Network Services Introduction

Linux Networking and ISC DHCP

Dr. Horia V. Corcalciuc
Horia Hulubei National Institute for R&D in Physics and Nuclear Engineering (IFIN-HH)

April 27, 2016

Introduction

- Network drivers are compiled in the Linux kernel either as **modules** or placed directly into the kernel in case of a **monolithic build**.
- Network drivers expose an **interface** once the module or kernel is loaded. The **network interface** can be used to send packets to the network that the computer is connected to.
- Once the driver module is loaded or the kernel boots, the **network interface** will be in a **down** state where it will receive packets but discard them until the **interface** has been configured.
- In order to configure an interface, Linux command-line tools are used in order to supply the **network interface** with parameters.
- An **interface** can be statically configured - by supplying parameters manually on the command line, or dynamically by polling a **Dynamic Host Configuration Protocol (DHCP)** server for information.
- A **network interface**'s configuration seldom resists a restart such that the interface must be configured again upon restart. Depending on the type of **Linux distribution**, the network interface can be configured in the relevant configuration files such that when the system reboots, the scripts responsible with configuring the network will bring the interface up.

The ifconfig Tool

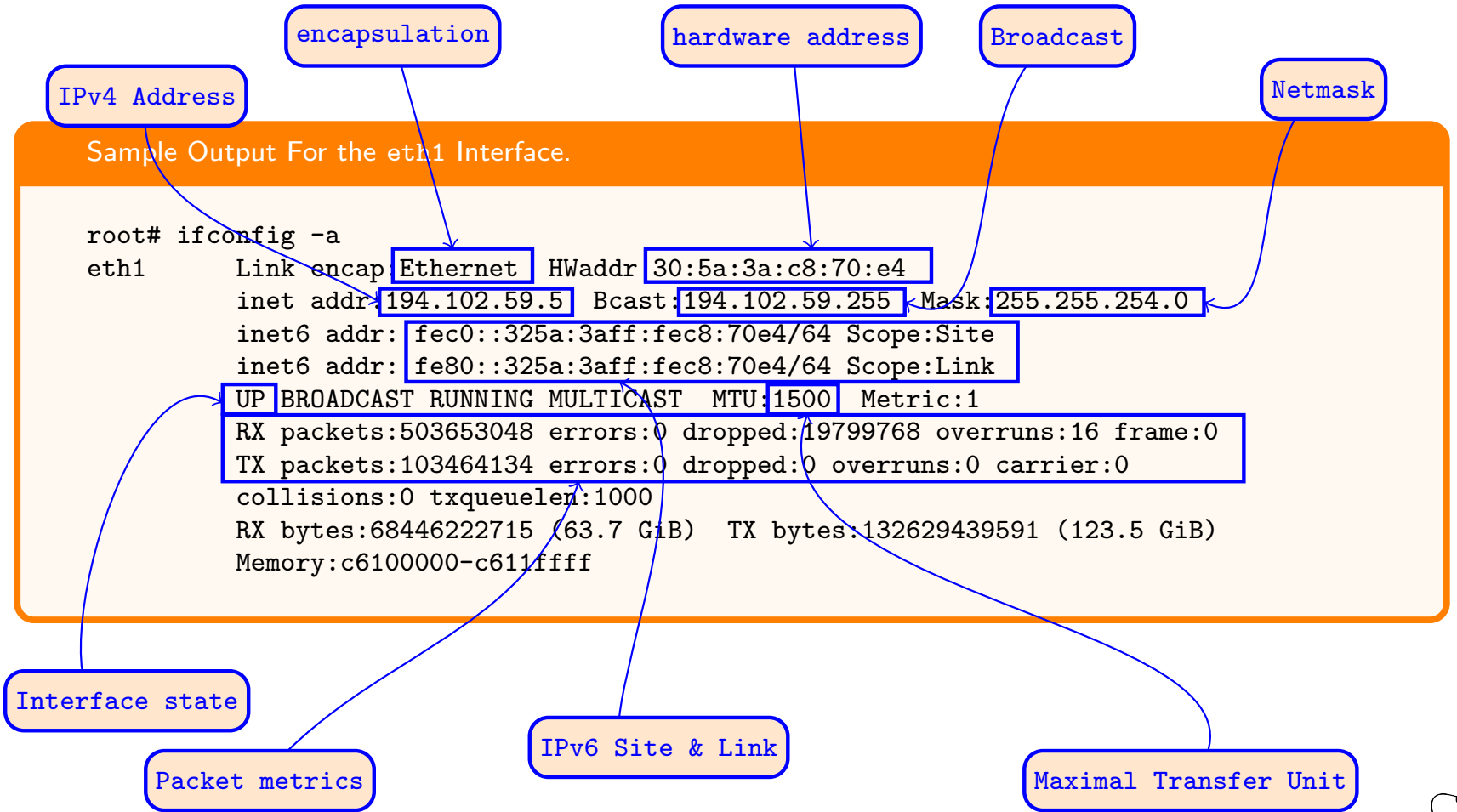
Sample Output For the eth1 Interface.

```
root# ifconfig -a
eth1      Link encap:Ethernet  HWaddr 30:5a:3a:c8:70:e4
          inet addr:194.102.59.5  Bcast:194.102.59.255  Mask:255.255.254.0
          inet6 addr: fec0::325a:3aff:fec8:70e4/64 Scope:Site
          inet6 addr: fe80::325a:3aff:fec8:70e4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:503653048 errors:0 dropped:19799768 overruns:16 frame:0
          TX packets:103464134 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:68446222715 (63.7 GiB)  TX bytes:132629439591 (123.5 GiB)
          Memory:c6100000-c611ffff
```

- The `ifconfig` (**if** - interface, **config** - configuration) tool is the main tool on all major Linux distributions (and BSD) that can be used from the command-line to configure or request information about a network interface.
- The `ifconfig` command has to be issued by a privileged user (conventionally, the `root` user) in order to be able to display the interfaces.
- The `-a` switch indicates to the `ifconfig` tool to list all available interfaces whether they are in an **up** or **down** state.

IFIN-HH

Interpreting ifconfig Output



IFIN-HH

ifconfig Remarks

Sample Output For the eth1 Interface.

```
root# ifconfig -a
eth1      Link encap:Ethernet  HWaddr 30:5a:3a:c8:70:e4
          inet addr:194.102.59.5  Bcast:194.102.59.255  Mask:255.255.254.0
          inet6 addr: fec0::325a:3aff:fec8:70e4/64 Scope:Site
          inet6 addr: fe80::325a:3aff:fec8:70e4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:503653048 errors:0 dropped:19799768 overruns:16 frame:0
          TX packets:103464134 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:68446222715 (63.7 GiB)  TX bytes:132629439591 (123.5 GiB)
          Memory:c6100000-c611ffff
```

- **IPv4** addresses are link-local only whilst **IPv6** have both link (with FE80::/10 prefix) and site addresses.
- The **hardware address** should be unique to the machine on the network and is stored inside the hardware EPROM - although, most operating systems allow **volatile changing** of the hardware address.
- Errors on both received (`RX`) and transmitted (`TX`) packets may sometimes indicate a hardware fault. Otherwise, it is normal for the interface to **drop** junk packets received from the network.
- The Maximal Transfer Unit (`MTU`) determines the maximal size of a sent packet before that packet must be fragmented. One can optimise the `MTU` through measurements and some hardware supports jumbo frames with an `MTU` reaching up to 9000.

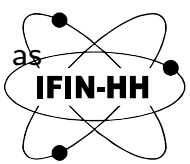
IFIN-HH

The Hardware Ethernet Tool

Sample Output For the `eth1` Interface.

```
root# ethtool eth1
Settings for eth1:
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Half 1000baseT/Full
        ...
        Advertised auto-negotiation: Yes
        Speed: 1000Mb/s
        Duplex: Full
        ...
        Auto-negotiation: on
        ...
        Supports Wake-on: pumbg
        Wake-on: g
        ...
        Link detected: yes
```

- Contrasted to `ifconfig` that configures the interface, `ethtool` allows setting hardware properties of the network interface.
- `ethtool` will show the supported link modes (**half duplex**, **full duplex**, etc...), it will show the speed at which the interface is running (that will be the **maximal speed**) as well as miscellaneous properties such as the **Wake-On-LAN** (`WOL`) settings for the adapter.



Advanced Hardware Ethernet Settings

Sample Output For the eth1 Interface.

```
root# ethtool eth1 -k | grep -v fixed
Features for eth1:
tx-checksumming: off
scatter-gather: off
tcp-segmentation-offload: off
generic-segmentation-offload: off [requested on]
generic-receive-offload: on
tx-nocache-copy: off
```

- Settings such as TCP segmentation offload (TSO), or Generic Segmentation Offload (GSO) can be turned on or off with ethtool.
- Some settings such as TSO can delegate the packet fragmentation to the interface hardware instead of using the machine's CPU. This allows the CPU to be freed and attend to other tasks but some of these features prove to degrade performance - for example, when virtualisation is involved.
- Not all settings can be changed and some will show up as [fixed] which means that the hardware does not allow settings the parameters.

Location and Purpose of Files

Files Relevant to Networking

File	Description
`/etc/services`	Reference for all services and on what ports (both UDP and TCP) they should be listening on.
`/etc/resolv.conf`	List of domain name servers queried by the machine. Lecture 1
`/etc/hosts`	A map of IP addresses to hosts. Lecture 1
`/etc/hosts.allow` `/etc/hosts.deny`	TCP wrappers allowing and rejecting connections from specified hosts. Not all daemons support TCP wrappers and these files are mostly obsolete on newer Linux distributions.
`/etc/nsswitch.conf`	General name service file - resolves users and hosts amongst others. Lecture 1
`/etc/rt_{tables, realms, scopes, etc...}`	Advanced IP routing specific to the Linux kernel (split routing, etc...). These are mostly relevant to the `iproute2` or `ip` command.
`/etc/named.conf`	The main configuration file for the DNS server. Lecture 1
`/etc/dhcpd.conf`	The main configuration file for the DHCP server. Lecture 1

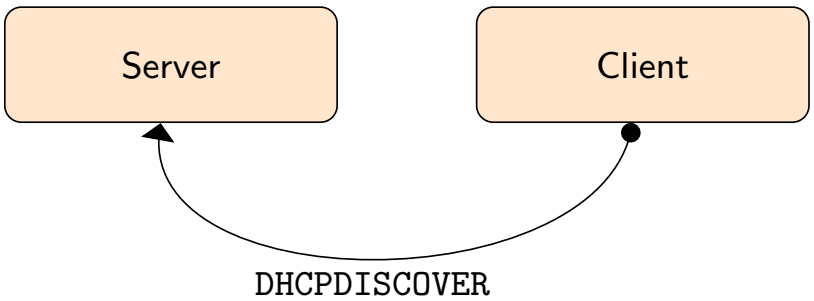
- The exact location of files is Linux **distribution specific** such that they may be placed in a different location depending on the used Linux distribution.

The Protocol and Tools

- A **Dynamic Host Configuration Protocol** (DHCP) server can be used for a network in order to feed connecting clients a network configuration.
- One of the most common DHCP servers is **ISC DHCP** that also comes with a DHCP client called `dhclient`. Other alternatives exist such as `dnsmasq` Lecture 1.
- Most of the configuration that can be set by `ifconfig` can be queried off a DHCP server as well as other settings relevant to DNS.
- A DHCP server usually listens for packets on on UDP port 67 which is documented in the protocol specification in RFC 2131. Note that the older **Bootstrap Protocol** (BOOTP) should also be receiving packets on the same port and that **ISC DHCP** has support for BOOTP clients.
- DHCP clients are also listeners on port 68 since DHCP sends all the information via broadcast instead of sending it via unicast directly to the client.

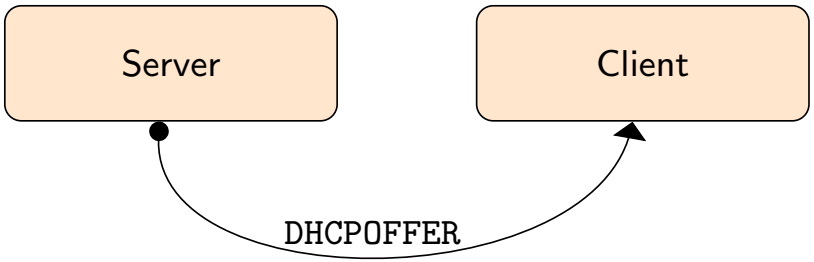
IFIN-HH

Discovery



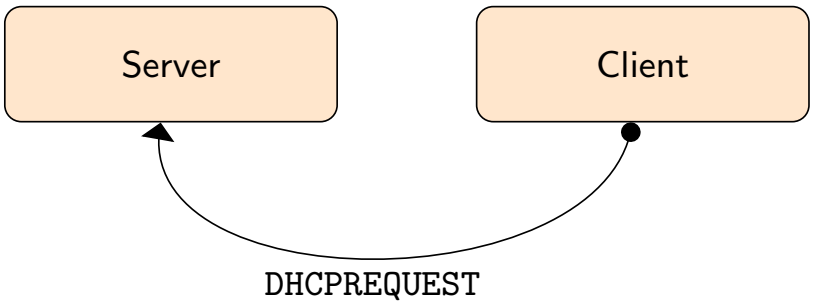
- When a client is first hooked-up to the network, the client issues (for example, by the invocation of the `dhclient` command) a `DHCPDISCOVER` message tot the **Dynamic Host Configuration Protocol** (`DHCP`) server in order to find out what configuration will be made available to the client.

Offers



- After a `DHCPDISCOVER` message sent by the client, the `DHCP` server answers with a `DHCPOFFER` message containing all the configuration information for that client.
- The client is not obliged to accept the configuration and it may reject the offer sent by the `DHCP` server and then issue another `DHCPDISOVER` message.

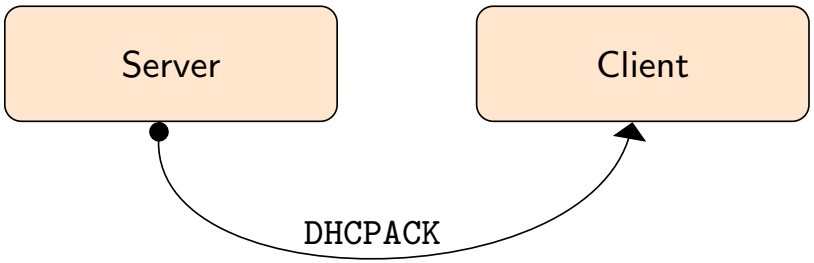
Requests



- Once an offer was made by the DHCP server to the client for a configuration, the client requests that configuration by sending a DHCPREQUEST message to the server.
- The DHCP server will then check if the request corresponds to the server configuration (for example, if no other machine has the same IP address).

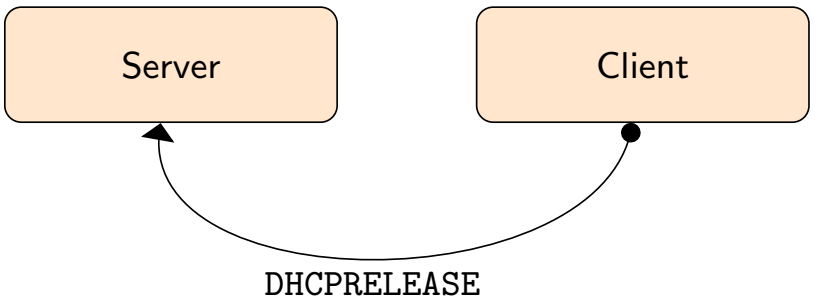
IFIN-HH

Acknowledgement



- In case the DHCP server approves of the DHCPREQUEST message, it will send a DHCPACK message to the client thereby confirming that the client is allowed to use the configuration it has requested from the server.

Release



- It is customary that when a client no longer needs an IP address that it has **leased** - for example, the client goes offline, that a `DHCPRELEASE` message is sent to the server telling it that the IP address is no longer in use and can be re-used by a different client.

General Configuration

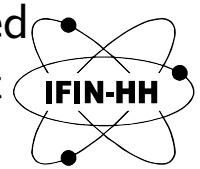
- **ISC DHCP** keeps its main configuration file usually at `/etc/dhcpd.conf` although that may vary for different Linux distributions.
- The configuration file contains general `DHCP` settings as well as **subnet declarations** or **pools** that the `DHCP` server is responsible for.
- Just like the **ISC BIND** `DNS` server, settings for the **ISC DHCP** server are contained in blocks and they apply to the immediate-enclosing block.
- There are very many options but they are all documented in the **ISC DHCP** manual page (available on Unix systems via the `man dhcpd` command).
- Some custom options can be defined in **ISC DHCP** and they are usually abused by companies such as Apple that need to pass some custom `DHCP` options from servers to clients (for example, Apple NetBoot).

Subnets

Subnet Declaration

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    option routers                  192.168.1.1;
    option subnet-mask              255.255.255.0;
    option domain-search            "nsi.dfcti";
    option domain-name-servers      192.168.1.1;
    range                           192.168.1.2 192.168.1.50;
}
```

- Each subnet declaration is given by the **subnet address** and the **netmask** for that subnet. Commonly, a `range` parameter is given within the subnet that specifies a pool of addresses that the DHCP server will be able to dish out to clients.
- Inside the subnet declaration, various configuration options can be specified that DHCP clients (such as `dhclient`) will query from the server and then apply to the interface that is brought up using DHCP.
- Since all the options such as `routers`, `subnet-mask`, etc... Are included within the subnet block, then these options will apply to all clients that receive an IP address from the given `range`.



Static IP Addresses using DHCP

Static Declaration

```
host dilbert {
    hardware ethernet 02:AC:35:7B:9F:AC;
    fixed-address 192.168.1.4;
}
```

- Once a DHCP client requests an address from the DHCP server, it will send the hardware address (MAC) along with the message (in this example, 02:AC:35:7B:9F:AC).
- Based on the hardware address, the DHCP server can declare a static IP address that will be used exclusively for that client - in this case, the address 192.168.1.4.

Scope Blocks

Blocks and Scopes

```
group {
        option domain-name-servers 192.168.1.1;

        shared-network home {
                host jackie {
                        ...
                }
                subnet 192.168.2.0 netmask 255.255.255.0 {
                        ...
                }
        }

        host dilbert {
                ...
        }
        subnet 192.168.1.0 netmask 255.255.255.0 {
                ...
        }
}
```

- **Groups** may contain **shared networks**, **static host declarations** and **subnets** for which the DHCP server is responsible for. textbfShared networks are declared with a name describing the network.
- Any **option** specified in the **parent scope** will apply to all **children of that parent**. In this example, the option `domain-name-servers` will apply to all the blocks unless the option is overridden by a new option `domain-name-servers`.

IFIN-HH

Command-Line Parameters

Running the Server

```
root# dhcpd -f -d eth0
```

- **ISC DHCPd** supports multiple command-line parameters, out of which the most useful are the `-f` (foreground) and `-d` (debug) parameters. They allow the server to be launched in the foreground with debugging on such that we can see the messages passed from clients to the server.
- Depending on the distribution. **ISC DHCPd** logs messages via the `syslog` facility and the logs can usually be found under the directory `/var/log`.
- Multiple interfaces can be specified instead of `eth0` and most distributions have a configuration file where the interfaces that **ISC DHCPd** has to listen on are specified.
- It does not really make sense to start the **ISC DHCPd** server without having the interface that it should listen on (in this case `eth0` in an `UP` state. As such, the interface `eth0` has to be brought up and configured with the corresponding parameters (IP address, netmask, broadcast, etc...) after which we can launch **ISC DHCPd** to listen on that interface.

Command-Line Parameters

Running the Client

```
root# dhclient eth0
```

- The client is usually started from startup scripts under Linux but we can always run the command manually.
- The `dhclient eth0` command will launch the **ISC DHCP** client on the network interface `eth0` and send discovery messages with the hope of finding a listening **ISC DHCPd** server.
- Once a server has been found, the `dhclient` command will fork and enter the background yielding back the control to the command line.

End Notes

A good reference for this seminar series, as well as a recommended book is Andrew Tannenbaum's "Modern Operating Systems". Andrew Tannenbaum created Minix (a free UNIX alternative) and Linus Torvalds was his student:

- Modern Operating Systems, Tanenbaum, A.S. and Bos, H., ISBN 9780133591620, LCCN 2013444331, 2014, Prentice Hall

The slides will be posted at:

- Personal Website

IFIN-HH