Debian: Configuration réseau aide-memoire

Table of Contents

La configuration sur cette page est basée sur Debian, mais devrait fonctionner avec d'autres distributions à base de debian (Ubuntu)

ip-command a besoin du package iproute2

nécessite le noyau 4.14 ou supérieur (pilote DSA pour la séparation des ports)

dans le noyau 4.14 eth0 est la connexion entre le CPU et le circuit de commutation (mt7530), sur lequel les ports wan et lan0-4 sont connectés. cette connexion doit d'abord être réglée sur “up”.

mettre ensuite le(s) port(s) up

```
ip link set eth0 up
ip link set eth1 up
```

ou via /etc/network/interfaces

```
auto eth0
iface eth0 inet manual
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down
```

```
auto eth1
iface eth1 inet manual
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down
```

le mappage des ports vers gmac est défini dans le fichier dts et peut être affiché avec “ip a”

Avec 4.14 >.52 sur mon repo, gmac #2 (eth1) est ajouté et wan est connecté à cela.

par défaut, chaque lan-port est séparé et a besoin d'une propre configuration IP dans différents sous-réseaux

la plupart des utilisateurs aiment utiliser tous les ports LAN dans 1 segment de réseau, ils peuvent donc être bridged together pour ne faire qu'une seule configuration IP pour “LAN”

MAC-Address

L'adresse MAC ne peut être définie que pour le GMAC (connexion entre le commutateur et la CPU). Dans Kernel 4.14, seul 1 GMAC est détecté (eth0). Il y a 2 GMAC dans Hardware.

Dans UDEV:

```
$ cat /etc/udev/rules.d/00-static-mac-address.rules
ACTION=="add", SUBSYSTEM=="net", KERNELS=="1b100000.ethernet",
RUN+="/sbin/ip link set dev %k address ae:fc:de:ad:be:ef"
```

dans le fichier interfaces

`/etc/network/interfaces`

```
iface lan0 inet static
  address 192.168.0.10
  netmask 255.255.255.0
  gateway 192.168.0.5
#  pre-up ip link set $IFACE up
  pre-up ip link set $IFACE address 02:01:02:03:04:08 up
```

en utilisant systemd

Créer un fichier `/etc/systemd/network/10-wan.link`

```
[Match]
OriginalName=wan

[Link]
```

```
MACAddress=XX:XX:XX:XX:XX:XX
```

http://forum.banana-pi.org/t/set-mac-address-on-boot/7224/7

dans device-tree

```
local-mac-address = [00 0a 35 00 00 01];
mac-address = [00 0a 35 00 00 01];
```

cela peut également être utilisé dans devicetree-overlays

via uboot

si devicetree (avec la propriété mac-address) est chargé séparément (fdt), un alias pour ethernet-node est défini et ethaddr-variable est défini dans uboot, il est utilisé dans Linux

Nom de l'interface

Ubuntu 18.4+ (et debiaan 10+) utilisant de nouveaux noms d'interface. Les appareils Wifi ne sont plus nommés comme wlanX mais plutôt comme wlpXsY

pour éviter cela, “net.ifnames=0” peut être ajouté à Kernel-Cmdline (uEnv.txt pour uboot) et/ou renommer via udev

`/etc/udev/rules.d/70-persistent-net.rules`

```
  SUBSYSTEM=="net", ACTION=="add", ATTR{address}=="f8:62:aa:50:15:c8",
NAME="wlan1"
```

trouver des attributs comme celui-ci

```
  udevadm info --attribute-walk /sys/class/net/<interface-name>
```

pour appliquer la ou les règles (mais ne renomme pas en arrière) :

```
  udevadm control --reload-rules && udevadm trigger
```

après redémarrage tout fonctionne

si le pilote est compilé en tant que module, il peut être rechargé (après avoir activé les règles udev)

```

```

```
modprobe -r mt76x2e
modprobe mt76x2e
```

IP

- Configurer une adresse de manière permanente dans `/etc/network/interfaces`:

```
#configurer d'abord le port en amont (NIC entre le CPU et le commutateur
MT7530)
auto eth0
iface eth0 inet manual
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down

auto eth1
iface eth1 inet manual
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down
#puis configurer les ports lan
auto lan0
iface lan0 inet static
  hwaddress ether 08:00:00:00:00:00 # if you want to set MAC manually
  address 192.168.0.10
  netmask 255.255.255.0
  gateway 192.168.0.5
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down
```

```
* Configurer une adresse IP de manière temporaire:
```

```
ifconfig lan0 192.168.0.10 netmask 255.255.255.0 broadcast 192.168.0.255
```

```
ip addr add 192.168.0.10/24 broadcast 192.168.0.255 dev lan0
```

il faut s'assurer qu'un seul port se trouve dans le sous-réseau spécifique.

```
  IP a
  #ou
  ip addr show lan0
```

DHCP

Configurer le client

dans /etc/network/interfaces:

```
auto lan3
allow-hotplug lan3
iface lan3 inet dhcp
```

Renouveler l'adresse IP via

```
sudo dhclient -v -r lan3
```

Configurer le serveur

dans /etc/dnsmasq.conf (activer la ligne en supprimant # au début de la ligne)

```
conf-dir=/etc/dnsmasq.d
```

/etc/dnsmasq.d/interfaces.conf

```
interface=wlan1
interface=ap0

# Le serveur DHCP n'est pas actif pour l'interface
no-dhcp-interface=eth0
no-dhcp-interface=eth1

#dhcp-authoritative (interface+range+leasetime, default-gateway-ip comme
option 3)
dhcp-range=ap0,192.168.10.100,192.168.10.150,255.255.255.0,48h
dhcp-option=ap0,3,192.168.10.1
dhcp-range=wlan1,192.168.11.100,192.168.11.150,255.255.255.0,48h
dhcp-option=wlan1,3,192.168.11.1
```

```
service dnsmasq start
```

IPv6

Pour désactiver IPV6 temporairement

```
echo 1 > /proc/sys/net/ipv6/conf/all/disable_ipv6
```

Pour le désactiver de façon permanente, créer le fichier /etc/sysctl.d/01-disable-ipv6.conf avec le contenu :

```
  net.ipv6.conf.all.disable_ipv6 = 1
```

puis vérifier:

```
ip addr show | grep inet6
```

NAT

pour activer la traduction d'adresses réseau (réseau avec des IP privées derrière une IP publique)

```
  ipt=/sbin/iptables
  if_wan=wan
  ${ipt} -t nat -A POSTROUTING -o ${if_wan} -j MASQUERADE
```

HW-Nat n'est actuellement disponible que dans LEDE (Kernel 4.9)

Routage

Pour activer le routage pour IPv4

```
  echo 1 > /proc/sys/net/ipv4/ip_forward
```

alternative:

```
nano /etc/sysctl.conf
#activer net.ipv4.ip_forward=1 et net.ipv6.conf.all.forwarding=1 en
supprimant # en début de ligne
sysctl -p /etc/sysctl.conf
```

manipuler la route par défaut :

```
ip route del default
ip route add default via 192.168.50.2
```

afficher la table de routage

```
ip route show
```

On aura besoin d'une résolution DNS (/etc/resolv.conf) pour traduire les domaines en

adresses IP

Ajouter des routes statiques à d'autres réseaux:

Les paquets sont envoyés à la passerelle par défaut, si le réseau n'est pas connu (directement connecté ou itinéraire disponible). Dans les réseaux domestiques normaux, il n'y a qu'un seul routeur et dans celui-ci, la passerelle par défaut est l'interface Internet et sur les PC clients, la passerelle par défaut est ce routeur.

des routes statiques sont nécessaires, si un réseau n'est pas directement connecté à un routeur et n'est pas accessible via sa passerelle par défaut:
- dans le routeur #1, une route statique doit être ajoutée pour le réseau 10.0.3.0/24 avec le prochain saut 10.0.2.2 (envoyer des paquets sur lan#2): `route ip add 10.0.3.0/24 via 10.0.2.2`
- dans le routeur #2, une route statique doit être ajoutée pour le réseau 10.0.1.0/24 avec le prochain saut 10.0.2.1 (envoyer des paquets sur lan # 1) `ip route add 10.0.1.0/24 via 10.0.2.1`

Exemple pour le net 192.168.50.x derrière le routeur avec ip 192.168.0.10

```
ip route addr 192.168.50.0/24 via 192.168.0.10
```

DNS

`/etc/resolv.conf` contient l'adresse IP du serveur de noms, par ex.

```
nameserver 192.168.0.10
```

Netbridge

si 2 ou plusieurs ports LAN doivent utiliser le même segment de réseau (configurez seulement 1 adresse IP pour “LAN”), vous pouvez relier les ports ensemble.

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

Pour configurer un pont permanent éditer `/etc/network/interfaces`:

```
auto lan1
iface lan1 inet manual
auto lan2
iface lan2 inet manual
```

```
auto br0
iface br0 inet static
    address 192.168.40.1
    netmask 255.255.255.0
    bridge_ports lan1 lan2
    bridge_fd 5
    bridge_stp no
```

Pour configurer un pont temporaire:

```
brctl addbr br0
brctl addif br0 lan1
brctl addif br0 lan2
ip addr add 192.168.40.1/24 dev br0
ip link set br0 up

brctl show br0
```

brctl étant obsolète, il faut utiliser ip/bridge à la place:

```
ip link add name br0 type bridge
ip link set dev br0 up
ip link set dev lan0 master br0
ip link set dev lan1 master br0

#supprimer l'interface du pont
ip link set dev lan0 nomaster

#supprimer le pont
ip link del br0
```

VLAN

Le support du vlan sur les ports dsa a besoin du patch suivant:

```
<codedoc toggle 0001-net-dsa-enable-vlan-without-bridge-on-dsa-user-
port.patch>
From 7caf1f5dce1b5a4b1001c37257b2b6fbd55e7c43 Mon Sep 17 00:00:00 2001
From: Landen Chao <landen.chao@mediatek.com>
Date: Tue, 31 Dec 2019 11:48:41 +0100
Subject: [PATCH] net: dsa: enable vlan without bridge on dsa user port

---
 drivers/net/dsa/mt7530.c | 14 ++++++++++++--
 1 file changed, 12 insertions(+), 2 deletions(-)

diff --git a/drivers/net/dsa/mt7530.c b/drivers/net/dsa/mt7530.c
index 1d8d36de4d20..7e285aa9bd7c 100644
```

```
--- a/drivers/net/dsa/mt7530.c
+++ b/drivers/net/dsa/mt7530.c
@@ -1165,8 +1165,13 @@ mt7530_port_vlan_add(struct dsa_switch *ds, int port,
    /* The port is kept as VLAN-unaware if bridge with vlan_filtering not
     * being set.
     */
-   if (!dsa_port_is_vlan_filtering(&ds->ports[port]))
+   if (!dsa_port_is_vlan_filtering(&ds->ports[port])){
+       /* Enable VLAN tagged in port-based vlan setting. */
+       if ((vlan->vid_begin != 0) && (vlan->vid_end != 0))
+           mt7530_rmw(priv, MT7530_PCR_P(port), EG_TAG(3),
+                  EG_TAG(2));
        return;
+   }

    mutex_lock(&priv->reg_mutex);

@@ -1196,8 +1201,13 @@ mt7530_port_vlan_del(struct dsa_switch *ds, int port,
    /* The port is kept as VLAN-unaware if bridge with vlan_filtering not
     * being set.
     */
-   if (!dsa_port_is_vlan_filtering(&ds->ports[port]))
+   if (!dsa_port_is_vlan_filtering(&ds->ports[port])) {
+       /* Disable VLAN tagged in port-based vlan setting. */
+       if ((vlan->vid_begin != 0) && (vlan->vid_end != 0))
+           mt7530_rmw(priv, MT7530_PCR_P(port), EG_TAG(3),
+                  EG_TAG(0));
        return 0;
+   }

    mutex_lock(&priv->reg_mutex);

--
2.17.1
</codedoc>
```

/etc/network/interfaces :

```
auto lan3.60
iface lan3.60 inet static
  address 192.168.60.10
  netmask 255.255.255.0
```

Pour activer temporairement le vlan :

```
ip addr add 192.168.40.11/24 dev lan1
ip link set lan1 up
ip link add link lan1 name vlan500 type vlan id 500
ip addr add 192.168.50.1/24 dev vlan500
```

```
ip link set vlan500 up
```

Avec 4.16, la prise en charge des vlan dans les ponts a été ajoutée.

vlan_filtering doit être activé avant que les ports dsa ne soient ajoutés au pont, sinon tout le trafic (non balisé également) est bloqué après ce paramètre.

```
#!/bin/bash
BRDEV=br-lan
LANDEV=lan2
BRIP=192.168.40.11/24
VLAN=500
VLANIP=192.168.50.11/24

#crée d'abord un pont avec vlan-suport et ajoutez dsa-port(s)
ip link set eth0 up #ifconfig eth0 up
brctl addbr $BRDEV
ip add add $BRIP dev $BRDEV
ip link set $BRDEV type bridge vlan_filtering 1
brctl addif $BRDEV $LANDEV
ip link set $BRDEV up
ip link set $LANDEV up

#ajout de vlan maintenant
bridge vlan add vid $VLAN dev $LANDEV master
bridge vlan add vid $VLAN dev $BRDEV self
ip link add link $BRDEV name $BRDEV.$VLAN type vlan id $VLAN
ip add add $VLANIP dev $BRDEV.$VLAN
ip link set $BRDEV.$VLAN up
bridge vlan show
```

On peut tester avec tcpdump:

```
  sudo tcpdump -ei lan1 arp ou icmp
```

-e affiche des informations sur la couche de liaison comme vlan

```
  sudo tcpdump -XXi lan1 arp ou icmp
```

affiche les paquets arp et icmp en tant que vidage hexadécimal sur l'interface

l'offset 0x0c devrait afficher 8100 suivi de la valeur hexadécimale du numéro de vlan (ici vlan 500 = 0x01f4)

```
12:16:26.491644 IP 192.168.50.11 > frank-G5: ICMP echo reply, id 4294, seq
5, length 64
0x0000:  3c18 a003 c3a4 c63a 3897 5920 8100 01f4  <......:8.Y.....
```

Monitoring

```
  sudo tcpdump -i eth0 port pas 22 > tcpdump.log
  sudo tcpdump -XXi lan1 arp # ou icmp
```

PPPoE

l'exemple crée une connexion pppoe sur vlan 140 comme celui d'un FAI

Pour le serveur:

```
sudo apt install pppoe

sudo ip link add link enx00e04c680683 name wan.140 type vlan id 140

/etc/ppp/pap-secrets
"bpi-r2"    *   "1234578"   *

/etc/ppp/pppoe-server-options
# options PPP pour le serveur PPPoE
# LIC : GPL
debug
#plugin /etc/ppp/plugins/rp-pppoe.so
require-pap
mtu 1492
mru 1492
ktune
proxyarp
lcp-echo-interval 10
lcp-echo-failure 2
nobsdcomp
noccp
novj
noipx
```

```
iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o enp3s0 -j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward

pppoe-server -I wan.140 -L 192.168.1.1 -R 192.168.1.100 -F &
```

Pour le client:

```
apt install pppoeconf
ip link add link wan name wan.140 type vlan id 140
ip link set wan.140 up
```

```
pppoeconf wan.140
```

`/etc/ppp/peers/dsl-provider` (doit être créé par pppoeconf)

```
# Fichier d'options par défaut minimaliste pour les connexions DSL/PPPoE

noipdefault
defaultroute
replacedefaultroute
hide-password
#lcp-echo-interval 30
#lcp-echo-failure 4
noauth
persist
#mtu 1492
#persist
#maxfail 0
#holdoff 20
plugin rp-pppoe.so wan.140
user "bpi-r2"
usepeerdns
```

```
pon dsl-provider
```

Il faut supprimer l'ancienne route vers le sous-réseau lan local qui est utilisé pour connecter le DNS (alors la route par défaut via ppp est utilisée).

Iptables

```
#delete previous rules
${ipt} -F
${ipt} -X
${ipt} -t nat -F
${ipt} -t nat -X
${ipt} -t mangle -F
${ipt} -t mangle -X

# Default-Rule for IPv4: drop all
${ipt} -P INPUT   DROP
${ipt} -P OUTPUT  DROP
${ipt} -P FORWARD DROP
```

```
# policy for TCP-Reset/UDP-Reject as alternative to "-j DROP"
${ipt} -N REJECTED
if [[ ! "${LOG}" = "" ]];
then
  echo "enable IPv4-Firewall-Logging (all)...";
  ${ipt} -A REJECTED -m limit --limit 10/min -j LOG --log-prefix
```

```
"NETFILTER4-REJECTED: " --log-level 4
fi
${ipt} -A REJECTED -p tcp -j REJECT --reject-with tcp-reset
${ipt} -A REJECTED -p udp -j REJECT --reject-with icmp-port-unreachable
${ipt} -A REJECTED -j DROP
```

```
# localhost
${ipt} -A INPUT  -i lo -j ACCEPT
${ipt} -A OUTPUT -o lo -j ACCEPT

${ipt} -A OUTPUT -j ACCEPT
${ipt} -A INPUT  -m state --state RELATED,ESTABLISHED -j ACCEPT # incoming
connetions which are requested
${ipt} -A INPUT  -p icmp -m limit --limit 5/s --icmp-type echo-request -j
ACCEPT # ICMP incoming, max 5/s
```

```
#Block Teredo-Stuff
#${ipt} -I FORWARD -p udp --dport 3544 -j REJECTED
#${ipt} -I FORWARD -p udp --sport 3544 -j REJECTED
#http://en.wikipedia.org/wiki/List_of_IP_protocol_numbers
${ipt} -A FORWARD -p 41 -j REJECTED #IPv6 Encapsulation
${ipt} -A FORWARD -p 43 -j REJECTED #Routing Header for IPv6
${ipt} -A FORWARD -p 44 -j REJECTED #Fragment Header for IPv6
${ipt} -A FORWARD -p 58 -j REJECTED #ICMP for IPv6
${ipt} -A FORWARD -p 59 -j REJECTED #No Next Header for IPv6
${ipt} -A FORWARD -p 60 -j REJECTED #Destination Options for IPv6
```

```
#ssh with rate-limit (replacing hosts.allow)
${ipt} -I INPUT -p tcp --dport 22 -i ${if_ext} -m state --state NEW -m
recent --set
${ipt} -I INPUT -p tcp --dport 22 -i ${if_ext} -m state --state NEW -m
recent --update --seconds 60 --hitcount 4 -j REJECTED #4 connections in 1
minute
${ipt} -A INPUT  -p tcp --dport  22 -j ACCEPT #SSH incoming
```

```
${ipt} -A FORWARD -i ${if_int} -o ${if_ext} -j ACCEPT #Forwarding Int->Ext
${ipt} -A FORWARD -i ${if_ext} -o ${if_int} -m state --state
ESTABLISHED,RELATED -j ACCEPT #Forwarding Ext->Int (only existing/requested
connections)

${ipt} -A INPUT -i ${if_int} -j ACCEPT #accept all request from internal
```

... (some other rules, e.g. port-forwardings)

```
# REJECT/RESET for everything else
${ipt} -A INPUT   -j REJECTED
${ipt} -A OUTPUT  -j REJECTED
${ipt} -A FORWARD -j REJECTED
```

additional options:

```
#Kernel-option for SYN-Cookies
echo 1 > /proc/sys/net/ipv4/tcp_syncookies  #enable syn cookies (prevent
against 'syn flood attack')

if [ -f /proc/sys/net/ipv4/conf/all/accept_redirects ]; then
    echo "   Kernel ignores all ICMP redirects"
    echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects
fi

if [ -f /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts ]; then
   echo "   Kernel ignores ICMP Echo requests sent to broadcast/multicast
addresses"
   echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
fi
```

Pour activer le port forwarding (par exemple forward du port 522 vers Client 192.168.0.5 port 22):

```
  ${ipt} -t nat -A PREROUTING -p tcp --dport 522 -j DNAT --to-destination
192.168.0.5:22
```

vérification

```
  iptables -L -t nat
  Chain PREROUTING (policy ACCEPT)
  target     prot opt source               destination
  DNAT       tcp  --  anywhere             anywhere            tcp dpt:522
to:192.168.0.5:22
```

pour autoriser active-ftp à partir d'un client, il faut charger 2 modules et définir 1 iptables-rule:

```
modprobe ip_conntrack_ftp
modprobe ip_nat_ftp ports=21

${ipt} -A INPUT -p tcp --sport 20 -m state --state
ESTABLISHED,RELATED -j ACCEPT
```