Test Lab Guide

Open BSD

Michael Pare <michael.pare@mymail.champlain.edu[1]> version 0.1, 2020-04-15

1. Abstract

This Test Lab Guide (TLG) will provide the user with step-by-step instructions on how to configure a basic network configuration using OpenBSD servers and an OpenBSD client. The resulting test lab environment will provide a stable base for building other test labs. It is recommended that users preserve the start of their test machines once this walk-through has been successfully completed. In a physical environment the hard drive of the machine can be imaged. In a virtual environment, the “snapshot” or equivalent feature can be used to preserve the current state of the operating system. Preserving the working conditions of the basic test lab will allow the user to experiment without fear of having to completely rebuild their environment. The ultimate goal of the TLG is to familiarize the user with the OpenBSD Operating system and how to deploy common network services based on that operating system, and ultimately enable the user to conduct their own experiments in a OpenBSD based environment.

2. Introduction

The purpose of the Test Lab Guide (TLG) documentation is to provide users with hands-on experience with current technologies in known working configurations. Using a TLG will tell you exactly what machines are needed, how they should be configured, and how to install and configure any additional software. Completing a TLG will allow you to get the entire experience of configuring different technologies from beginning to end. The purpose of this Test Lab Guide is to guide you in creating a Local Area Network (LAN) using machines running OpenBSD. Once complete, you will have a LAN that will have the following services:

- Dynamic Host Configuration Protocol (DHCP)
- Domain Name System (DNS)

[1] mailto:michael.pare@mymail.champlain.edu

- HTTP
- NAT (Network Address Translation)
- SMB (Samba)

Once you complete this lab, you may wish to save this initial configuration. How you save this configuration will depend on how you've created and deployed your test environment. If you are working in a physical environment, you can image the drives of each machine. If you are working in a virtual environment, you can take snapshots of each machine which will save all the current settings and configurations. Having a way to preserve the initial functioning base lab configuration is important because it allows you to quickly revert to this state. This can be helpful when you want a fresh start to test new products or if you make an error. These images or snapshots will allow you to focus on deploying new technologies successfully instead of having to constantly rebuild this base. It is because of this that you should image the drives or take snapshots of each machine upon completion of this TLG.

Note: If you are completing these TLG's in a virtual environment using VMWare and have access to an electronic copy of this document, and you have VMWare Tools installed on the Virtual Machine, utilize the ability to copy and paste text from the host machine to the VM. Copying and pasting will help to reduce typos and command errors.

- Highlight and right-click a command from this document
- Click Copy
- Right-click in the virtual machine where you would like to copy the text to and click Paste
- If you are working inside a terminal, you may only need to right click in order to copy the command over

3. Disclaimer

This website contains work created for informational purposes. Information may be out of date, or changed or updated without notice. By using this website, you

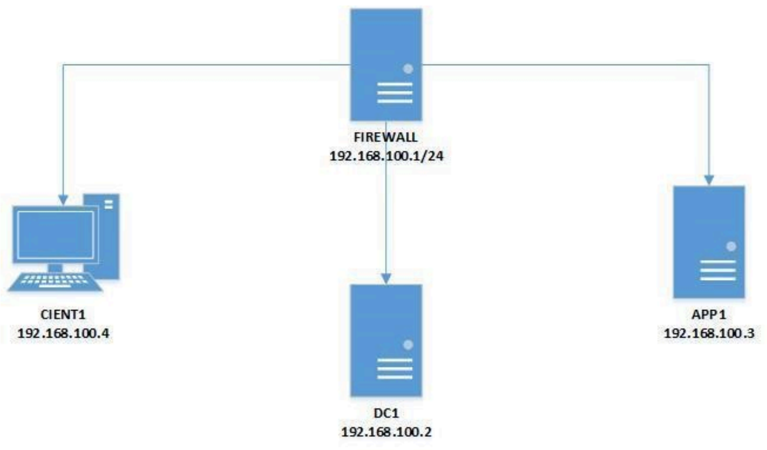
4. Overview of this Test Lab Guide

The OpenBSD TLG is comprised of:

- One server running OpenBSD named DC1 that is configured as the intranet Domain Name System (DNS), and Dynamic Host Configuration Protocol (DHCP) server
- One server running OpenBSD named FIREWALL that has two NIC’s and is configured to act as a router/firewall
- One client running OpenBSD named CLIENT1 that will have the ability to surf the web using Firefox
- One app server for running applications such as smb and http

The OpenBSD Transparent Proxy Server TLG is comprised of one subnet: * An intranet, known as the Busnet subnet (192.168.100.0/24) The machines on each of the respective subnets are connected using a physical hub or switch, or a virtual switch.

4.1. Network Design



Firewall

IP Address: 192.168.100.1/24

Hostname: firewall.college.com

The Firewall's purpose is to handle traffic between the Intranet and Internet. This means that it will act as a router directing traffic. There are two interfaces on Firewall, one is the external/Internet-facing called em0 and the other is the internal/Intranet-facing called em1.

DC1

IP Address: 192.168.100.2

Gateway: 192.168.100.1

Hostname: dc1.college.com

This will be configured as the Intranet's Domain Name System (DNS) and the Dynamic Host Configuration Protocol (DHCP) server.

App1

IP Address: 192.168.100.3

Gateway: 192.168.100.1

DNS Server: 192.168.100.2

Hostname: app1.college.com

> This is an internal web and application server.

Client1

IP Address: Dynamic

DNS Server: 192.168.100.2

Hostname: client1.college.com

> Client1 is a workstation that has the ability to switch between the Intranet and Internet subnet.

4.2. Hardware and Software Requirements

Minimum requirements:

- Four functioning computers or VMs for configuration with OpenBSD installed.

4.3. Using “Sudo”

When running many commands throughout these TLGs you may be required to use ‘sudo’ depending on the level of privilege the current user has. The ‘sudo’ command allows the command following it to be run at superuser privileges. It is important to note that the user must be in the ‘sudoers’ file in order to successfully use this command. If you make the account an administrator while installing, it will be added to the ‘sudoers’ file automatically. Once a command is preceded by ‘sudo’ the user will be prompted for the password for the account and then the command can be run at superuser privilege. For details on how to give a user ‘sudo’ privilege see the Appendix (How to Give a User to Sudo Privileges).

5. Configuring the Environment

5.1. Step One - Configuring the Firewall

The below instructions detail the setup of Firewall. Here you will give your firewall a hostname, configure it for IP forwarding and set the IP addresses for it two interfaces

1. Configure hostname on Firewall
    a. Elevate to root
        i. `su -`
    b. Get nano text installer
        i. `pkg_add nano`

        > Note: To exit out of nano hit ctrl-x, press y to confirm the changes and press enter.

    c. Change the hostname
        i. `nano /etc/myname`
        ii. Enter the following: `fw.college.com`
2. Configure networking on Firewall
    a. Enable IP Forwarding
        i. `echo 'net.inet.ip.forwarding=1' >> /etc/sysctl.conf`
    b. Enable DHCP on em0
        i. `echo 'dhcp' > /etc/hostname.em0`

c. Give internal interface (em1) a static IP

   i. `echo 'inet 192.168.100.1 255.255.255.0 192.168.100.255' > /etc/hostname.em1`

d. Restart Networking

   i. `sh /etc/netstart`

5.2. Configuring NAT

Here we will configure Network Address Translation so that our internal computers can get to the Internet

1. Edit firewall configuration file

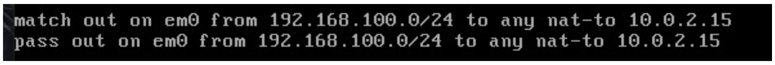
   a. `nano /etc/pf.conf`

   > Note In my case em0 was my external interface yours might be different Same thing for the IP 10.0.2.15, this is my external IP address yours might be different

   b. Add the following line at the bottom of pf.conf:

   ```
   match out on em0 from 192.168.100.0/24 to any nat-to 10.0.2.15
   pass out on em0 from 192.168.100.0/24 to any nat-to 10.0.2.15
   ```

   c. It should look like this:

   ```
   match out on em0 from 192.168.100.0/24 to any nat-to 10.0.2.15
   pass out on em0 from 192.168.100.0/24 to any nat-to 10.0.2.15
   ```

   d. Apply changes to firewall

      i. `pfctl -f /etc/pf.conf`

2. Enable IP Forwarding

   a. Enable IP forwarding so our internal addresses can reach the internet

      i. `echo 'net.inet.ip.forwarding=1' >> /etc/sysctl.conf`

   b. Add this line if you intend on using IPv6

      i. `echo 'net.inet6.ip6.forwarding=1' >> /etc/sysctl.conf`

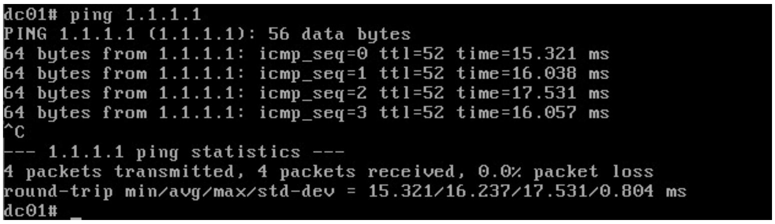
   c. Restart your firewall

      i. `reboot`

3. To test this on one of your internal clients see if you can reach an external server

   a. `ping 1.1.1.1`

      i. You should see something like this



```
dc01# ping 1.1.1.1
PING 1.1.1.1 (1.1.1.1): 56 data bytes
64 bytes from 1.1.1.1: icmp_seq=0 ttl=52 time=15.321 ms
64 bytes from 1.1.1.1: icmp_seq=1 ttl=52 time=16.038 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=52 time=17.531 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=52 time=16.057 ms
^C
--- 1.1.1.1 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 15.321/16.237/17.531/0.804 ms
dc01# _
```

5.3. Snapshot the Configuration

Preserving a functional state of the machine is important in case changes are made accidently. This can be done using VMWare’s snapshot functionality or other software packages for hardware labs.

5.4. Step Two - Configuring the DC

The below instructions detail the setup of your domain controller (DC). Here you will configure DC’s hostname and IP address

1. Configure hostname on DC

   a. Elevate to root

      i. `su -`

> Note: To save and exit in vi mode type :wq

b. Change the hostname

   i. `vi /etc/myname`

   ii. Enter the following: `dc01.college.com`

2. Configure networking on DC

   a. Set gateway

      i. `nano /etc/mygate`

      ii. Enter the following: "192.168.100.1"

   b. Give your Client an IP address

      i. `echo 'inet 192.168.100.2 255.255.255.0 192.168.100.255' > /etc/hostname.em0`

   c. Restart Networking

      i. `sh /etc/netstart`

5.5. Configuring DHCP

Here you will configure dc as a DHCP server for your network to automatically assign addresses

1. Configure DHCP on DC

   a. Enable DHCP to run at boot

      i. `rcctl enable dhcpd`

   b. Set DHCP flags so that DHCP traffic can be sent out

      i. `rcctl set dhcpd flags em0`

   c. Enter the dhcp file end enter the following information for a DHCP client's network settings

      i. `vi /etc/dhcpd.conf`

```
subnet 192.168.100.0 netmask 255.255.255.0 {
 option routers 192.168.100.1;
```

```
     option domain-name-servers 192.168.100.2;
     range 192.168.100.5 192.168.100.254;
    }
```

5.6. Configuring DNS

Here Domain Name Services (DNS) will be configured on DC so that computers inside your network can navigate to outside websites via their hostnames

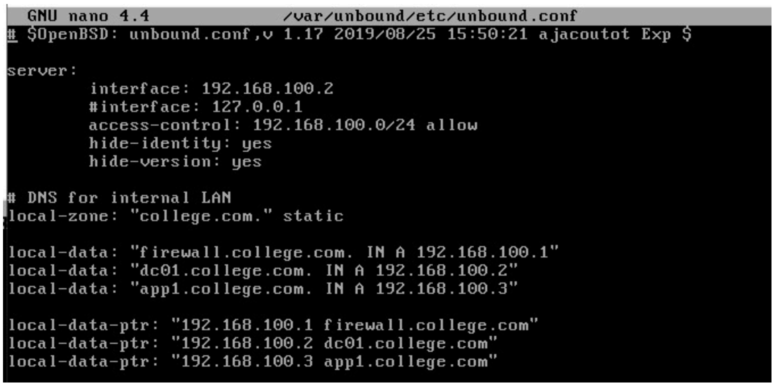
1. Enable unbound to run at boot

   a. `rcctl enable unbound`

2. Edit the unbound configuration file so that it looks like the following

   a. `vi /var/unbound/etc/unbound.conf`

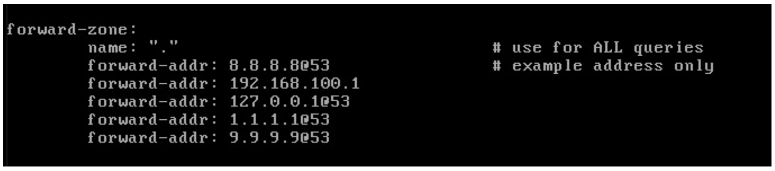
```
server:
        interface: 192.168.100.2
        #interface: 127.0.0.1
        access-control: 192.168.100.0/24 allow
        hide-identity: yes
        hide-version: yes

# DNS for Internal LAN
local-zone: "college.com." static

local-data: "firewall.college.com. IN A 192.168.100.1"
local-data: "dc01.college.com. IN A 192.168.100.2"
local-data: "app1.college.com. IN A 192.168.100.3"

local-data-ptr: "192.168.100.1 firewall.college.com"
local-data-ptr: "192.168.100.2 dc01.college.com"
local-data-ptr: "192.168.100.3 app1.college.com"
```



```
  GNU nano 4.4              /var/unbound/etc/unbound.conf
# $OpenBSD: unbound.conf,v 1.17 2019/08/25 15:50:21 ajacoutot Exp $

server:
        interface: 192.168.100.2
        #interface: 127.0.0.1
        access-control: 192.168.100.0/24 allow
        hide-identity: yes
        hide-version: yes

# DNS for internal LAN
local-zone: "college.com." static

local-data: "firewall.college.com. IN A 192.168.100.1"
local-data: "dc01.college.com. IN A 192.168.100.2"
local-data: "app1.college.com. IN A 192.168.100.3"

local-data-ptr: "192.168.100.1 firewall.college.com"
local-data-ptr: "192.168.100.2 dc01.college.com"
local-data-ptr: "192.168.100.3 app1.college.com"
```

b. After the records add the forward zones in the same file

```
forward-zone:
    name: "."                                #use for ALL queries
    forward-addr: 8.8.8.8@53                 #example address only
    forward-addr: 192.168.100.1
    forward-addr: 127.0.0.1@53
    forward-addr: 1.1.1.1@53
    forward-addr: 9.9.9.9@53
```



```
forward-zone:
        name: "."                               # use for ALL queries
        forward-addr: 8.8.8.8@53                # example address only
        forward-addr: 192.168.100.1
        forward-addr: 127.0.0.1@53
        forward-addr: 1.1.1.1@53
        forward-addr: 9.9.9.9@53
```

3. Restart the unbound service

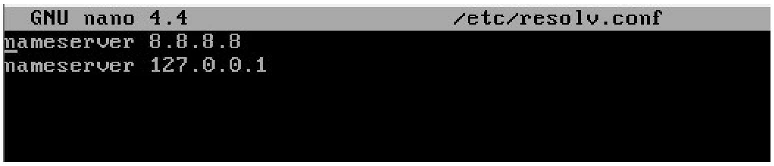
   a. `rcctl restart unbound`

4. Edit "/etc/resolv.conf" on dc01

   a. Edit "/etc/resolv.conf" on dc01 so it looks like this

      i. `vi /etc/resolv.conf`

```
nameserver 8.8.8.8
nameserver 127.0.0.1
```



```
  GNU nano 4.4                              /etc/resolv.conf
nameserver 8.8.8.8
nameserver 127.0.0.1
```

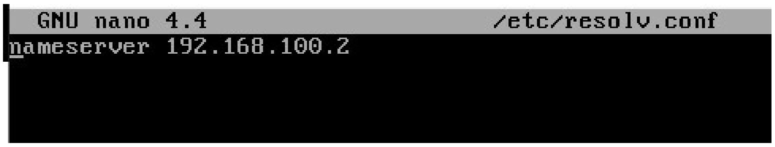
ii. You should be able to `ping google.com`

5. Edit "/etc/resolv.conf" on any machine using dc01 for DNS

   a. Edit "/etc/resolv.conf" so it looks like this

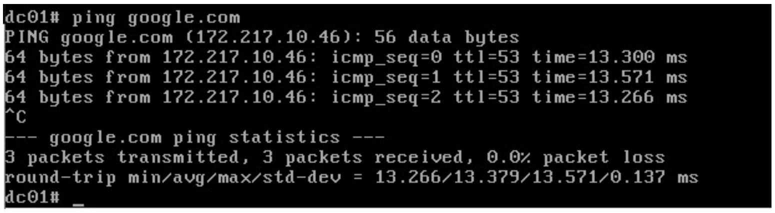
      i. `vi /etc/resolv.conf`

```
nameserver 192.168.100.2
```



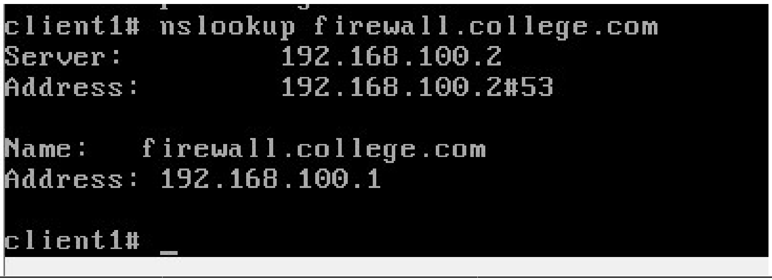
```
  GNU nano 4.4                              /etc/resolv.conf
nameserver 192.168.100.2
```

ii. You should be able to `ping google.com`



```
dc01# ping google.com
PING google.com (172.217.10.46): 56 data bytes
64 bytes from 172.217.10.46: icmp_seq=0 ttl=53 time=13.300 ms
64 bytes from 172.217.10.46: icmp_seq=1 ttl=53 time=13.571 ms
64 bytes from 172.217.10.46: icmp_seq=2 ttl=53 time=13.266 ms
^C
--- google.com ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 13.266/13.379/13.571/0.137 ms
dc01# _
```

iii. And also have successful name resolutions in your LAN



```
client1# nslookup firewall.college.com
Server:         192.168.100.2
Address:        192.168.100.2#53

Name:   firewall.college.com
Address: 192.168.100.1

client1# _
```

5.7. Snapshot the Configuration

Preserving a functional state of the machine is important in case changes are made accidently. This can be done using VMWare's snapshot functionality or other software packages for hardware labs.

5.8. Step Three - Configuring the Client

The below instructions detail the setup of Client. Here you will configure Client's hostname and IP address

1. Configure hostname on Client
   a. Elevate to root
      i. `su -`
   b. Change the hostname
      i. `vi /etc/myname`
      ii. Enter the following: `client1.college.com`
2. Configure networking on Client
   a. Set gateway
      i. `nano /etc/mygate`
      ii. Enter the following: "192.168.100.1"
   b. Give your Client an IP address
      i. `echo 'inet 192.168.100.4 255.255.255.0 192.168.100.255' > /etc/hostname.em0`
   c. Restart Networking
      i. `sh /etc/netstart`

5.9. Configuring Client to use DHCP

Here you will configure DHCP on your clients so that they get their IP Addresses automatically assigned

1. Configure DHCP on Client

   a. Enable DHCP to run at boot

      i. `rcctl enable dhcpd`

   b. Set DHCP flags so that DHCP traffic can be sent out

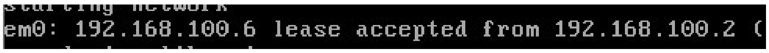
      i. `rcctl set dhcpd flags em0`

   c. Write to the networking file dhcp so client receives an address from dc01

      i. `echo “dhcp” > /etc/hostname.em0`

      ii. `sh /etc/restart`

   d. You should get a message saying you received an ip address:



```
em0: 192.168.100.6 lease accepted from 192.168.100.2 (
```

   e. If you don’t then rebooting your machine might get you an address

      i. reboot

5.10. Snapshot the Configuration

Preserving a functional state of the machine is important in case changes are made accidently. This can be done using VMWare’s snapshot functionality or other software packages for hardware labs.

5.11. Step Four - Configure App1

The below instructions detail the setup of your application server. Here you will configure app1’s hostname and IP address

1. Configure hostname on App1

   a. Elevate to root

      i. `su -`

   b. Change the hostname

      i. `vi /etc/myname`

ii. Enter the following: `app1.college.com`

2. Configure networking on App1

   a. Set gateway

      i. `nano /etc/mygate`

      ii. Enter the following: "192.168.100.1"

   b. Give your Client an IP address

      i. `echo 'inet 192.168.100.3 255.255.255.0 192.168.100.255' > /etc/hostname.em0`

   c. Restart Networking

      i. `sh /etc/netstart`

5.12. Configuring HTTPD

Here app1 will be configured so that it runs an apache (httpd) web server

1. Enable httpd to start at boot

   a. `rcctl enable httpd`

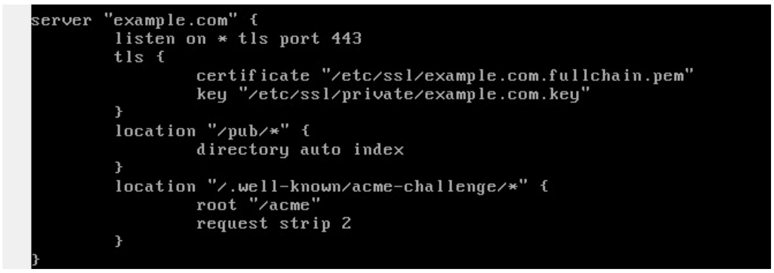
2. Copy example httpd.conf file to "/etc" directory

   a. `cp /etc/examples/httpd.conf /etc`

3. Delete SSL configurations from "httpd.conf"

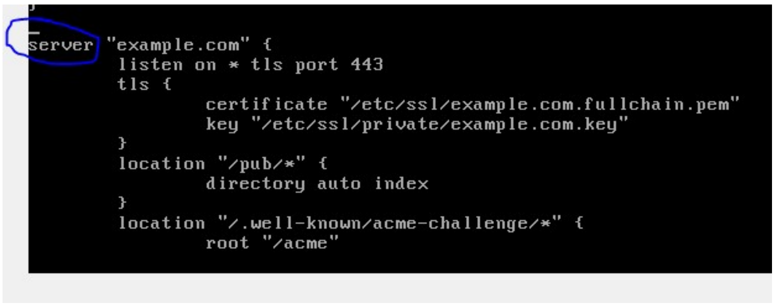
   a. `vi /etc/httpd.conf`

   b. Navigate to where the configuration for ssl starts, might look something like this:



```
server "example.com" {
        listen on * tls port 443
        tls {
                certificate "/etc/ssl/example.com.fullchain.pem"
                key "/etc/ssl/private/example.com.key"
        }
        location "/pub/*" {
                directory auto index
        }
        location "/.well-known/acme-challenge/*" {
                root "/acme"
                request strip 2
        }
}
```

c. In vi make sure your curso is above the tls settings like this



```
}
_
server "example.com" {
        listen on * tls port 443
        tls {
                certificate "/etc/ssl/example.com.fullchain.pem"
                key "/etc/ssl/private/example.com.key"
        }
        location "/pub/*" {
                directory auto index
        }
        location "/.well-known/acme-challenge/*" {
                root "/acme"
```

d. To get rid of all lines below where your cursor is do

i. `:+,$d`

4. Check for syntax errors in file

a. `httpd -n`

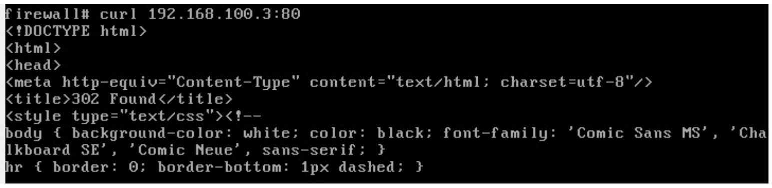
b. If there are no errors then start the service Start the service

c. `rcctl start httpd` Check to see if you get a response from your firewall

d. `pkg_add`

e. `curl “192.168.100.3”`

f. You should get back something like this:



```
firewall# curl 192.168.100.3:80
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>302 Found</title>
<style type="text/css"><!--
body { background-color: white; color: black; font-family: 'Comic Sans MS', 'Cha
lkboard SE', 'Comic Neue', sans-serif; }
hr { border: 0; border-bottom: 1px dashed; }
```

5. Edit app1’s firewall to allow port 80 traffic and restart the firewall

a. If you want this webserver to be viewed by the outside word

b. `nano /etc/examples/pf.conf`

c. Add pass in on egress proto tcp from any to any port 80

d. `pfctl -f /etc/examples/pf.conf`

6. Edit firewall
    a. `nano /etc/examples/pf.conf`
    b. pass in on egress proto tcp from any to any port 80
    c. `pfctl -f /etc/examples/pf.conf`
    d. `nano /etc/rc.conf.local`
7. Got back to App1 and allow for http flags
    a. `nano /etc/rc.conf.local`
    b. `httpd_flags=`
    c. `pkg_scripts=”httpd”`
8. Test connection on firewall
    a. `curl “192.168.100.3”`

You should see the following:

+ image::curlipapp1.jpg[firewall test connection to app1, align="center"]

5.13. Configuring SMB

Here app1 will be configured to host a samba server that can be used to share files across your network from multiple different computers

1. Download SMB
    a. `pkg_add samba`
2. Enable SMB to start at boot
    a. `rcctl enable smbd nmbd`
3. Edit the Configuration file if you want
    a. `vi /etc/samba/smb.conf` (you can use previous config)
4. Create a directory for SMB service
    a. `mkdir -p /home/samba`
5. Change permission to nobody so that multiple people can get to the shares

   a. `chown nobody:nobody /home/samba`

6. Start the service

   a. `rcctl start smbd nmbd`

5.14. Snapshot the Configuration

Preserving a functional state of the machine is important in case changes are made accidently. This can be done using VMWare's snapshot functionality or other software packages for hardware labs.

6. Troubleshooting

To test network connectivity run `ping 8.8.8.8`. If it fails, run `ping 192.168.100.1` to see if you can ping the router. If you can't ping the router make sure your computer is on the correct network. If you can but you can't reach 8.8.8.8, make sure the computer has an ip address and that NAT is working properly.

If you look up any error codes or messages you are getting, most of the time someone has ran into the same problem and you can find the solution.

If you run `ping google.com` and it fails but you can `ping 8.8.8.8` then the problem is with your DNS configuration.

7. Contributors

- Brandon Ledyard
- Michael Pare