Wi-Fi to eth (bridge) routing

This Howto describes, how to interconnect wireless and wired network interfaces on the same Linux computer, to enable unmodified TCP/IP packets to pass from one interface to the other. In other places this is mentioned as network bridge or Wi-Fi line extender or Wi-Fi Internet share.

The reason for this HOWTO: the word bridge is misleading

For a network bridge we assume a device that transfers unmodified network packets from one network connection to the other. One can create a bridge device (virtual) and add members to it. This works only for bridge members of type wired - *eth* network cards. A network Bridge “connects” members on level 3 of the OSI model. That means communication on TCP/IP level.
When you want to add a Wi-Fi device to the bridge, you hit a barrier: Wi-Fi devices communicate on level 2 of the OSI model. You can find many manuals on the Internet that document how to circumvent this (in the form of putting the Wi-Fi card in 4addr mode). This simply DOES NOT WORK!
The Wi-Fi network card (member of the bridge) authenticates and connects to the Wireless Access Point (AP), but TCP/IP packets do not travel over the connection.
So searching for “wifi eth bridge” does not return any useful solution. The culprit is the word “bridge”.

General solution

A working solution is “Proxy ARP Routing”.
You simply enable IP forwarding and then for every device connected to the wired (eth) side of a “bridge” you have to add a routing line to the routing table. This can be automated by a program like parprouted - the Proxy ARP routing daemon.

Reading of the parprouted man page is highly recommended to be aware of its limitations

Solution for Slackware, step-by-step

Tested and working on Slackware64-14.2, kernel-4.11.6, CPU i5-7200

This solution is for static IP addresses. See below link of original source for a scenario that uses DHCP.

Assumptions:
We want to interconnect one Wi-Fi and one wired (eth) network card - the network devices `wlan0` and `eth0`.
Prepare Slackware box so that you are able to communicate over the Wi-Fi adapter (using NetworkManager, rc.inet1 or other means…), making sure that the wired (eth) adapter is not being used.
I had set up WPA2 AES verification with NetworkManager to get a usable `wpa_supplicant.conf` configuration file which I used later with rc.inet1.

Disable all on-boot network configurations (i.e. make sure that `rc.networkmanager` or other files for network setup are not executable) and set `rc.inet1` executable.

1. IP forwarding must be enabled in the kernel (since the 2.1 release the Linux kernel does not require an explicit compilation option for this)
2. download & compile & install http://www.hazard.maks.net/parprouted/parprouted-0.7.tar.gz ⚠
3. edit `/etc/rc.d/rc.inet1.conf` so to enable `wlan0` and `eth0`. Assign them static IP addresses and set `wlan0` to the lowest index and connect to the AP
   Below are example lines from `/etc/rc.d/rc.inet1.conf` - the only ones without comment sign “#” at the beginning and for WPA2 Wi-Fi authentication

   ```
   IFNAME[1]="eth0"
   IPADDR[1]="10.200.200.223"
   NETMASK[1]="255.255.255.0"

   GATEWAY="10.200.200.1"
   DEBUG_ETH_UP="no"

   IFNAME[0]="wlan0"
   IPADDR[0]="10.200.200.222"
   NETMASK[0]="255.255.255.0"
   USE_DHCP[0]=""
   DHCP_HOSTNAME[0]=""

   WLAN_MODE[0]=Managed
   WLAN_ESSID[0]="R7500"

   WLAN_WPA[0]="wpa_supplicant"
   WLAN_WPADRIVER[0]="wext"
   ```

4. set `/etc/rc.d/rc.ip_forward` executable:

   ```
   # chmod +x /etc/rc.d/rc.ip_forward
   ```

5. add a line

   ```
   /usr/local/sbin/parprouted wlan0 eth0
   ```

   to `/etc/rc.d/rc.local` and make sure that this file is executable

That's all. Reboot and you have a working Wi-Fi - eth bridge, also called Wi-Fi extender or Wi-Fi Internet share.

⚠ Note on parprouted compilation:
The `parprouted` man page section “Requirements” says: *“parprouted requires the “ip” program from iproute2 tools to be installed in /sbin. If it is installed in another location, please replace ”/sbin/ip“ occurrences in the source with the correct path”*.
Slackware installs the `ip` program as `/sbin/ip` so you should be OK.

DHCP enabled variant

Look below for a solution in a source link.

Sources

* Written by Zdenko Dolar, August 2017

* Original source:https://wiki.debian.org/BridgeNetworkConnectionsProxyArp

howtos, network, WiFi, bridge, author slacker