`, `sudo`, or `chmod` for ideas. (Running Apache as root is not an allowed option since this is a huge security hole.)
4. You do not need to recompile the kernel for this project. The provided kernel has iptables support.
5. All of the servers (Apache web server and dhcpd) and development software (PHP, Perl, and C++) are already installed on your notebook. The DHCP configuration is in dhcpd.conf and will need to be modified. The web server configuration file, httpd.conf, will need to be modified slightly to support the authentication function.
6. Packet capture utilities such as Ethereal and tcpdump can provide some help in debugging.
7. Functionality is the key. Make sure all your parts work together before you worry about aesthetics.
8. There is a size limit to the number of rules in iptables, be sure to flush the chains during development if you are not removing rules.

9. Test to make sure your firewall is working correctly. The 192.168.1.0/24 network should not be accessible from the outside world. No packets marked as being from 192.168.1.0/24 should be sent out onto the external network.
10. Backup any configuration files before modifying – just in case!

Optional Features

Here are some additional features that you may want to add. You can obtain full credit by just implementing the basic features. But, inclusion of any of these features will be considered in grading to compensate for at most 5 points for any weaknesses in the project or report. The maximum project grade is 100 points.

- Administration page – this is to be implemented separately from the login and status pages. It may include the following capabilities.
  - Ability to add users to the password file. (1 point)
  - Ability to remove users from the password file. (1 point)
  - User status listing all currently logged in users and how long they have been logged in. This must be separate from the status page that the client sees. (1 point)
  - Add insecure authentication to the administration page. (1 point)
  - Add secure authentication to the administration page (using the same method as used for the rest of the project) (2 points)
  - Impose time limits on users. After a user has been logged in for a specified (usually purchased) amount of time, they are automatically logged out (and forced to buy more time). (1 point)
  - Add a deny list based on IP address. This should be maintained through the administration page. (1 point)
- Implement a log file to record successful and unsuccessful login attempts; must include the time, username, IP address and success level. (1 point)

Note that credit will only be given for one type of administration authentication (xor).

Part III - Demonstration

Each group must demonstrate their working hot spot service during the week of May 2. Demonstration times will be scheduled in Blacksburg and at the NVC. This demonstration will consist of the following steps.

- The group will bring their notebook computer, fully configured for the hot spot service (as shown in Figure 1), to the scheduled testing session.
- The tester will provide an access point and a test host for the "public" Internet.
- The tester will test the hot spot service using one or more clients of his or her choosing. The testing will include access to the server in the "public" Internet.

In the demonstration, a generic client should be able to connect to your internal network, acquire an IP address using DHCP, authenticate itself to the hot spot service running on the notebook, get status information about the connection, access services in the outside network (e.g., an external web server), and log off of the network. Functionality is the priority. Each group should be prepared to demonstrate all working functionality, including any optional features.

Part IV – Deliverables, Report, and Grading Guidelines

Deliverables

You need to provide the following deliverables by the project due date.

1. The startup and shutdown shell scripts.
2. All script files, source files, and any other files that you have created to implement the authentication mechanism. All code, including scripts, should be thoroughly commented.
3. Ethereal traces that show the authentication process and the DHCP process. Perform the capture on the internal interface. The authentication capture should include the HTTP request, the authentication process, and the retransmission of the HTTP request and its fulfillment.
4. A project report, as described below. The report should be submitted as a Microsoft Word or Adobe Acrobat PDF file.

All deliverables should be submitted as a single .zip file named:

P10_Partner1LastName_Partner2LastName.zip

This file must be uploaded to Blackboard's Dropbox by the due date and time. Adhere to the submission guidelines provided earlier in the semester. Each group must submit one set of deliverables for the entire group.

Project Report

The project report should contain the following items in the order specified. Adhere to the page limits specified.

- Cover Page (1 page): Specify the course number and name; the project name (P10: Wireless Hot Spot Project); the name, student identification number, and email address of each group member; location (Blacksburg or NVC); and the date of submission.
- File List (≤1 page): Include a listing and brief description of all files included with the deliverables.
- Firewall (≤1.5 pages): Include a list and analysis of all firewall rules, (iptables –t filter –L; iptables –t nat –L; iptables –t mangle –L). Include a description of any problems that were encountered and how solutions were devised.
- Routing (≤1.5 pages): Include the routing table and a packet trace from an internal host to an external host. Document the packet trace pointing out the changes in packet headers as it traverses from the internal to external networks. Include any additional comments relevant to your design.
- Authentication (≤2 pages): Include an overview of the process and diagram showing the different stages of authentication, events, and relevant files. (Example: when the user clicks the 'login' button the information is passed to validate.php…) Explain your authentication method and why it was chosen. Finally, discuss all problems encountered and the corresponding resolutions.
- Optional Features (≤1 page): Include a list and brief discussion of any optional features that you implemented.
- Provide answers to the following questions (≤5 pages). Your answers should be concise, but complete.
    1. Identify at least two security issues related to your implementation.

2. Are there ways to avoid these security issues? If so, what are they? (These may be outside the scope of this project.)
3. What are at least two applications (or application layer protocols) that have issues with IP masquerading (these may helped by kernel modules)?
4. What were the major problems encountered during this project?
5. Were there any strange occurrences or other matters about which you wish to comment? (Provide details.)
6. Rate your change in understanding of each of the following concepts. Use the following scale: 1-still clueless … 5-I can explain it to my mom, 0-knew it already. (Your response will not be graded.)
    i. General firewall concepts
    ii. General routing/masquerading concepts
    iii. Use of IPTables for the above
    iv. Basic Linux network administration
    v. Basic Linux web administration
    vi. HTML
    vii. JavaScript
    viii. Server side processing method (PHP, Perl, etc.)

Grading

Grading will be based on the following factors. The project will be graded based on a maximum score of 100 points.

- Correct operation as demonstrated (70 points). Partial credit will be awarded for partial functionality. Be prepared to demonstrate all functionality that works, including optional features.
- Report content, including discussion of firewall methods, routing, authentication, and answers to the questions (25 points).
- Overall quality of the submission, including writing of the report, comments in code, etc. (5 points).
- Implementation of optional features can compensate for up to 5 points, depending on the number and difficulty of optional features. The overall grade cannot exceed 100 points.

Appendix A – Useful Sources

This appendix provides links to information that may be helpful to you in completing this project.

Linux Networking

- Basic Networking knowledge and basic Linux implementations
  http://www.linuxdocs.org/HOWTOs/Net-HOWTO/index.html
- Advanced Routing Howto – if you want to know more about what is going on underneath
  http://www.linuxdocs.org/HOWTOs/Adv-Routing-HOWTO.html

DHCPd

- dhcpd.conf man page

- DHCPd mini-howto
  http://www.tldp.org/HOWTO/mini/DHCP/x369.html
- Linux Magazine dhcpd configuration help
  http://www.linux-mag.com/2000-04/networknirvana_01.html

iptables

- iptables man page
- iptables home page
  http://www.netfilter.org/
- Iptables tutorial (very thorough)
  http://iptables-tutorial.frozentux.net/

Perl (version 5.8.0 comes with Red Hat 9.0)

- Perl books from O'Reilly
- Perl Homepage
  http://www.perl.com/

CGI

- Use those Google searching skills! There are plenty of online references.
- Apache documentation on CGI
  http://httpd.apache.org/docs/howto/cgi.html
- Decoding forms with CGI, Links to CGI libraries for C++, Perl, and Bash
  http://hoohoo.ncsa.uiuc.edu/cgi/forms.html

PHP (version 4.3.10 comes with Red Hat 9.0)

- PHP.net, the documentation section is excellent, including a basic tutorial for PHP programming and lots of example code
  http://www.php.net/
- PHP Manual (see sections on "Getting started – Introduction," "Getting started – A Simple Tutorial," "Language Reference," "Variables – Predefined Variables - $_SERVER, $_POST, $_GET," and "Filesystem Functions")
  http://www.php.net/docs.php

Apache (version 2.0.40 comes with Red Hat 9.0)

- Apache configuration help (a minor modification is required)
  http://builder.cnet.com/webbuilding/pages/Servers/Apache/ss02.html

HTML

- Basics
  http://www.htmlprimer.com/lesson1.shtml
- Forms
  http://www.htmlprimer.com/forms.shtml

Javascript

- Safari Books (VT Libraries link; on campus or through Lib proxy server)
  http://proquest.safaribooksonline.com/?uicode=vatech

Appendix B – Food for Thought (if your mind is still hungry)

Think about these issues.

- Stateless and stateful features of the Linux IP masquerading implementation
- Other uses for network address translation
- Security applications of IP masquerading
- Security shortcomings of IP masquerading
- Concerns with IP masquerading for TCP, UDP, ICMP, and other layers above IP