WifiDocs/WirelessAccessPoint

目录

Introduction

In computer networking, a wireless access point (WAP) is a device that connects wireless communication devices together to form a wireless network. The WAP can connect to a wired or wireless network, relaying data between devices. There are several scenarios WAPs are used in:

1. Setup a home network.
2. Link together WAPs to form a larger network that allows clients to freely roam between without being disconnected.
3. Wireless client devices manage themselves in an ad-hoc network.

Setting up a wireless-to-wireless router

Overview

This is tested and geared to a home, or home office setup. It applies well to the following scenarios:

1. You don't want to have run ethernet cords all over the place.
2. your ISP restricts you to one IP address via a modem, a handful of IP addresses, or your ISP provides a wireless only router.
3. You want to avoid buying a load of additional, potentially expensive equipment.

The diagram for this:

```
-> Android smartphone
                                                                  /
[ISP Device] <----------> [WiFi Adapter][Computer][WiFi Adapter]--> Ubuntu/Mac/Windows Clients
                    (wlan1)                        (wlan0)    \
                                                           -> iPhone
```

Prerequisites

1. Dedicated computer to be your router
2. Kubuntu Desktop 14.04.1 x64
3. wlan0 that supports being an access point. One may review what a card supports via a terminal:

   ```
   iw list
   ```

   ```
   * AP
   ```

   What was used in this test: Qualcomm Atheros AR9287 Wireless Network Adapter (PCI-Express) [168c:002e] (rev 01)
4. wlan1 that supports managed mode. One may review what a card supports via a terminal:

   ```
   iw list
   ```

   ```
   * Managed
   ```

   What was used in this test: Belkin N300 Wireless USB Adapter F9L1002v1 050d:845a
5. Wireless client device(s)
6. ISP provided wireless only router

Install Kubuntu

When installing, setup the initial user to automatically log in.

Setup hostapd, port forwarding, haveged, and dnsmasq

Install and configure hostapd to automatically start on boot up using wlan0, and communicate on 802.11n, 2.4GHz (150 Mbps), with WPA2-Personal:

```
sudo apt-get -y install hostapd
sudo nano /etc/default/hostapd
```

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

```
sudo nano /etc/hostapd/hostapd.conf
```

```
auth_algs=1
beacon_int=50
channel=3
country_code=US
disassoc_low_ack=1
driver=nl80211
hw_mode=g
ht_capab=[HT40+][HT40-][SHORT-GI-40][RX-STBC1]
ieee80211d=1
ieee80211n=1
interface=wlan0
require_ht=0
rsn_pairwise=CCMP
ssid=YOURSSID
wmm_enabled=1
wpa=2
wpa_key_mgmt=WPA-PSK
wpa_passphrase=YOURPASSPHASE
```

Replace the country_code, ssid, and wpa_passphrase with the applicable information. For some reason, a Windows 8.1 client (Qualcomm Atheros AR956x Wireless Network Adapter) would not connect when require_ht=1. Event Viewer noted "A response was not received from the router or access point." However, this same client booted into Ubuntu worked fine, as well as Mac OS X Yosemite, iPhone, and a android smartphone.

Next, have port forwarding enabled automatically on boot up:

```
sudo nano /etc/sysctl.conf
```

```
net.ipv4.ip_forward=1
net.ipv6.conf.all.forwarding=1
```

Next, have NAT traffic initiated on wlan0 forwarded to wlan1 enabled automatically on boot up:

```
sudo nano /etc/rc.local
```

```
iptables -t nat -A POSTROUTING -o wlan1 -j MASQUERADE
iptables -A FORWARD -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i wlan0 -o wlan1 -j ACCEPT
exit 0
```

Next, increase the default amount of entropy on your router and linux clients via haveged. When a linux operating system runs low on entropy (< 1000), it causes severe encrypted communcation latency (among other problems):

```
sudo apt-get -y install haveged
```

Next, install and configure dnsmasq for DHCP mapping IP addresses to MAC addresses:

```
sudo apt-get -y install dnsmasq
sudo nano /etc/dnsmasq.conf
```

```
interface=wlan0
except-interface=wlan1
dhcp-range=10.0.0.2,10.0.0.14,12h
```

Note: One may want to setup DHCP reservations to bind a MAC address to a specific IP address (in the case of servers or clients you want to remote into by IP address) by adding to this file:

```
dhcp-host:XX:XX:XX:XX:XX,10.0.Y.Z
```

Where the "X"s are replaced with the MAC address, and both Y and Z with the relevant IP address numbers.

Configuring networking

One has a number of options for setting up networking. One may use the GUI driven NetworkManager for the managed interface and /etc/network/interfaces for access point. This has the benefit of being more user friendly. Another option is setting up both interfaces via /etc/network/interfaces. This has the benefit of reducing computer resources overhead.

Method 1: NetworkManager and /etc/network/interfaces

Connect wlan1 to the upstream router via NetworkManager (have "All users may connect to this network" checked). As well, it would be best to set the IP address statically so you may remote into this interface by IP address easily. It is recommended to setup x11vnc tunneled through SSH to easily administer the computer remotely.

Next, setup the wlan0 interface to have a static IP address, set to wireless master mode, and come up automatically on boot up via a terminal:

```
sudo nano /etc/network/interfaces
```

```
auto lo wlan0
iface lo inet loopback
#access point
iface wlan0 inet static
address 10.0.0.1
netmask 255.255.255.240
gateway 10.0.0.1
dns-nameservers 192.168.1.1
wireless-mode Master
```

Next, set NetworkManager to ignore the wlan0 interface:

```
sudo nano /etc/NetworkManager/NetworkManager.conf
```

```
[keyfile]
unmanaged-devices=mac:XX:XX:XX:XX:XX:XX
```

Replace the "X"s with the MAC address for wlan0.

Considerations

1. One may want to utilize a computer with two internal PCIe WiFi adapters, versus a USB WiFi dongle. This would maximize the speed of internal adapter-to-adapter communication, and avoid any potential USB overheat issues.
2. One would want to avoid using devices with less developed WiFi drivers (ex. Belkin N300 Wireless USB Adapter F9L1002v1 050d:845a which uses staging driver rtl8712u) over more developed ones (ex. ath9k). In the case of the above device with rtl8712u, this is due to how lowering the transmission power of the adapter via the following terminal command is not yet developed:

   ```
   iwconfig wlan0 txpower 14
   ```
3. While using a GUI eases administration, it incurs additional resource overhead to run it.

Setting up a bridged wireless-to-ethernet router

Overview

This is an advanced, command line only setup for creating a bridged wireless router with Ubuntu Server. Here is the network diagram for this setup:

Note: This inkscape diagram `network.svg` is available for download from: More Actions > Attachments.

Prerequisites

1. One gigabit ethernet switch.
2. One USB drive with Ubuntu Server ISO setup for install (Verified with 7.10).
3. A dedicated computer that will function as your router, with the following hardware:
    1. One wireless NIC (ath0), that supports master mode. For more on this, please see here.
    2. One ethernet NICs (eth0).
4. Optional: A client computer to remotely configure your router. If your client is using Windows:
    1. SSH
    2. putty - remote console
    3. WinSCP - remote file transfer

Install Ubuntu Server

Install Ubuntu Server Edition. One may choose the following options to install:

```
[*] DNS server
[*] OpenSSH server
```

Update and upgrade Ubuntu:

```
sudo apt-get update
sudo apt-get dist-upgrade
```

To install DNS server and SSH Server after an ubuntu installation use the command:

```
tasksel
```

Setup SSH

Assuming that your ubuntu box is connected to your upstream router, you will need to find your IP address of your ubuntu box so you can connect with **putty**:

```
ip addr
```

On your windows machine install putty. Type in your ubuntu IP address into putty then connect.You can now cut and paste the following the commands. If you want to transfer files use WinSCP.

Setup the network

eth0 is the WAN interface (gateway) eth1 is the LAN interface ath0 is the wireless card br0 is the bridged connection of ath0 and eth1

Setup bridging:

```
sudo apt-get install bridge-utils
```

Then edit the network config:

```
sudo nano /etc/network/interfaces
```

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

#Gateway -
auto eth0
iface eth0 inet dhcp
pre-up iptables-restore < /etc/iptables.rules
post-down iptables-save > /etc/iptables.rules

#Wireless Setup
auto ath0
iface ath0 inet manual
wireless-mode master
wireless-essid pivotpoint

#Bridge interface
auto br0
iface br0 inet static
    address 10.1.1.1
    network 10.1.1.0
    netmask 255.255.255.0
    broadcast 10.1.1.255
    bridge-ports eth1 ath0
```

Setup firewall rules

Run these commands:

```
sudo iptables -t nat -A POSTROUTING -s 10.1.1.0/24 -o eth0 -j MASQUERADE
sudo iptables -A FORWARD -s 10.1.1.0/24 -o eth0 -j ACCEPT
sudo iptables -A FORWARD -d 10.1.1.0/24 -m conntrack --ctstate ESTABLISHED,RELATED -i eth0 -j ACCEPT
```

For logging add:

```
sudo iptables -A INPUT -m conntrack --ctstate NEW -p tcp --dport 80 -j LOG --log-prefix "NEW_HTTP_CONN: "
```

The above log will also appear in /var/log/messages, /var/log/syslog, and /var/log/kern.log.

Save to /etc/iptables.rules via:

```
sudo sh -c "iptables-save > /etc/iptables.rules"
```

NOTE: This is a basic setup that only routes NAT packets. Please investigate further securing your firewall.

Enable packet forwarding:

```
sudo nano /etc/sysctl.conf
```

Add the following line:

```
net.ipv4.ip_forward = 1
```

Diagnostic tools

Immediately allow the forwarding of packets. The configuration is not preserved on reboot but sets a flag in the kernel itself:

```
echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
```

Test the current setting of the kernel:

```
cat /proc/sys/net/ipv4/ip_forward
```

Note: The /proc directory is not on your hard drive but is present in the running kernel.

Setup the DHCP server

This will create a 10 machine DHCP server via a terminal:

```
sudo apt-get install dhcp3-server
```

Configure the server:

```
sudo nano /etc/dhcp3/dhcpd.conf
```

```
# Subnet for DHCP Clients
subnet 10.1.1.0 netmask 255.255.255.0 {
        option domain-name-servers 10.1.1.1;
        max-lease-time 7200;
        default-lease-time 600;
        range 10.1.1.50 10.1.1.60;
        option subnet-mask 255.255.255.0;
        option broadcast-address 10.1.1.255;
        option routers 10.1.1.1;
          }
```

NOTE: If a DNS server (bind9) hasn't been installed change "option domain-name-servers 10.1.1.1" to the IP address of the DNS server provided by your ISP. Optionally, one may use a public DNS server such as Google DNS: 8.8.8.8 or 8.8.4.4.

You also need to edit the file /etc/default/dhcp in order to specify the interfaces dhcpd should listen to. By default it listens to eth0. We need to only have it listen to our local NIC (br0):

```
sudo nano /etc/default/dhcp3-server
```

Then add:

```
INTERFACES="br0"
```

Restart.

Optional: Monitoring with darkstat

Stats with a http server:

```
sudo apt-get install darkstat
```

Edit the configuration file:

```
sudo nano /etc/darkstat/init.cfg
```

```
 # Turn this to yes when you have configured the options below.
START_DARKSTAT=yes

# Don't forget to read the man page.

# You must set this option, else darkstat may not listen to
# the interface you want
INTERFACE="-i eth1"

PORT="-p 8888"
#BINDIP="-b 127.0.0.1"
#LOCAL="-l 10.1.1.0/24"
#FIP="-f 127.0.0.1"
#DNS="-n"
#SPY="--spy eth1"
```

To see this point a browser to: http://10.1.1.1:8888

Optional: Monitoring with saidar

Saidar shows server usage:

```
sudo apt-get install saidar
```

then:

```
saidar
```

Optional: Disabling IPv6

Depending on your hardware, speed improvements may be found by disabling IPv6. For more on this, please see WebBrowsingSlowIPv6IPv4.

Backup

Reference = http://doc.gwos.org/index.php/Backup_restore_system

```
sudo su -
cd /
tar cvpjf backup.tar.bz2 --exclude=/proc --exclude=/media --exclude=/mnt --exclude=/dev --
exclude=/lost+found --exclude=/backup.tar.bz2 --exclude=/tmp --exclude=/sys /
```

You will then have a tar ball that is your server.

External links

1. Man page for random discussing entropy http://manpages.ubuntu.com/manpages/trusty/man4/random.4.html
2. hostapd.conf http://w1.fi/cgit/hostap/plain/hostapd/hostapd.conf
3. How to setup an Atheros-based Access Point with WPA-PSK on Ubuntu 8.04 server

CategoryHardware CategoryNetworking CategoryWireless

The material on this wiki is available under a free license, see Copyright / License for details
You can contribute to this wiki, see Wiki Guide for details