Network Configuration

From ArchWiki

This page explains how to set up a **wired** connection to a network. If you need to set up **wireless** networking see the Wireless Setup page.

Related articles

Jumbo Frames

Firewalls

Wireless Setup

List of Applications#Network Managers

Contents

Check the connection

Note: If you receive an error like `ping: icmp open socket: Operation not permitted` when executing ping, try to re-install the `iputils` package.

Many times, the basic installation procedure has created a working network configuration. To check if this is so, use the following command:

Note: The `-c 3` option calls it three times. See `man ping` for more information.

```
$ ping -c 3 www.google.com
```

```
PING www.l.google.com (74.125.224.146) 56(84) bytes of data.
64 bytes from 74.125.224.146: icmp_req=1 ttl=50 time=437 ms
64 bytes from 74.125.224.146: icmp_req=2 ttl=50 time=385 ms
64 bytes from 74.125.224.146: icmp_req=3 ttl=50 time=298 ms

--- www.l.google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 298.107/373.642/437.202/57.415 ms
```

If it works, then you may only wish to personalize your settings from the options below.

If the previous command complains about unknown hosts, it means that your machine was unable to resolve this domain name. It might be related to your service provider or your router/gateway. You can try pinging a static IP address to prove that your machine has access to the Internet.

```
$ ping -c 3 8.8.8.8
```

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_req=1 ttl=53 time=52.9 ms
```

```
64 bytes from 8.8.8.8: icmp_req=2 ttl=53 time=72.5 ms
64 bytes from 8.8.8.8: icmp_req=3 ttl=53 time=70.6 ms

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 52.975/65.375/72.543/8.803 ms
```

Note: `8.8.8.8` is a static address that is easy to remember. It is the address of Google's primary DNS server, therefore it can be considered reliable, and is generally not blocked by content filtering systems and proxies.

If you are able to ping this address, you may try adding this nameserver to your `/etc/resolv.conf` file.

Set the hostname

A hostname is a unique name created to identify a machine on a network: it is configured in `/etc/hostname`. The file can contain the system's domain name, if any. To set the hostname, do:

```
# hostnamectl set-hostname myhostname
```

This will put **myhostname** in `/etc/hostname`.

See `man 5 hostname` and `man 1 hostnamectl` for details.

Note:

- `hostnamectl` supports FQDNs
- You no longer need to edit `/etc/hosts`, `systemd (https://www.archlinux.org/packages/?name=systemd)` will provide host name resolution, and is installed on all systems by default.

To set the hostname temporarily (until a reboot), use the `hostname` command from `inetutils (https://www.archlinux.org/packages/?name=inetutils)`:

```
# hostname myhostname
```

Device Driver

Check the driver status

udev should detect your network interface card (NIC) and automatically load the necessary module at start up. Check the "Ethernet controller" entry (or similar) from the `lspci -v` output. It should tell you which kernel module contains the driver for your network device. For example:

```
$ lspci -v
```

```
02:00.0 Ethernet controller: Attansic Technology Corp. L1 Gigabit Ethernet Adapter (rev b0)
        ...
```

```
        Kernel driver in use: atl1
        Kernel modules: atl1
```

Next, check that the driver was loaded via `dmesg | grep module_name`. For example:

```
$ dmesg | grep atl1
    ...
    atl1 0000:02:00.0: eth0 link is up 100 Mbps full duplex
```

Skip the next section if the driver was loaded successfully. Otherwise, you will need to know which module is needed for your particular model.

Load the device module

Google for the right module/driver for the chipset. Some common modules are `8139too` for cards with a Realtek chipset, or `sis900` for cards with a SiS chipset. Once you know which module to use, try to load it manually. If you get an error saying that the module was not found, it's possible that the driver is not included in Arch kernel. You may search the AUR for the module name.

If udev is not detecting and loading the proper module automatically during bootup, see Kernel modules#Loading.

Network Interfaces

Device names

For computers with multiple NICs, it is important to have fixed device name. Many configuration problems are caused by interface name changing.

udev is responsible for which device gets which name. Systemd v197 introduced Predictable Network Interface Names (http://www.freedesktop.org/wiki/Software/systemd/PredictableNetworkInterfaceNames), which automatically assigns static names to network devices. Interfaces are now prefixed with `en` (ethernet), `wl` (WLAN), or `ww` (WWAN) followed by an automatically generated identifier, creating an entry such as `enp0s25`.

This behavior may be disabled by adding a symlink:

```
# ln -s /dev/null /etc/udev/rules.d/80-net-name-slot.rules
```

Users upgrading from an earlier systemd version will have a blank rules file created automatically. So if you want to use persistent device names, just delete the file.

Tip: You can run `ip link` or `ls /sys/class/net` to list all available interfaces.

Note: When changing the interface naming scheme, do not forget to update all network-related configuration files and custom systemd unit files to reflect the change. Specifically, if you have netctl static profiles enabled, run `netctl reenable profile` to update the generated service file.

Change device name

You can change the device name by defining the name manually with an udev-rule. For example:

```
/etc/udev/rules.d/10-network.rules
```

```
SUBSYSTEM=="net", ACTION=="add", ATTR{address}=="aa:bb:cc:dd:ee:ff", NAME="net1"
SUBSYSTEM=="net", ACTION=="add", ATTR{address}=="ff:ee:dd:cc:bb:aa", NAME="net0"
```

A couple things to note:

- To get the MAC address of each card, use this command: `cat /sys/class/net/device_name/address`
- Make sure to use the lower-case hex values in your udev rules. It doesn't like upper-case.

If you network card has dynamic MAC, you can use DEVPATH for example

```
/etc/udev/rules.d/10-network.rules
```

```
SUBSYSTEM=="net", DEVPATH=="/devices/platform/wemac.*", NAME="int"
```

Note: When choosing the static names **it should be avoided to use names in the format of "eth*X*" and "wlan*X*"**, because this may lead to race conditions between the kernel and udev during boot. Instead, it is better to use interface names that are not used by the kernel as default, e.g.: `net0` , `net1` , `wifi0` , `wifi1` . For further details please see the systemd (http://www.freedesktop.org/wiki/Software/systemd /PredictableNetworkInterfaceNames) documentation.

Set device MTU and queue Length

You can change the device MTU and queue length by defining manually with an udev-rule. For example:

```
/etc/udev/rules.d/10-network.rules
```

```
ACTION=="add", SUBSYSTEM=="net", KERNEL=="wl*", ATTR{mtu}="1480", ATTR{tx_queue_len}="2000"
```

Get current device names

Current NIC names can be found via sysfs

```
$ ls /sys/class/net
```

```
lo eth0 eth1 firewire0
```

Enabling and disabling network interfaces

You can activate or deactivate network interfaces using:

```
# ip link set eth0 up
```

```
# ip link set eth0 down
```

To check the result:

```
$ ip link show dev eth0
```

```
2: eth0: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master br0 state UP mode DEFAULT ql
...
```

Configure the IP address

You have two options: a dynamically assigned address using DHCP, or an unchanging "static" address.

Dynamic IP address

Manually run DHCP Client Daemon

Note: `dhcpcd` (DHCP *client* daemon) is not the same as `dhcpd` (DHCP *(server)* daemon).

```
# dhcpcd eth0
```

```
dhcpcd: version 5.1.1 starting
dhcpcd: eth0: broadcasting for a lease
...
dhcpcd: eth0: leased 192.168.1.70 for 86400 seconds
```

And now, `ip addr show dev eth0` should show your inet address.

For some people, `dhclient` (from the `dhclient (https://www.archlinux.org/packages/?name=dhclient)` package) works where `dhcpcd` fails.

Run DHCP at boot

If you simply want to use DHCP for your Ethernet connection, you can use `dhcpcd@.service` (provided by the `dhcpcd (https://www.archlinux.org/packages/?name=dhcpcd)` package).

To start DHCP for `eth0` , simply use:

```
# systemctl start dhcpcd@eth0.service
```

You can enable the service to automatically start at boot with:

```
# systemctl enable dhcpcd@eth0.service
```

If you use DHCP and you do **not** want your DNS servers automatically assigned every time you start your network, be sure to add the following to the last section of `dhcpcd.conf` :

/etc/dhcpcd.conf

```
nohook resolv.conf
```

Also, if you are on a network with DHCPv4 that filters Client IDs based on MAC addresses, you may need to change the following line:

/etc/dhcpcd.conf

```
# Use the same DUID + IAID as set in DHCPv6 for DHCPv4 Client ID as per RFC4361.
duid
```

To:

/etc/dhcpcd.conf

```
# Use the hardware address of the interface for the Client ID (DHCPv4).
clientid
```

Else, you may not obtain a lease since the DHCP server may not read your DHCPv6-style (http://en.wikipedia.org/wiki/DHCPv6) Client ID correctly. See RFC 4361 (http://tools.ietf.org/html/rfc4361) for more information.

To prevent `dhcpcd` from adding domain name servers to `/etc/resolv.conf`, use the `nooption` option:

/etc/dhcpcd.conf

```
nooption domain_name_servers
```

Then add your own DNS name server to `/etc/resolv.conf`.

You may use the `openresolv (https://www.archlinux.org/packages/?name=openresolv)` package if several different processes want to control `/etc/resolv.conf` (e.g. `dhcpcd (https://www.archlinux.org/packages/?name=dhcpcd)` and a VPN client). No additional configuration for `dhcpcd (https://www.archlinux.org/packages/?name=dhcpcd)` is needed to use `openresolv (https://www.archlinux.org/packages/?name=openresolv)`.

DHCP static route(s)

If you need to add a static route client-side, create a new dhcpcd hook-script in `/usr/lib/dhcpcd/dhcpcd-hooks`. The example shows a new hook-script which adds a static route to a VPN subnet on 10.11.12.0/24 via a gateway machine at 192.168.192.5:

/usr/lib/dhcpcd/dhcpcd-hooks/40-vpnroute

```
ip route add 10.11.12.0/24 via 192.168.192.5
```

The *40* prefix means that it is the final hook-script to run when dhcpcd starts.

Static IP address

There are various reasons why you may wish to assign static IP addresses on your network. For instance, one may gain a certain degree of predictability with unchanging addresses, or you may not have a DHCP server available.

Note: If you share your Internet connection from a Windows machine without a router, be sure to use static IP addresses on both computers to avoid LAN problems.

You need:

- Static IP address
- Subnet mask
- Broadcast address
- Gateway's IP address

If you are running a private network, it is safe to use IP addresses in 192.168.*.* for your IP addresses, with a subnet mask of 255.255.255.0 and a broadcast address of 192.168.*.255. The gateway is usually 192.168.*.1 or 192.168.*.254.

Manual assignment

You can assign a static IP address in the console:

```
# ip addr add <IP address>/<subnet mask> broadcast <broadcast> dev <interface>
```

For example:

```
# ip addr add 192.168.1.2/24 broadcast 192.168.1.255 dev eth0
```

Note: The subnet mask was specified using CIDR notation.

For more options, see `man ip`.

Add your gateway like so:

```
# ip route add default via <default gateway IP address>
```

For example:

```
# ip route add default via 192.168.1.1
```

If you the get the error "No such process", it means you have to run `ip link set dev eth0 up` as root.

Manual connection at boot using systemd

Ensure that dhcpcd is stopped and disabled:

```
# systemctl status dhcpcd.service
```

Create a configuration file for the systemd service. Replace `<interface>` with the proper network interface name:

```
/etc/conf.d/network@<interface>
```

```
address=192.168.0.15
netmask=24
broadcast=192.168.0.255
gateway=192.168.0.1
```

Create a systemd unit file:

```
/etc/systemd/system/network@.service
```

```
[Unit]
Description=Network connectivity (%i)
Wants=network.target
Before=network.target
BindsTo=sys-subsystem-net-devices-%i.device
After=sys-subsystem-net-devices-%i.device

[Service]
Type=oneshot
RemainAfterExit=yes
EnvironmentFile=/etc/conf.d/network@%i

ExecStart=/usr/bin/ip link set dev %i up
ExecStart=/usr/bin/ip addr add ${address}/${netmask} broadcast ${broadcast} dev %i
ExecStart=/usr/bin/ip route add default via ${gateway}

ExecStop=/usr/bin/ip addr flush dev %i
ExecStop=/usr/bin/ip link set dev %i down

[Install]
WantedBy=multi-user.target
```

Enable the unit and start it, passing the name of the interface, in this case eno1:

```
# systemctl enable network@eno1.service
# systemctl start network@eno1.service
```

If you need to manually set the nameservers, edit `/etc/resolv.conf` :

```
/etc/resolv.conf
```

```
nameserver 208.67.222.222
nameserver 208.67.220.220
```

If `/etc/resolv.conf` keeps getting overwritten see Resolv.conf for more details on how to address this issue.

There are several different ways to utilise systemd to start the network using similar, but different, file contents, and it is also possible to hard code the parameters that can then be picked up from the file referenced in the

EnvironmentFile definition. It is therefore possible, as an alternative, to put the interface name into the EnvironmentFile in /etc/conf.d/ and then use the interface name directly instead of %i. In that case the name of the file does not need the "@". It is also useful to read the documentation on systemd service files at http://www.freedesktop.org/software/systemd/man/systemd.unit.html

Calculating addresses

You can use `ipcalc` provided by the `ipcalc (https://www.archlinux.org/packages/?name=ipcalc)` package to calculate IP broadcast, network, netmask, and host ranges for more advanced configurations. For example, I use ethernet over firewire to connect a windows machine to arch. For security and network organization, I placed them on their own network and configured the netmask and broadcast so that they are the only 2 machines on it. To figure out the netmask and broadcast addresses for this, I used ipcalc, providing it with the IP of the arch firewire nic 10.66.66.1, and specifying ipcalc should create a network of only 2 hosts.

```
$ ipcalc -nb 10.66.66.1 -s 1
```

```
Address:   10.66.66.1

Netmask:   255.255.255.252 = 30
Network:   10.66.66.0/30
HostMin:   10.66.66.1
HostMax:   10.66.66.2
Broadcast: 10.66.66.3
Hosts/Net: 2                     Class A, Private Internet
```

Load configuration

To test your settings either reboot the computer or reload the relevant systemd services:

```
# systemctl restart dhcpcd@eth0
```

Try pinging your gateway, DNS server, ISP provider and other Internet sites, in that order, to detect any connection problems along the way, as in this example:

```
$ ping -c 3 www.google.com
```

Additional settings

ifplugd for laptops

> **Tip:** `dhcpcd (https://www.archlinux.org/packages/?name=dhcpcd)` provides the same feature out of the box.

`ifplugd (https://www.archlinux.org/packages/?name=ifplugd)` in Official Repositories is a daemon which will automatically configure your Ethernet device when a cable is plugged in and automatically unconfigure it if the cable is pulled. This is useful on laptops with onboard network adapters, since it will only configure the interface when a cable is really connected. Another use is when you just need to restart the network but do not want to restart the computer or do it from the shell.

By default it is configured to work for the `eth0` device. This and other settings like delays can be configured in `/etc/ifplugd/ifplugd.conf`.

Note: Netctl package includes `netctl-ifplugd@.service`, otherwise you can use `ifplugd@.service` from `ifplugd (https://www.archlinux.org/packages/?name=ifplugd)` package. Use for example `systemctl enable ifplugd@eth0.service`.

Bonding or LAG

See netctl#Bonding.

IP address aliasing

IP aliasing is the process of adding more than one IP address to a network interface. With this, one node on a network can have multiple connections to a network, each serving a different purpose. Typical uses are virtual hosting of Web and FTP servers, or reorganizing servers without having to update any other machines (this is especially useful for nameservers).

Example

You will need `netctl (https://www.archlinux.org/packages/?name=netctl)` from the Official Repositories.

Prepare the configuration:

/etc/netctl/mynetwork

```
Connection='ethernet'
Description='Five different addresses on the same NIC.'
Interface='eth0'
IP='static'
Address=('192.168.1.10' '192.168.178.11' '192.168.1.12' '192.168.1.13' '192.168.1.14' '192.168.1.15')
Gateway='192.168.1.1'
DNS=('192.168.1.1')
```

Then simply execute:

```
$ netctl start mynetwork
```

Change MAC/hardware address

See MAC Address Spoofing.

Internet Sharing

See Internet Sharing.

Router Configuration

See Router.

Local network hostname resolution

The pre-requisite is to #Set the hostname after which hostname resolution works on the local system itself

```
$ ping hostname
```

```
PING hostname (192.168.1.2) 56(84) bytes of data.
64 bytes from hostname (192.168.1.2): icmp_seq=1 ttl=64 time=0.043 ms
```

To enable other machines to address the host by name, either a manual configuration of the respective `/etc/hosts` files or a service to propagate/resolve the name is required.

When setting up a DNS server such as BIND or Unbound is overkill, manually editing your `/etc/hosts` is too cumbersome, or when you want more flexibility with dynamic leaving and joining of hosts to the network, it is possible to handle hostname resolution on your local network using zero-configuration networking. There are two options available:

- Samba provides hostname resolution via Microsoft's **NetBIOS**. It only requires installation of `samba (https://www.archlinux.org/packages/?name=samba)` and enabling of the `nmbd.service` service. Computers running Windows, OS X, or Linux with `nmbd` running, will be able to find your machine.
- Avahi provides hostname resolution via **zeroconf**, also known as Avahi or Bonjour. It requires slightly more complex configuration than Samba: see Avahi#Hostname resolution for details. Computers running OS X, or Linux with an Avahi daemon running, will be able to find your machine. Windows does not have an built-in Avahi client or daemon.

Troubleshooting

Swapping computers on the cable modem

Some cable ISPs (videotron for example) have the cable modem configured to recognize only one client PC, by the MAC address of its network interface. Once the cable modem has learned the MAC address of the first PC or equipment that talks to it, it will not respond to another MAC address in any way. Thus if you swap one PC for another (or for a router), the new PC (or router) will not work with the cable modem, because the new PC (or router) has a MAC address different from the old one. To reset the cable modem so that it will recognise the new PC, you must power the cable modem off and on again. Once the cable modem has rebooted and gone fully online again (indicator lights settled down), reboot the newly connected PC so that it makes a DHCP request, or manually make it request a new DHCP lease.

If this method does not work, you will need to clone the MAC address of the original machine. See also Change MAC/hardware address.

The TCP window scaling problem

TCP packets contain a "window" value in their headers indicating how much data the other host may send in return. This value is represented with only 16 bits, hence the window size is at most 64Kb. TCP packets are cached for a while (they have to be reordered), and as memory is (or used to be) limited, one host could easily

run out of it.

Back in 1992, as more and more memory became available, RFC 1323 (http://www.faqs.org/rfcs/rfc1323.html) was written to improve the situation: Window Scaling. The "window" value, provided in all packets, will be modified by a Scale Factor defined once, at the very beginning of the connection.

That 8-bit Scale Factor allows the Window to be up to 32 times higher than the initial 64Kb.

It appears that some broken routers and firewalls on the Internet are rewriting the Scale Factor to 0 which causes misunderstandings between hosts.

The Linux kernel 2.6.17 introduced a new calculation scheme generating higher Scale Factors, virtually making the aftermaths of the broken routers and firewalls more visible.

The resulting connection is at best very slow or broken.

How to diagnose the problem

First of all, let's make it clear: this problem is odd. In some cases, you will not be able to use TCP connections (HTTP, FTP, ...) at all and in others, you will be able to communicate with some hosts (very few).

When you have this problem, the `dmesg` 's output is OK, logs are clean and `ip addr` will report normal status... and actually everything appears normal.

If you cannot browse any website, but you can ping some random hosts, chances are great that you're experiencing this problem: ping uses ICMP and is not affected by TCP problems.

You can try to use Wireshark. You might see successful UDP and ICMP communications but unsuccessful TCP communications (only to foreign hosts).

How to fix it (The bad way)

To fix it the bad way, you can change the tcp_rmem value, on which Scale Factor calculation is based. Although it should work for most hosts, it is not guaranteed, especially for very distant ones.

```
# echo "4096 87380 174760" > /proc/sys/net/ipv4/tcp_rmem
```

How to fix it (The good way)

Simply disable Window Scaling. Since Window Scaling is a nice TCP feature, it may be uncomfortable to disable it, especially if you cannot fix the broken router. There are several ways to disable Window Scaling, and it seems that the most bulletproof way (which will work with most kernels) is to add the following line to `/etc/sysctl.d/99-disable_window_scaling.conf` (see also sysctl)

```
net.ipv4.tcp_window_scaling = 0
```

How to fix it (The best way)

This problem is caused by broken routers/firewalls, so let's change them. Some users have reported that the broken router was their very own DSL router.

More about it

This section is based on the LWN article TCP window scaling and broken routers (http://lwn.net/Articles/92727/) and a Kernel Trap article: Window Scaling on the Internet (http://kerneltrap.org/node/6723).

There are also several relevant threads on the LKML.

Realtek no link / WOL problem

Users with Realtek 8168 8169 8101 8111(C) based NICs (cards / and on-board) may notice a problem where the NIC seems to be disabled on boot and has no Link light. This can usually be found on a dual boot system where Windows is also installed. It seems that using the offical Realtek drivers (dated anything after May 2007) under Windows is the cause. These newer drivers disable the Wake-On-LAN feature by disabling the NIC at Windows shutdown time, where it will remain disabled until the next time Windows boots. You will be able to notice if this problem is affecting you if the Link light remains off until Windows boots up; during Windows shutdown the Link light will switch off. Normal operation should be that the link light is always on as long as the system is on, even during POST. This problem will also affect other operative systems without newer drivers (eg. Live CDs). Here are a few fixes for this problem:

Method 1 - Rollback/change Windows driver

You can roll back your Windows NIC driver to the Microsoft provided one (if available), or roll back/install an official Realtek driver pre-dating May 2007 (may be on the CD that came with your hardware).

Method 2 - Enable WOL in Windows driver

Probably the best and the fastest fix is to change this setting in the Windows driver. This way it should be fixed system-wide and not only under Arch (eg. live CDs, other operative systems). In Windows, under Device Manager, find your Realtek network adapter and double-click it. Under the Advanced tab, change "Wake-on-LAN after shutdown" to Enable.

```
In Windows XP (example)
Right click my computer
--> Hardware tab
  --> Device Manager
    --> Network Adapters
      --> "double click" Realtek ...
        --> Advanced tab
          --> Wake-On-Lan After Shutdown
            --> Enable
```

Note: Newer Realtek Windows drivers (tested with *Realtek 8111/8169 LAN Driver v5.708.1030.2008*, dated 2009/01/22, available from GIGABYTE) may refer to this option slightly differently, like *Shutdown Wake-On-LAN --> Enable*. It seems that switching it to `Disable` has no effect (you will notice the Link light still turns off upon Windows shutdown). One rather dirty workaround is to boot to Windows and just reset the system (perform an ungraceful restart/shutdown) thus not giving the Windows driver a chance to disable LAN. The Link light will remain on and the LAN adapter will remain accessible after POST - that is until you

boot back to Windows and shut it down properly again.

Method 3 - Newer Realtek Linux driver

Any newer driver for these Realtek cards can be found for Linux on the realtek site. (untested but believed to also solve the problem).

Method 4 - Enable *LAN Boot ROM* in BIOS/CMOS

It appears that setting *Integrated Peripherals --> Onboard LAN Boot ROM --> Enabled* in BIOS/CMOS reactivates the Realtek LAN chip on system boot-up, despite the Windows driver disabling it on OS shutdown.

Note: This was tested successfully multiple times with GIGABYTE system board GA-G31M-ES2L with BIOS version F8 released on 2009/02/05. YMMV.

DLink G604T/DLink G502T DNS problem

Users with a DLink G604T/DLink G502T router, using DHCP and have firmware v2.00+ (typically users with AUS firmware) may have problems with certain programs not resolving the DNS. One of these programs are unfortunatley pacman. The problem is basically the router in certain situations is not sending the DNS properly to DHCP, which causes programs to try and connect to servers with an IP address of 1.0.0.0 and fail with a connection timed out error

How to diagnose the problem

The best way to diagnose the problem is to use Firefox/Konqueror/links/seamonkey and to enable wget for pacman. If this is a fresh install of Arch Linux, then you may want to consider installing `links` through the live CD.

Firstly, enable wget for pacman (since it gives us info about pacman when it is downloading packages) Open `/etc/pacman.conf` with your favourite editor and uncomment the following line (remove the # if it is there)

```
XferCommand=/usr/bin/wget --passive-ftp -c -O %o %u
```

While you are editing `/etc/pacman.conf` , check the default mirror that pacman uses to download packages.

Now open up the default mirror in an Internet browser to see if the mirror actually works. If it does work, then do `pacman -Syy` (otherwise pick another working mirror and set it to the pacman default). If you get something similar to the following (notice the 1.0.0.0),

```
ftp://mirror.pacific.net.au/linux/archlinux/extra/os/i686/extra.db.tar.gz
           => '/var/lib/pacman/community.db.tar.gz.part'
Resolving mirror.pacific.net.au... 1.0.0.0
```

then you most likely have this problem. The 1.0.0.0 means it is unable to resolve DNS, so we must add it to `/etc/resolv.conf` .

How to fix it

Basically what we need to do is to manually add the DNS servers to our `/etc/resolv.conf` file. The problem is that DHCP automatically deletes and replaces this file on boot, so we need to edit `/etc/conf.d/dhcpcd` and change the flags to stop DHCP from doing this.

When you open `/etc/conf.d/dhcpcd`, you should see something close to the following:

```
DHCPCD_ARGS="-t 30 -h $HOSTNAME"
```

Add the `-R` flag to the arguments, e.g.,

```
DHCPCD_ARGS="-R -t 30 -h $HOSTNAME"
```

Note: If you are using `dhcpcd (https://www.archlinux.org/packages/?name=dhcpcd)` >= 4.0.2, the `-R` flag has been deprecated. Please see the #Run DHCP at boot section for information on how to use a custom `/etc/resolv.conf` file.

Save and close the file; now open `/etc/resolv.conf`. You should see a single nameserver (most likely 10.1.1.1). This is the gateway to your router, which we need to connect to in order to get the DNS servers of your ISP. Paste the IP address into your browser and log in to your router. Go to the DNS section, and you should see an IP address in the Primary DNS Server field; copy it and paste it as a nameserver **ABOVE** the current gateway one.

For example, `/etc/resolv.conf` should look something along the lines of:

```
nameserver 10.1.1.1
```

If my primary DNS server is 211.29.132.12, then change `/etc/resolv.conf` to:

```
nameserver 211.29.132.12
nameserver 10.1.1.1
```

Now restart the network daemon by running `systemctl restart dhcpcd@<interface>` and do `pacman -Syy`. If it syncs correctly with the server, then the problem is solved.

More about it

This is the whirlpool forum (Australian ISP community) which talks about and gives the same solution to the problem:

http://forums.whirlpool.net.au/forum-replies-archive.cfm/461625.html

Check DHCP problem by releasing IP first

Problem may occur when DHCP get wrong IP assignment. For example when two routers are tied together through VPN. The router that is connected to me by VPN may assigning IP address. To fix it. On a console, as root, release IP address:

```
# dhcpcd -k
```

Then request a new one:

```
# dhcpcd
```

Maybe you had to run those two commands many times.

No eth0 with Atheros AR8161

Note: With the 3.10.2-1-ARCH kernel update, the alx ethernet driver module is included in the package.

With the Atheros AR8161 Gigabit Ethernet card, the ethernet connection is not working out-of-the-box (with the installation media of March 2013). The module "alx" needs to be loaded but is not present.

The driver from compat-wireless (http://linuxwireless.org/en/users/Download/stable/#compat-wireless_stable_releases) (that has become compat-drives (https://backports.wiki.kernel.org/index.php/Releases) since linux 3.7) need to be installed. The "-u" postfix annotates that Qualcomm have applied a driver under a unified driver.

```
$ wget https://www.kernel.org/pub/linux/kernel/projects/backports/2013/03/28/compat-drivers-2013-03-28-5-u.tar.b
$ tar xjf compat*
$ cd compat*
$ ./scripts/driver-select alx
$ make
$ sudo make install
$ sudo modprobe alx
```

The alx driver has not been added to Linux kernel due to various problems. Compatibility between the different kernel versions has been spotty. For better support follow the mailing list (http://lists.infradead.org/mailman/listinfo/unified-drivers)and alx page (http://www.linuxfoundation.org/collaborate/workgroups/networking/alx)for latest working solution for alx.

The driver must be built and installed after every kernel change.

Alternatively you can use the AUR package for compat drivers (https://aur.archlinux.org/packages/compat-drivers-patched/), which installs many other drivers.

No eth0 with Atheros AR9485

The ethernet (eth0) for Atheros AR9485 are not working out-of-the-box (with installation media of March 2013). The working solution for this is to install the package compat-drivers-patched (https://aur.archlinux.org/packages/compat-drivers-patched/) from AUR.

No carrier / no connection after suspend

After suspend to RAM no connection is found although the network cable is plugged in. This may be caused by PCI power management. What is the output of

```
# ip link show eth0
```

If the line contains "NO-CARRIER" even though there's a cable connected to your eth0 port, it is possible that the device was auto-suspended and the media sense feature doesn't work. To solve this, first you need to find your ethernet controllers PCI address by

```
# lspci
```

This should look similar to this:

```
...
00:19.0 Ethernet controller: Intel Corporation 82577LM Gigabit Network Connection (rev 06)
...
```

So the address is 00:19.0. Now check the PM status of the device by issuing

```
# cat "/sys/bus/pci/devices/0000:00:19.0/power/control"
```

substituting 00:19.0 with the address obtained from lspci. If the output reads "auto", you can try to bring the device out of suspend by

```
# echo on > "/sys/bus/pci/devices/0000:00:19.0/power/control"
```

Don't forget to substitute the address again.

Note: This appears to be a bug in kernel 3.8.4.1- (3.8.8.1 is still affected): Forum discussion. (https://bbs.archlinux.org/viewtopic.php?id=159837&p=2) It also appears a fix is on the way. (It will be likely fixed in 3.9.) (https://lkml.org/lkml/2013/1/18/147) In the meantime, the above is a suitable workaround.

Broadcom BCM57780

This Broadcom chipset sometimes does not behave well unless you specify the order of the modules to be loaded. The modules are `broadcom` and `tg3` , the former needing to be loaded first.

These steps should help if your computer has this chipset:

```
$ lspci | grep Ethernet
02:00.0 Ethernet controller: Broadcom Corporation NetLink BCM57780 Gigabit Ethernet PCIe (rev 01)
```

If your wired networking is not functioning in some way or another, try unplugging your cable then doing the following (as root):

```
# modprobe -r tg3
```

```
# modprobe broadcom
# modprobe tg3
```

Now plug you network cable in. If this solves your problems you can make this permanent by adding `broadcom` and `tg3` (in this order) to the `MODULES` array in `/etc/mkinitcpio.conf` :

```
MODULES=".. broadcom tg3 .."
```

Then rebuild the initramfs:

```
# mkinitcpio -p linux
```

Note: These methods may work for other chipsets, such as BCM57760.

Retrieved from "https://wiki.archlinux.org/index.php?title=Network_Configuration&oldid=292235"
Categories: Networking | Getting and installing Arch
