Network

Network tools (Debug common problems), DNS and DHCP issues, traffic filtering.

Kravets Dmitriy

Senior Systems Engineer at EPAM

Years in EPAM: 4

City: Moscow

Key skills: DevOps.IaC

COMMON NETWORK ISSUES AND DEBUGGING TOOLS

A list of network debugging tools in Linux

Sometimes there are going to an issue with network in Linux and you need to know how to troubleshoot it. We are going to discuss several network debugging tools which are commonly used in Linux.

- ***Ip addr***
- ***ping***
- ***route***
- ***netstat (ss)***
- ***telnet***
- ***tcpdump***

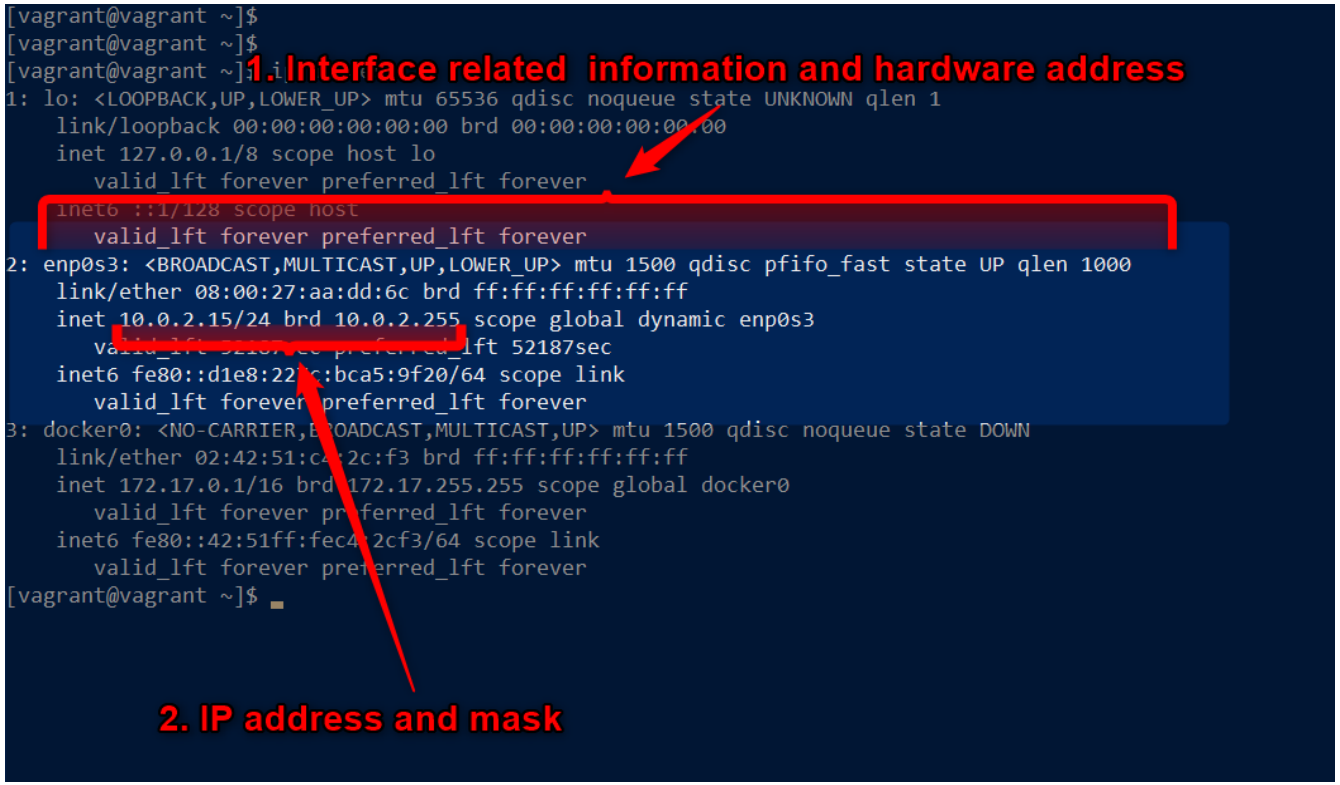
Linux network diagnostic

"ip address" command.

The ip address command displays addresses and their properties, adds new addresses and deletes old ones.

Information you can obtain:

- Interfaces
- MAC addresses
- IPv4 and IPv6 addresses and masks



Linux network diagnostic

"ip route" command.

routing table management

ip route add

add new route

ip route change

change route

ip route replace

change or add new one

```
[root@vagrant ~]#
[root@vagrant ~]# ip route
default via 10.0.2.2 dev enp0s3 proto static metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]# ip route add 10.0.3.0/24 dev enp0s3 metric 2000
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]# ip route
default via 10.0.2.2 dev enp0s3 proto static metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
10.0.3.0/24 dev enp0s3 scope link metric 2000
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]# ip route delete 10.0.3.0/24
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]# ip route show
default via 10.0.2.2 dev enp0s3 proto static metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
[root@vagrant ~]#
```

route table entries

"send ip packets to 10.0.3.0/24 via interface enp0s3. Route will have metric equal to 2000"

delete the route entry

Linux network diagnostic

“ping” command.

ping is a command-line network utility that allows you to test the IP-level connectivity of a given host on the network.

Some examples of ping command options:

#ping -n -c 5 -s 100 -i 2 -t 255 ya.ru

```
[root@vagrant ~]#
[root@vagrant ~]#
[root@vagrant ~]# ping 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_seq=1 ttl=107 time=19.3 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=107 time=19.3 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=107 time=17.5 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=107 time=17.2 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=107 time=19.0 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4011ms
rtt min/avg/max/mdev = 17.287/18.507/19.356/0.910 ms
[root@vagrant ~]#
```

ICMP packets sent by "ping" utility

Ping statistics

Linux network diagnostic

"ss" or "netstat" utilities.

ss is used to dump socket statistics. It allows showing information similar to netstat. It can play more TCP and state information than other tools.

```
[root@vagrant ~]# ss -tulnp
Netid State      Recv-Q Send-Q                                    Local Address:Port                                    Peer Address:Port
udp   UNCONN     0      0                                                 *:68                                                 *:*
 users:(("dhclient",pid=800,fd=6))
udp   UNCONN     0      0                                        127.0.0.1:323                                                 *:*
 users:(("chronyd",pid=581,fd=1))
udp   UNCONN     0      0                                              ::1:323                                                :::*
 users:(("chronyd",pid=581,fd=2))
tcp   LISTEN     0      128                                               *:22                                                 *:*
 users:(("sshd",pid=993,fd=3))
tcp   LISTEN     0      100                                      127.0.0.1:25                                                 *:*
 users:(("master",pid=1332,fd=13))
tcp   LISTEN     0      128                                              :::22                                                :::*
 users:(("sshd",pid=993,fd=4))
tcp   LISTEN     0      100                                              ::1:25                                               :::*
 users:(("master",pid=1332,fd=14))
[root@vagrant ~]# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      993/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1332/master
tcp6       0      0 :::22                   :::*                    LISTEN      993/sshd
tcp6       0      0 ::1:25                  :::*                    LISTEN      1332/master
udp        0      0 0.0.0.0:68              0.0.0.0:*                           800/dhclient
udp        0      0 127.0.0.1:323           0.0.0.0:*                           581/chronyd
udp6       0      0 ::1:323                 :::*                                581/chronyd
[root@vagrant ~]#
```

ss/netstat utilites allow us to see which linux process communicate via tcp/udp ports.

Linux network diagnostic

Telnet (teletype network protocol) uyility.

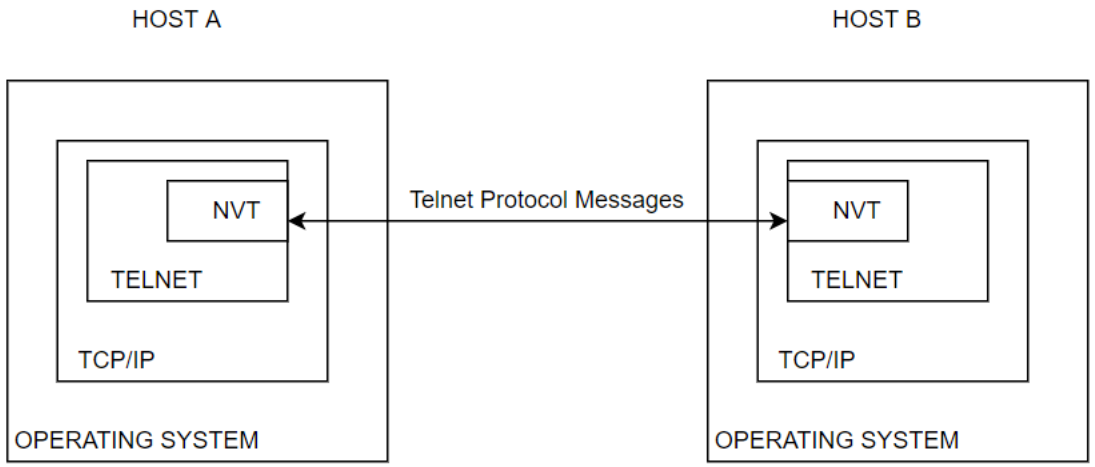
Usually, we use telnet to check if a remote host has a target port opened.

$ telnet www.example.com 80
GET /path/to/file.html HTTP/1.1
Host: www.example.com
Connection: close

```
[root@vagrant ~]# telnet www.google.com 80
Trying 74.125.130.103...
Connected to www.google.com.
Escape character is '^]'.
GET / HTTP/1.1

HTTP/1.1 200 OK
Date: Mon, 30 Aug 2021 13:26:42 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
```

Example of "telnetting" to a google's web server via TCP port 80.
Received a 200 OK response.



Linux network diagnostic

nmap utility

Official site: https://nmap.org

Very powerful tool for security scans.

- List port scanning techniques
- Host range for scanning
- Port range for scanning
- Different protocols
- Service and version detection
- OS detection
- Nmap script engine (using LUA)
- Timing and performance
- Output formatting

```
[root@vagrant ~]# nmap -A -T4 scanme.nmap.org

Starting Nmap 6.40 ( http://nmap.org ) at 2021-08-30 14:43 UTC
Warning: 45.33.32.156 giving up on port because retransmission cap hit (6).
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.032s latency).
Not shown: 995 closed ports
PORT      STATE    SERVICE    VERSION
22/tcp    open     ssh        (protocol 2.0)
| ssh-hostkey: 1024 ac:00:a0:1a:82:ff:cc:55:99:dc:67:2b:34:97:6b:75 (DSA)
| 2048 20:3d:2d:44:62:2a:b0:5a:9d:b5:b3:05:14:c2:a6:b2 (RSA)
|_256 96:02:bb:5e:57:54:1c:4e:45:2f:56:4c:4a:24:b2:57 (ECDSA)
53/tcp    filtered domain
80/tcp    open     http       Apache httpd 2.4.7 ((Ubuntu))
|_http-title: Go ahead and ScanMe!
9929/tcp  open     nping-echo Nping echo
31337/tcp open     tcpwrapped
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint
Device type: general purpose|VoIP phone
Running (JUST GUESSING): QEMU (91%), Cisco embedded (85%)
OS CPE: cpe:/o:qemu:qemu cpe:/h:cisco:unified_ip_phone_7912
Aggressive OS guesses: QEMU user mode network gateway (91%), Cisco IP Phone 7912-series (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops

TRACEROUTE (using port 80/tcp)
HOP RTT     ADDRESS
1   0.52 ms 10.0.2.2
2   0.27 ms scanme.nmap.org (45.33.32.156)

OS and Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 131.99 seconds
[root@vagrant ~]#
```

SSH related information

service detection

OS detection

WINDOWS NETWORK DIAGNOSTIC TOOLS

Windows Network Utils

Utils for network configuration and troubleshooting:

- hostname
- ipconfig
- arp
- ping
- tracert
- route
- Netstat
- Test-NetConnection

Windows Network Utils

Test-NetConnection - Displays diagnostic information for a connection. It supports ping test, TCP test, route tracing, and route selection diagnostics. Good in scripts.

Windows PowerShell

```
PS C:\>
PS C:\>
PS C:\> Test-NetConnection -ComputerName google.com


ComputerName            : google.com
RemoteAddress           : 142.250.181.238
InterfaceAlias          : Wi-Fi
SourceAddress           : 192.168.43.128
PingSucceeded           : True
PingReplyDetails (RTT) : 78 ms



PS C:\> if ((Test-Connection -ComputerName google.com).PingSucceeded) {Write-Host "Internet access is OK"}
Internet access is OK
PS C:\>
PS C:\>
PS C:\> Test-NetConnection -ComputerName google.com -CommonTCPPort HTTP


ComputerName     : google.com
RemoteAddress    : 172.217.16.142
RemotePort       : 80
InterfaceAlias   : Wi-Fi
SourceAddress    : 192.168.43.128
TcpTestSucceeded : True
```

DNS ISSUES AND TROUBLESHOOTING TOOLS

Linux network diagnostic. DNS issues.

Common DNS issues:

- Time to live (TTL):

An expiration time set to DNS record. This DNS mechanism allows end users to get relevant DNS information.

- manually set DNS records

Sometimes DNS resolution is maintained manually with errors.

- common network connectivity issues causing latency

Latency sometimes causes DNS issues causing delay for domain name resolution process.

- DDOS attacks

Sometimes DNS servers experience problems caused by DDOS attack when DNS queries overwhelm DNS servers.

Utilities to troubleshoot DNS issues

nslookup utility

dig utility

Linux network diagnostic. DNS issues.

Utilities to troubleshoot DNS issues

nslookup utility

nslookup [-option] [name | -] [server]

EXAMPLES:

$nslookup -type=ns example.com

$ nslookup -type=soa example.com

- MX reocord lookup

$ nslookup -query=mx example.com

- lookup using a specific server

$ nslookup example.com ns1.nsexample.com

- reverse lookup

$ nslookup 10.20.30.40

- lookup with timeout

$ nslookup -timeout=20 example.com

- txt record type lookup

$ nslookup -type=txt example.com

```
[vagrant@vagrant ~]$ nslookup -query=soa epam.com
Server:         10.0.2.3
Address:        10.0.2.3#53

Non-authoritative answer:
epam.com
        origin = a1-195.akam.net
        mail addr = helpdesk.epam.com
        serial = 2021090500
        refresh = 900
        retry = 600
        expire = 864000
        minimum = 3600

Authoritative answers can be found from:
epam.com        nameserver = a1-195.akam.net.
epam.com        nameserver = a20-67.akam.net.
epam.com        nameserver = a7-64.akam.net.
epam.com        nameserver = a10-64.akam.net.
epam.com        nameserver = a11-65.akam.net.
epam.com        nameserver = a14-66.akam.net.
a10-64.akam.net internet address = 96.7.50.64
a11-65.akam.net internet address = 84.53.139.65
a7-64.akam.net  internet address = 23.61.199.64
a1-195.akam.net internet address = 193.108.91.195
a14-66.akam.net internet address = 184.26.161.66
a20-67.akam.net internet address = 95.100.175.67
a1-195.akam.net has AAAA address 2600:1401:2::c3
a20-67.akam.net has AAAA address 2a02:26f0:67::43
```

query type

configured DNS server

Linux network diagnostic. DNS issues.

Utilities to troubleshoot DNS issues

dig utility

dig [@server] [**-b** *address*] [**-c** *class*] [**-f** *filename*] [**-k** *filename*] [**-m**] [**-p** *port#*] [**-q** *name*] [**-t** *type*] [**-x** *addr*] [**-y** *[hmac:]name:key*] [**-4**] [**-6**] [name] [type] [class] [queryopt...]

EXAMPLE with options:

$ dig @a20-67.akam.net. vacation.epam.com -t CNAME +short

```
[vagrant@vagrant ~]$ dig epam.com

; <<>> DiG 9.11.4-P2-RedHat-9.11.4-26.P2.el7_9.5 <<>> epam.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44933
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;epam.com.                      IN      A

;; ANSWER SECTION:
epam.com.               356     IN      A       3.214.134.159

;; Query time: 17 msec
;; SERVER: 10.0.2.3#53(10.0.2.3)
;; WHEN: Sun Sep 05 20:05:47 UTC 2021
;; MSG SIZE  rcvd: 61
```

dns query header and description

question section

answer section

configured DNS server

DHCP ISSUES AND TROUBLESHOOTING TOOLS

Linux network diagnostic. DHCP issues.

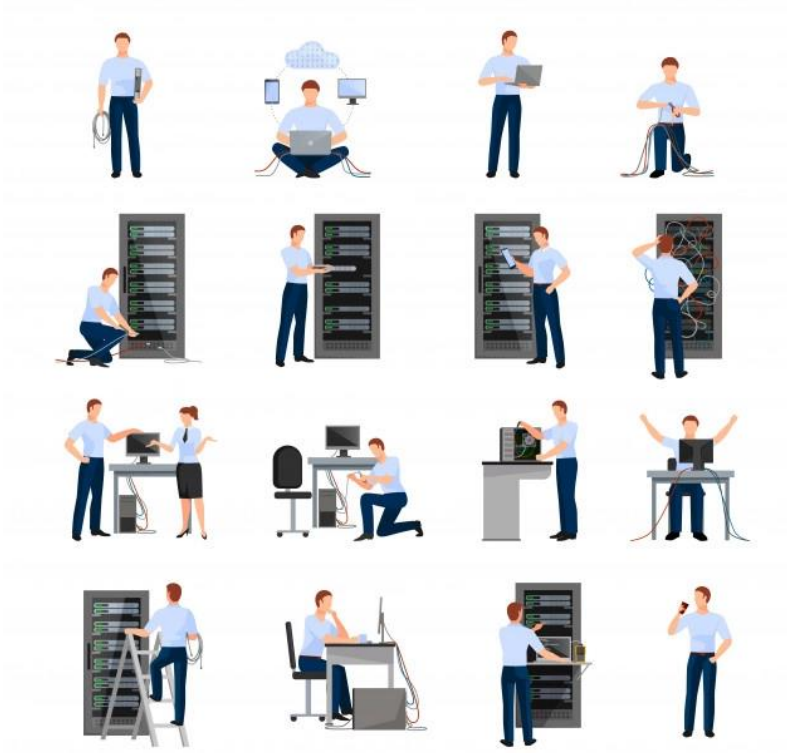
Common problems with getting address via DHCP:
- a host is not getting an IP address or other DHCP options
- a host is getting a wrong IP address or other DHCP options

Possible root causes:
- local network outage (Ethernet loop)
- internal DHCP server error (wrong DHCP configuration, ip address poll exhausted)
- malicious DHCP server (some other device was configured to assign ip address in your LAN)
- firewalls or some other devices in your LAN block DHCP communication
- ip addresses conflicts (someone assigned "your" address manually)

Linux network diagnostic. DHCP issues.

Possible troubleshooting actions:

- verify physical connectivity (***ip addr*** linux command)
- test network connectivity by configuring a proper IP address manually (***ip addr*** linux command)
- force an IP address renewal via DHCP (Windows: ***ipconfig /renew,*** *Linux:* ***dhclient -r eth0****)*
- sniff traffic using special utilities (**tcpdump**, **wireshark** utilities)
- check firewalls (**IPTABLES** utility)
- look through possible logs (Depends on your OS default logging)
- reach networking team to check network devices like routers, switches, firewalls



Linux network diagnostic. DHCP and DNS issues.

Capturing network traffic.

From Wikipedia:

A **packet analyzer** or **packet sniffer** is a computer program or computer hardware such as a packet capture appliance, that can intercept and log traffic that passes over a computer network or part of a network.[1] **Packet capture** is the process of intercepting and logging traffic.

Tcpdump utility for capturing traffic on linux machines.

Example:

$tcpdump -i enp0s3 port 53 or port 443 -e -n –w '/tmp/dns.pcap'

Command meaning:

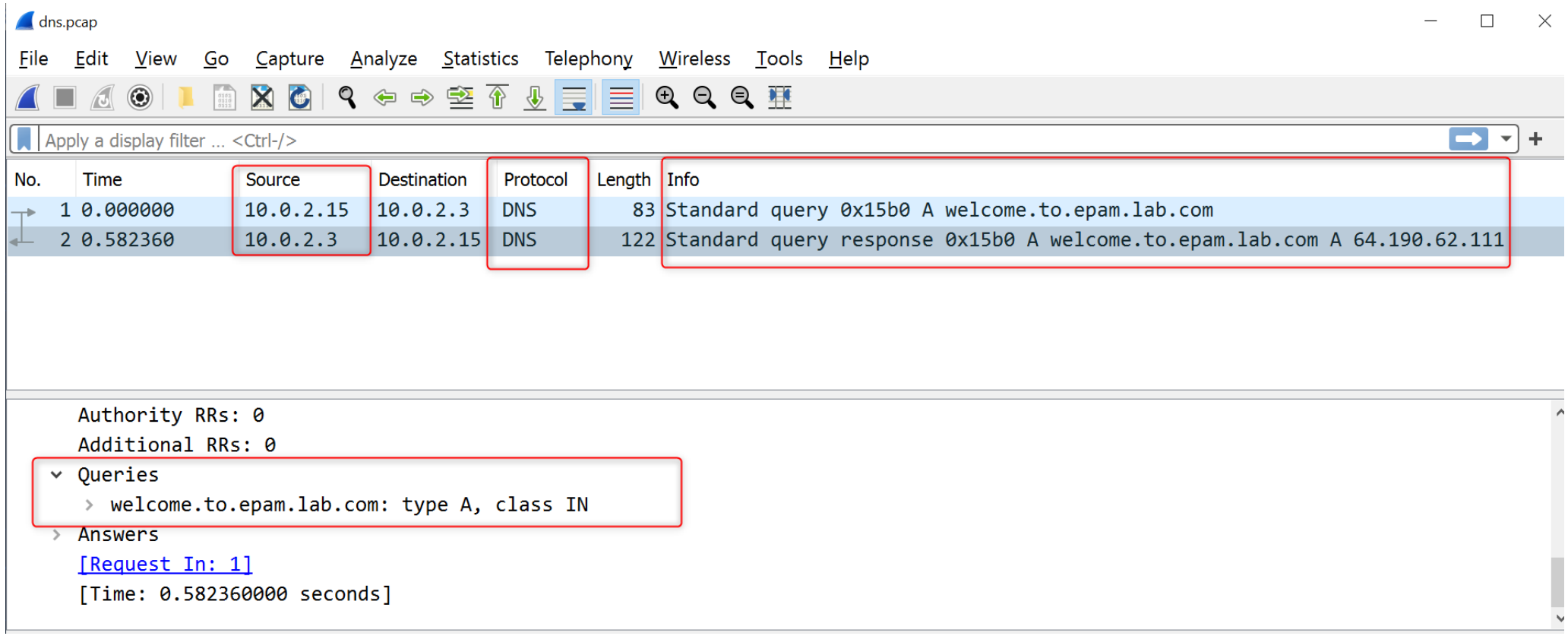
"- Capture network packets sent to a port 53 or a port 443
- On the network interface enp0s3.
- Don't convert addresses.
- Print mac-addresses info
- store the output to a file /tmp/dns.pcap"

Linux network diagnostic. DHCP and DNS issues.

Capturing network traffic.

Wireshark utility for capturing traffic on linux machines.

Graphically can represent captured traffic



IPTABLES

Linux iptables.

iptables is a user-space utility program that allows a system administrator to configure the IP packet filter rules of the Linux kernel firewall, implemented as different Netfilter modules. (from Wikipedia)

There are also some tools like **ebtables**, **ipv6tables** which are based on non ipv4 packets.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv4 Packet Format from RFC 791

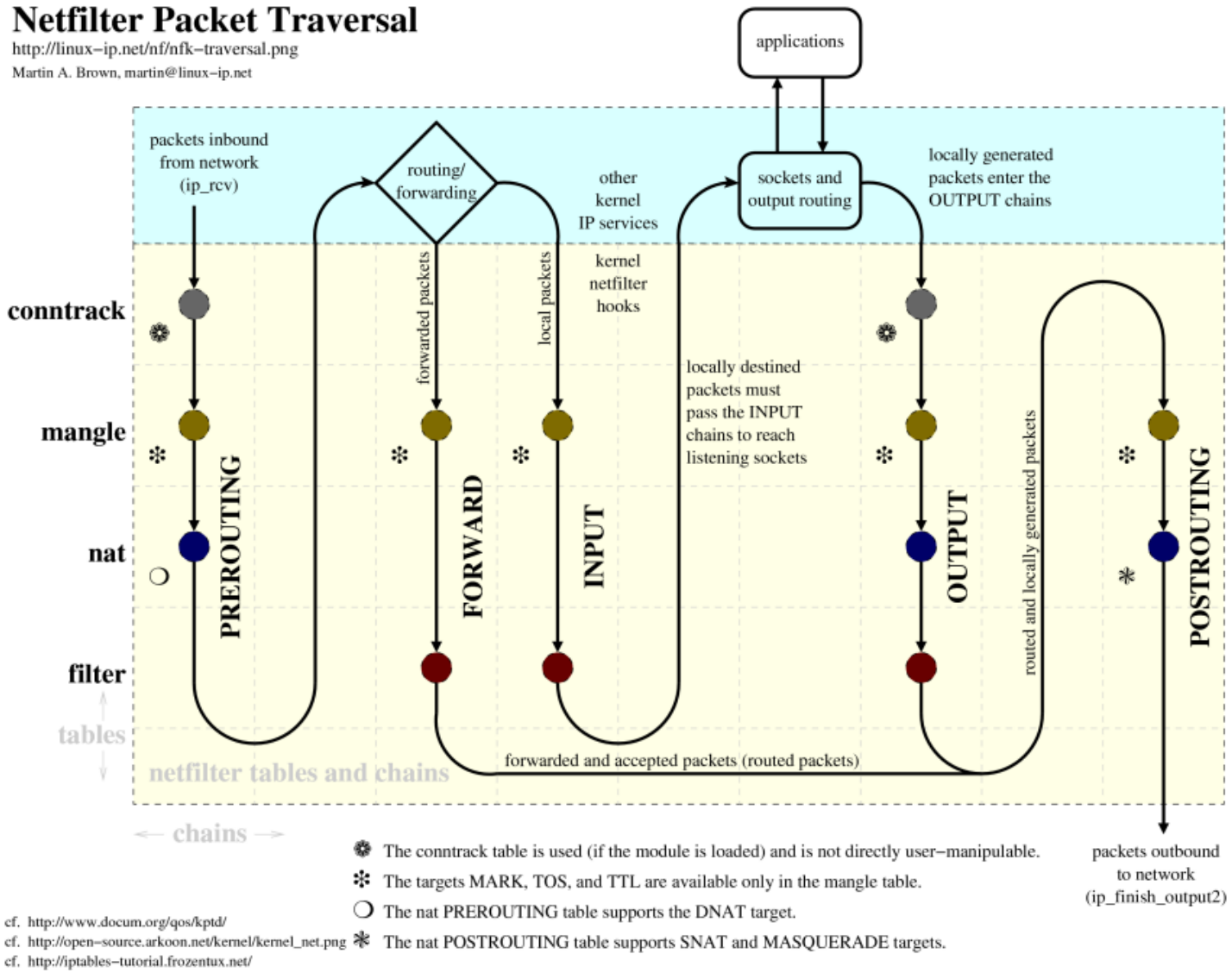
Linux iptables.

iptables is a user-space utility program that allows a system administrator to configure the IP packet filter rules of the Linux kernel firewall, implemented as different Netfilter modules. (from Wikipedia)

Iptables fundamentals

Tables	Chains	Rules
mangle nat filter raw security	PREROUTING POSTROUTING INPUT FORWARD OUTPUT	Matches Targets

Linux iptables.



Linux iptables.

Three common cases for ip packet traversing.

- Incoming packets destined for the local system: PREROUTING -> INPUT
- Incoming packets destined to another host: PREROUTING -> FORWARD -> POSTROUTING
- Locally generated packets: OUTPUT -> POSTROUTING

Iptables rules.

Rule order is a

- Matching

The matching portion of a rule specifies the criteria that a packet must meet in order for the associated action (or “target”) to be executed.

- Targets

A target is the action that are triggered when a packet meets the matching criteria of a rule. Targets are generally divided into two categories: terminating and non-terminqting targets.

Linux iptables.

Iptables tips and tricks.

- migrate from firewalld to iptables
There are some other modern tools for managing your firewall.

- backup your iptables
/sbin/iptables-save > /root/works-iptables-`date +%F`. To restore it please use /sbin/iptables-restore tool

- setup the default policy as DROP

- put most specific rules at the top of your iptables configuration. Order matters! Think twice before moving your rules!!!

- you can LOG and make reports based on your iptables rules. You also can grab a statistic of hitting your rules

- understand all your rules. Do not store unnecessary rules

Linux iptables.

Useful iptables examples.

- Set default chain policies to DROP:
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP

- **allow SSH connection**
iptables -A INPUT -i eth0 -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT

- **allow incoming HTTP and HTTPS connections**
iptables -A INPUT -i eth0 -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT

- **port forwarding**
iptables -t nat -A PREROUTING -p tcp -d 10.10.10.5 --dport 422 -j DNAT --to 10.10.10.5:22

Linux networking.

Home task.

1. Let's assume you set up a new VM in the private network. This network has access to the internet through a NAT. Your company has a set of publicly available corporate services behind the firewall:
   - Hashicorp Vault credential storage
   - nginx web server
   - mail server
   - dns server

Assuming you have admin access to all the machines and a firewall. Your newly created machine should have access to the services listed above through the firewall.

Please present your solution to implement the setup described above. Please feel free to add as much details as possible: 1)assign ip addresses and draw network diagram; 2)implement firewall using iptables; 3) implement NAT using iptables; 4) add any details you think you need to add.

Definition of done: a short presentation of your solution with configs, diagrams and test cases.

2. Please implement a simple setup where you need to implement an ip router using a linux machine: host1 <-> router <-> host2.

Definition of done: a short presentation of your solution with configs, diagrams and test cases.

THANK YOU