|  |
| --- |
| **Linux TCP/IP Network Configuration Files:** |

|  |  |
| --- | --- |
| **File** | **Description** |
| /etc/resolve.conf | List DNS servers for internet domain name resolution. Manual page for: /etc/[resolv.conf](http://man.yolinux.com/cgi-bin/man2html?cgi_command=resolv.conf) |
| /etc/hosts | Lists hosts to be resolved locally (not by DNS). Manual page for: /etc/[hosts](http://man.yolinux.com/cgi-bin/man2html?cgi_command=resolv.conf) |
| /etc/nsswitch.conf | List order of host name search. Typically look at local files, then NIS server, then DNS server. |

|  |
| --- |
| **Domain Resolution Configuration Files:** |

* File: /etc/[resolv.conf](http://man.yolinux.com/cgi-bin/man2html?cgi_command=resolv.conf) **- host name resolver configuration file**

|  |
| --- |
| search *name-of-domain.com* - ***Name of your domain or ISP's domain if using their name server***  nameserver *XXX.XXX.XXX.XXX* - ***IP address of primary name server***  nameserver *XXX.XXX.XXX.XXX* - ***IP address of secondary name server*** |

* This configures Linux so that it knows which DNS server will be resolving domain names into IP addresses. If using DHCP client, this will automatically be sent to you by the ISP and loaded into this file as part of the DHCP protocol. If using a static IP address, ask the ISP or check another machine on your network.   
  Red Hat/Fedora GUI: /usr/sbin/system-config-network (select tab "DNS").
* File: /etc/[hosts](http://man.yolinux.com/cgi-bin/man2html?cgi_command=resolv.conf) **- locally resolve node names to IP addresses**

|  |
| --- |
| 127.0.0.1 *your-node-name.your-domain.com* localhost.localdomain localhost  *XXX.XXX.XXX.XXX* *node-name* |

* Note when adding hosts to this file, place the fully qualified name first. (It helps sendmail identify your server correctly) i.e.:
* *XXX.XXX.XXX.XXX* superserver.yolinux.com superserver
* This informs Linux of local systems on the network which are not handled by the DNS server. (or for all systems in your LAN if you are not using DNS or NIS)

|  |
| --- |
| **Assigning an IP address:** |

Computers may be assigned a static IP address or assigned one dynamically. Typically a server will require a static IP while a workstation will use DHCP (dynamic IP assignment). The Linux server requires a static IP so that those who wish to use its resources can find the system. It is more easily found if the IP address does not change and is static. This is not important for the Linux client workstation and thus it is easier to use an automated Dynamic Host Configuration Protocol (DHCP) for IP address assignment.

**Static IP address assignment:**

Choose one of the following methods:

* Command Line:

/sbin/[ifconfig](http://man.yolinux.com/cgi-bin/man2html?cgi_command=ifconfig) eth0 192.168.10.12 netmask 255.255.255.0 broadcast 192.168.10.255

Network address by convention would be the lowest: 192.168.10.0   
Broadcast address by convention would be the highest: 192.168.10.255   
The gateway can be anything, but following convention: 192.168.10.1

Note: the highest and lowest addresses are based on the netmask. The previous example is based on a netmask of 255.255.255.0

|  |
| --- |
| Command line IP Configuration: ifconfig |

ifconfig *interface* [aftype] options | address ...

where:

* interface: eth0, eth1, eth2 represent the computer ethernet interfaces
* aftype: inet (TCP/IP, default), inet6 (IPv6), ax25 (AMPR Packet Radio), ddp (Appletalk Phase 2), ipx (Novell IPX) or netrom (AMPR Packet radio)

Options:

|  |  |
| --- | --- |
| **Option** | **Description** |
| up | Activate the interface. Implied if IP addresses are specified. |
| down | Shut down interface |
| arp | Enable ARP protocol on this interface. Allow ARP to detect the addresses of computer hosts attached to the network. |
| -arp | Disable ARP protocol on this interface |
| promisc | Enable promiscuous mode. Receive all packets on the network not just those destined for this interface. |
| -promisc | Disable promiscuous mode. |
| mtu ## | Specify the Maximum Transfer Unit (MTU) of the interface. The MTU is the maximum number of octets the interface is able to handle in a single transaction. Defaults: Ethernet: 1500 SLIP: 296 |
| broadcast XXX.XXX.XXX.XXX | Set the network broadcast address for this interface. |
| netmask XXX.XXX.XXX.XXX | Set the IP network mask for this interface. |

|  |
| --- |
| **Network IP aliasing:** |

**Assign more than one IP address to one ethernet card:**

ifconfig eth0 *XXX.XXX.XXX.XXX* netmask 255.255.255.0 broadcast *XXX.XXX.XXX*.255

ifconfig eth0:0 192.168.10.12 netmask 255.255.255.0 broadcast 192.168.10.255

ifconfig eth0:1 192.168.10.14 netmask 255.255.255.0 broadcast 192.168.10.255

route add -host *XXX.XXX.XXX.XXX* dev eth0

route add -host 192.168.10.12 dev eth0

route add -host 192.168.10.14 dev eth0

In this example 0 and 1 are aliases in addition to the regular eth0. The result of the ifconfig command:

eth0 Link encap:Ethernet HWaddr 00:10:4C:25:7A:3F

inet addr:*XXX.XXX.XXX.XXX* Bcast:*XXX.XXX.XXX*.255 Mask:255.255.255.0

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:14218 errors:0 dropped:0 overruns:0 frame:0

TX packets:1362 errors:0 dropped:0 overruns:0 carrier:0

collisions:1 txqueuelen:100

Interrupt:5 Base address:0xe400

eth0:0 Link encap:Ethernet HWaddr 00:10:4C:25:7A:3F

inet addr:192.168.10.12 Bcast:192.168.10.255 Mask:255.255.255.0

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

Interrupt:5 Base address:0xe400

eth0:1 Link encap:Ethernet HWaddr 00:10:4C:25:7A:3F

inet addr:192.168.10.14 Bcast:192.168.10.255 Mask:255.255.255.0

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

Interrupt:5 Base address:0xe400

This is a three step process:

1. Issue the command: [hostname](http://man.yolinux.com/cgi-bin/man2html?cgi_command=hostname) *new-host-name*
2. Change network configuration file: /etc/sysconfig/network   
   Edit entry: HOSTNAME=*new-host-name*
3. Restart systems which relied on the hostname (or reboot):
   * Restart network services: service network restart   
     (or: /etc/init.d/network restart)
   * Restart desktop:
     + Bring down system to console mode: [init](http://man.yolinux.com/cgi-bin/man2html?cgi_command=init) 3
     + Bring up X-Windows: init 5

One may also want to check the file /etc/hosts for an entry using the system name which allows the system to be self aware.

The hostname may be changed at runtime using the command: [sysctl](http://man.yolinux.com/cgi-bin/man2html?cgi_command=sysctl) -w kernel.hostname="*superserver*"

Note that hostnames may only contain alphanumeric characters, minus signs ("-"), and periods ("."). They must begin with an alphabetic character and end with an alphanumeric character.

|  |
| --- |
| **Enable Forwarding:** |
|  |

Forwarding allows the network packets on one network interface (i.e. eth0) to be forwarded to another network interface (i.e. eth1). This will allow the Linux computer to connect ("ethernet bridge") or route network traffic.

The bridge configuration will merge two (or several) networks into one single network topology. IpTables firewall rules can be used to filter traffic.

A router configuration can support multicast and basic IP routing using the "route" command. IP masquerading (NAT) can be used to connect private local area networks (LAN) to the internet or load balance servers.

* Turn on IP forwarding to allow Linux computer to act as a gateway or router.   
  echo 1 > /proc/sys/net/ipv4/ip\_forward   
  Default is 0.

|  |
| --- |
| **Route:** |

The Linux OS manages outbound and inbound IP (Internet Protocol) traffic. Inbound traffic is captured based on ARP and IP address configuration. Outbound traffic is managed by routes. Routing determines the path these packets take so that they are sent to their destinations. This is required for all IP traffic, local and remote, including when multiple network interfaces are available. Routes are held by the kernel routing table.

Direct routing table entries occur when the source and destination hosts are on the same physical network and packets are sent directly from the source to the destination.

Indirect routing table entries occur when the source and destination hosts are on different physical networks. The destination host must be reached through one or more IP gateways. The first gateway is the only one which is known by the host system.

Default routing defines a gateway to use when the direct network route and the indirect host routes are not defined for a given IP address.

**Static routes:** IP uses a routing table to determine where packets should be sent. First the packet is examined to see if its' destination is for the local or remote network. If it is to be sent to a remote network, the routing table is consulted to determine the path. If there is no information in the routing table then the packet is sent to the default gateway. Static routes are set with the route command and with the configuration file:

* Red Hat/Fedora: /etc/sysconfig/network-scripts/route-eth0
* Red Hat 7: /etc/sysconfig/static-routes
* S.u.s.e. 9.2: /etc/sysconfig/network/routes

|  |
| --- |
| 10.2.3.0/16 via 192.168.10.254 |

See command: /etc/sysconfig/network-scripts/ifup-routes eth0

**Dynamic routes:** RIP (Routing Information Protocol) is used to define dynamic routes. If multiple routes are possible, RIP will choose the shortest route. (Fewest hops between routers not physical distance.) Routers use RIP to broadcast the routing table over UDP port 520. The routers would then add new or improved routes to their routing tables.

Man pages:

* [route](http://man.yolinux.com/cgi-bin/man2html?cgi_command=route) - show / manipulate the IP routing table (Static route)

Show routes:

|  |  |
| --- | --- |
| **Option** | **Description** |
| -n | display IP addresses. Do not resolve host names for faster results. |
| -e | Print more extensive information about routes. |
| -v | Verbose. |
| --help | Route command information. |

Manipulate routes:

|  |  |
| --- | --- |
| **Option** | **Description** |
| add or del or neither | Add or delete route information. If not specified then print route table information. |
| -host XXX.XXX.XXX.XXX | Add a single computer host identified by the IP address. |
| -net XXX.XXX.XXX.XXX | Add a network identified by the network address, to the route. |
| gw XXX.XXX.XXX.XXX | Specify the network gateway. |
| netmask XXX.XXX.XXX.XXX | Specify the network netmask. |
| default | Of all the routes specified, identify one as the default network route.  (typically the gateway is specified as the default route) |

Examples:

* + Show routing table: route -e
  + Access individual computer host specified via network interface card eth1:   
    route add -host 123.213.221.231 eth1
  + Access ISP network identified by the network address and netmask using network interface card eth0:   
    route add -net 10.13.21.0 netmask 255.255.255.0 gw 192.168.10.254 eth0   
    Conversely: route del -net 10.13.21.0 netmask 255.255.255.0 gw 192.168.10.254 eth0
  + Specify default gateway to use to access remote network via network interface card eth0:   
    route add default gw 201.51.31.1 eth0   
    (Gateway can also be defined in /etc/sysconfig/network)
  + Specify two gateways for two network destinations: (i.e. one external, one internal private network. Two routers/gateways will be specified.)   
    Add internet gateway as before: route add default gw 201.51.31.1 eth0   
    Add second private network: route add -net 10.0.0.0 netmask 255.0.0.0 gw 192.168.10.254 eth0
* [routed](http://man.yolinux.com/cgi-bin/man2html?cgi_command=routed) - network routing daemon. Uses RIP protocol to update routing table.

|  |
| --- |
| **Useful Linux networking commands:** |

* /etc/rc.d/init.d/network   start - command to start, restart or stop the network
* [netstat](http://man.yolinux.com/cgi-bin/man2html?cgi_command=netstat) - Display connections, routing tables, stats etc
  + List externally connected processes: netstat -punta
    - -a: Show both listening and non-listening sockets.
    - -p: Show PID of process owning socket
    - -u: Show UDP
    - -t: Show TCP
    - -n: Show IP addresses only. Don't resolve host names
    - -g: Show multi-cast group membership info
    - -c: Continuous mode - update info every second
    - -v: Verbose
    - -e: Extended information
    - -o: show network timer information
  + List all connected processes: netstat -nap
  + Show network statistics: netstat -s
  + Display routing table info: netstat -rn

|  |
| --- |
| $ netstat -nr  Kernel IP routing table  Destination Gateway Genmask Flags MSS Window irtt Iface  192.168.1.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0  169.254.0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth0  0.0.0.0 192.168.1.1 0.0.0.0 UG 0 0 0 eth0 |

* + Flags:
    - G: route uses gateway
    - U: Interface is "up"
    - H: Only a single host is accessible (eg. loopback)
    - D: Entry generated by ICMP redirect message
    - M: Modified by ICMP redirect message
  + Display interface statistics: netstat -i

|  |
| --- |
| $ netstat -i  Kernel Interface table  Iface MTU Met RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg  eth0 1500 0 2224 0 0 0 1969 0 0 0 BMRU  lo 16436 0 1428 0 0 0 1428 0 0 0 LRU |

* + Where:
    - RX-OK/TX-OK: number of packets transmitted/received error free
    - RX-ERR/TX-ERR: number of damaged/error packets transmitted/received
    - RX-DRP/TX-DRP: number of dropped packets
    - RX-OVR/TX-OVR: number of packets dropped because of a buffer overrun

Flags:

* + - B: A broadcast address has been set
    - L: This interface is a loopback device
    - M: All packets are received
    - N: Trailers are avoided
    - O: ARP is turned off for this interface
    - P: Point-to-point connection
    - R: Interface is running
    - U: Interface is up

|  |
| --- |
| inetd: |

Configuration file: /etc/inetd.conf   
Entries in this file consist of a single line made up of the following fields:

service socket-type protocol wait user server cmdline

* **service**: The name assigned to the service. Matches the name given in the file /etc/services
* **socket-type**:
  + **stream**: connection protocols (TCP)
  + **dgram**: datagram protocols (UDP)
  + **raw**
  + **rdm**
  + **seqpacket**
* **protocol**: Transport protocol name which matches a name in the file /etc/protocols. i.e. udp, icmp, tcp, rpc/udp, rpc/tcp, ip, ipv6
* **wait:** Applies only to datagram protocols (UDP).
  + **wait[.max]**: One server for the specified port at any time (RPC)
  + **nowait[.max]**: Continue to listen and launch new services if a new connection is made. (multi-threaded)

Max refers to the maximum number of server instances spawned in 60 seconds. (default=40)

* **user[.group]**: login id of the user the process is executed under. Often nobody, root or a special restricted id for that service.
* **server**: Full path name of the server program to be executed.
* **cmdline**: Command line to be passed to the server. This includes argument 0 (argv[0]), that is the command name. This field is empty for internal services. Example of internal TCP services: echo, discard, chargen (character generator), daytime (human readable time), and time (machine readable time). (see RFC)

Sample File: /etc/inetd.conf

|  |
| --- |
| #echo stream tcp nowait root internal  #echo dgram udp wait root internal  ftp stream tcp nowait root /usr/sbin/tcpd in.ftpd -l -a  #pop-3 stream tcp nowait root /usr/sbin/tcpd ipop3d  #swat stream tcp nowait.400 root /usr/sbin/swat swat |

A line may be commented out by using a '#' as the first character in the line. This will turn the service off. The maximum length of a line is 1022 characters.

The inet daemon must be restarted to pick up the changes made to the file:   
/etc/rc.d/init.d/inetd restart

For more information see the man pages "inetd" and "inetd.conf".

|  |
| --- |
| **ARP: Address Resolution Protocol** |

Ethernet hosts use the Address Resolution Protocol (ARP) to convert a 32-bit internet IP addresses into a 48-bit Ethernet MAC address used by network hardware. (See: [RFC 826](http://www.ietf.org/rfc/rfc0826.txt)) ARP broadcasts are sent to all hosts on the subnet by the data transmitting host to see who replies. The broadcast is ignored by all except the intended receiver which recognizes the IP address as its own. The MAC addresses are remembered (ARP cache) for future network communications. Computers on the subnet typically keep a cache of ARP responses (typically 20 min but can store permanent information for diskless nodes). ARP broadcasts are passed on by hubs and switches but are blocked by routers.

Reverse ARP (See: [RFC 903](http://www.ietf.org/rfc/rfc0903.txt)) is a bootstrap protocol which allows a client to broadcast requesting a server to reply with its IP address.

View ARP tables:

* Shows other systems on your network (including IP address conflicts): /sbin/arp -a
* Show ARP table Linux style: /sbin/arp -e
* List ARP table: cat /proc/net/arp

Note that the use of a switch instead of a hub will limit your view of other hosts. Typically all you will see in the arp table is your router or gateway.

Set/Configure ARP tables:

* Add a host's IP address: /sbin/arp -s *hostname* XX:XX:XX:XX:XX:XX pub
* Delete a host from the table: /sbin/arp -d *hostname*   
  This can be used to remove a duplicate IP or force a new interface to provide info.

Man pages:

* [arp (8) man page](http://man.yolinux.com/cgi-bin/man2html?cgi_command=arp) - manipulate the system ARP cache
* [arpwatch (8) man page](http://man.yolinux.com/cgi-bin/man2html?cgi_command=arpwatch) - keep track of ethernet/ip address pairings
* [arpsnmp (8) man page](http://man.yolinux.com/cgi-bin/man2html?cgi_command=arpsnmp) - keep track of ethernet/ip address pairings. Reads information generated by [snmpwalk](http://man.yolinux.com/cgi-bin/man2html?cgi_command=snmpwalk)
* [arping (8) man page](http://man.yolinux.com/cgi-bin/man2html?cgi_command=arping) - send ARP REQUEST to a neighbor host   
  Print ARP reply (similar to arp -a): arping *192.168.10.99*
* [ip (8) man page](http://man.yolinux.com/cgi-bin/man2html?cgi_command=ip&cgi_section=8) - show / manipulate routing, devices, policy routing and tunnels   
  View ARP table: ip neighbor

ARP is something that simply works. No Linux system configuration is necessary. It's all part of the ethernet and IP protocol. The aforementioned information is just part of the Linux culture of full visibility into what is going on.