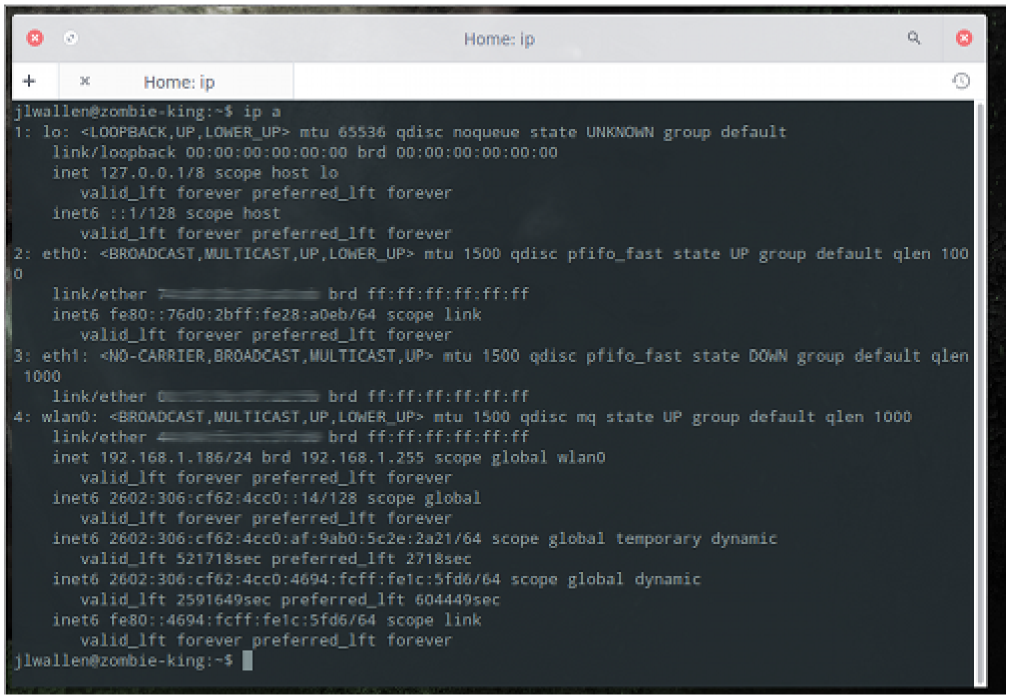
Published on *Linux.com* | *The source for Linux information* (https://www.linux.com)

Replacing ifconfig with ip [1]

Submitted by jlwallen [2] on August 14, 2015



```
jlwallen@zombie-king:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 100
0
    link/ether [redacted] brd ff:ff:ff:ff:ff:ff
    inet6 fe80::76d0:2bff:fe28:a0eb/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state DOWN group default qlen
 1000
    link/ether [redacted] brd ff:ff:ff:ff:ff:ff
4: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether [redacted] brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.186/24 brd 192.168.1.255 scope global wlan0
       valid_lft forever preferred_lft forever
    inet6 2602:306:cf62:4cc0::14/128 scope global
       valid_lft forever preferred_lft forever
    inet6 2602:306:cf62:4cc0:af:9ab0:5c2e:2a21/64 scope global temporary dynamic
       valid_lft 521718sec preferred_lft 2718sec
    inet6 2602:306:cf62:4cc0:4694:fcff:fe1c:5fd6/64 scope global dynamic
       valid_lft 2591649sec preferred_lft 604449sec
    inet6 fe80::4694:fcff:fe1c:5fd6/64 scope link
       valid_lft forever preferred_lft forever
jlwallen@zombie-king:~$ 
```

If you've been around Linux long enough, you know tools come and go. This was assumed to be the case back around 2009 when the debian-devel mailing list announced plans on deprecating the net-tools package due to lack of maintenance. It is now 2015 and net-tools [3] is still around. In fact, as of Ubuntu 14.10, you can still issue the ifconfig command to manage your network configuration.

However, in some instances (e.g., Ubuntu Docker [4]container), the net-tools suite isn't installed by default. This means the ifconfig command isn't available. Although you can install net-tools with the command

```
sudo apt-get install net-tools
```

it is most often recommended to move forward with the command that has replaced ifconfig. That command is ip, and it does a great job of stepping in for the out-of-date ifconfig.

Thing is, ip is not a drop-in replacement for ifconfig. There are differences in the structure of the commands. Even with these differences, both commands are used for similar purposes. In fact, ip can do the following:

- Discover which interfaces are configured on a system
- Query the status of a network interface
- Configure the network interfaces (including local loop-back, and Ethernet)

- Bring an interface up or down
- Manage both default and static routing
- Configure tunnel over IP
- Configure ARP or NDISC cache entry

With all of that said, let's embark on replacing ifconfig with ip. I'll offer a few examples of how the replacement command is used. Understand that this command does require admin privileges (so you'll either have to su to root or make use of sudo — depending upon your distribution). Because these commands can make changes to your machine's networking information, use them with caution.

NOTE: All addresses used in this how-to are examples. The addresses you will use will be dictated by your network and your hardware.

Now, on with the how-to.

Gathering Information

The first thing most people learn with the ifconfig command is how to find out what IP address has been assigned to an interface. This is usually done with the command ifconfig and no flags or arguments. To do the same with the ip command, it is run as such:

```
ip a
```

This command will list all interfaces with their associated information (Figure 1 above).

Let's say you only want to see IPv4 information (for clarity). To do this, issue the command:

```
ip -4 a
```

Or, if you only want to see IPv6 information:

```
ip -6 a
```

What if you only want to see information regarding a specific interface? You can list information for a wireless connection with the command:

```
ip a show wlan0
```

You can even get more specific with this command. If you only want to view IPv4 on the wlan0 interface, issue the command:

```
ip -4 a show wlan0
```

You can even list only the running interface using:

```
ip link ls up
```

Modifying an Interface

Now we get into the heart of the command… using it to modify an interface. Suppose you wanted to assign a specific address to the first ethernet interface, eth0. With the ifconfig command, that would look like:

```
ifconfig eth0 192.168.1.101
```

With the ip command, this now looks like:

```
ip a add 192.168.1.101/255.255.255.0 dev eth0
```

You could shorten this a bit with:

```
ip a add 192.168.1.101/24 dev eth0
```

Clearly, you will need to know the subnet mask of the address you are assigning.

What about deleting an address from an interface? With the ip command, you can do that as well. For example, to delete the address just assigned to eth0, issue the following command:

```
ip a del 192.168.1.101/24 dev eth0
```

What if you want to simply flush all addresses from all interfaces? The ip command has you covered with this command:

```
ip -s -s a f to 192.168.1.0/24
```

Another crucial aspect of the ip command is the ability to bring up/down an interface. To bring eth0 down, issue:

```
ip link set dev eth0 down
```

To bring eth0 back up, use:

```
ip link set dev eth0 up
```

With the ip command, you can also add and delete default gateways. This is handled like so:

```
ip route add default via 192.168.1.254
```

If you want to get really detailed on your interfaces, you can edit the transmit queue. You can set the transmit queue to a low value for slower interfaces and a higher value for faster interfaces. To do this, the command would look like:

```
ip link set txqueuelen 10000 dev eth0
```

The above command would set a high transmit queue. You can play around with this value to find what works best for your hardware.

You can also set the Maximum Transmission Unit (MTU) of your network interface with the command:

```
ip link set mtu 9000 dev eth0
```

Once you've made the changes, use ip a list eth0 to verify the changes have gone into effect.

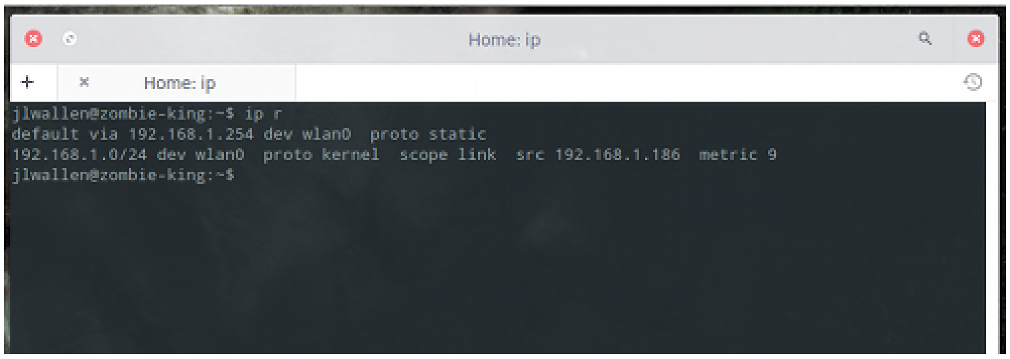
Managing the Routing Table

With the ip command you can also manage the system's routing tables. This is a very powerful element of the ip command, and you should use it with caution.

Suppose you want to view all routing tables. To do this, you would issue the command:

```
ip r
```

The output of this command will look like that shown in Figure 2.



```
jlwallen@zombie-king:~$ ip r
default via 192.168.1.254 dev wlan0  proto static
192.168.1.0/24 dev wlan0  proto kernel  scope link  src 192.168.1.186  metric 9
jlwallen@zombie-king:~$
```

Now, say you want to route all traffic via the 192.168.1.254 gateway connected via eth0 network interface: To do that, issue the command:

```
ip route add 192.168.1.0/24 dev eth0
```

To delete that same route, issue:

```
ip route del 192.168.1.0/24 dev eth0
```

This article should serve as merely an introduction to the ip command. This, of course, doesn't mean you must immediately jump from ifconfig. Because the deprecation of ifconfig has been so slow, the command still exists on many a distribution. But, on the occasion of ifconfig finally vanishing from sight, you'll be ready to make the transition with ease. For more detailed information on the ip command, take a look at the ip man page by issuing the command man ip from a terminal window.

Tutorial Category:
Tutorials [5]

Source URL: https://www.linux.com/learn/replacing-ifconfig-ip

Links:
[1] https://www.linux.com/learn/replacing-ifconfig-ip
[2] https://www.linux.com/users/jlwallen
[3] http://www.linuxfoundation.org/collaborate/workgroups/networking/net-tools
[4] https://docs.docker.com/installation/ubuntulinux/
[5] https://www.linux.com/tutorials/category/tutorials