4. Networking Administration

IT5406 - Systems and Network Administration

Level III - Semester 5

Overview

- In this topic we discuss about the essential topics related to
  - Linux networking,
  - the domain name system,
  - single sign on, and
  - web hosting.

Intended Learning Outcomes

- At the end of this lesson, you will be able to;
  - Configure and troubleshoot networks using Linux
  - Configure and deploy a domain name system
  - Explain Single Sign on and its implementations
  - Configure and implement web hosting

List of sub topics

4.1. Networking [Ref 1: Pg. (417-458)]

4.2 The Domain Name System

4.3 Single Sign-On

4.4 Web Hosting

4.1. Networking

4.1.1. Linux Networking

- There are several methods that you can use to configure networks in Linux systems.
  - Using commands (ip or ifconfig)
  - Using NetworkManager
  - Using Configurations files
- After changing the configurations, it is recommended to reboot the system or restart the network service or turn off *(ifdown)* the interface and turn on *(*ifup*)* the interface to apply the settings.

4.1. Networking

4.1.1. Linux Networking

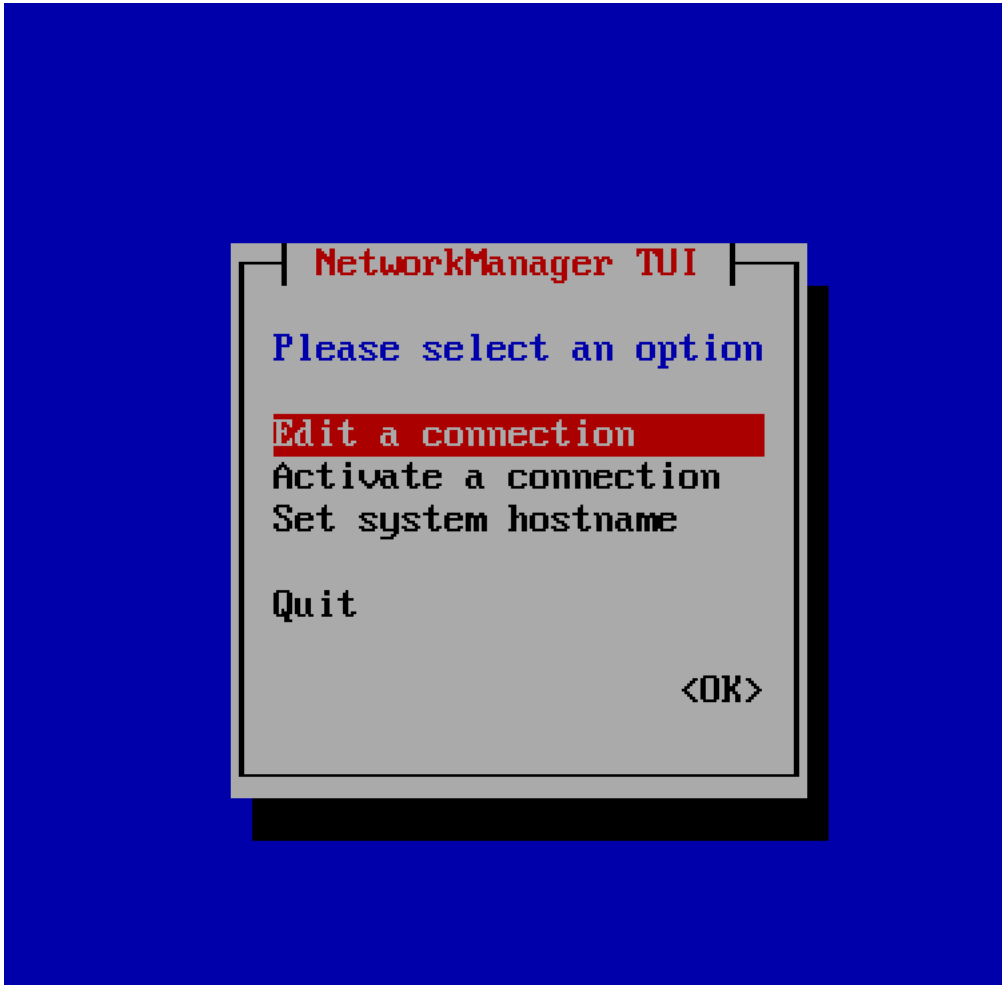
- NetworkManager
  - Available with a GUI or CLI.
  - In some Linux systems NetworkManager is always running in the background, but in some Linux systems it does not run by default.
  - It continually assesses the available networks and shifts service to “preferred” networks as they become available. Wired networks are most preferred, followed by familiar wireless networks.
  - It is designed to be run and managed by users rather than system administrators.

Ref 1: Pg. (417)

4.1. Networking

4.1.1. Linux Networking

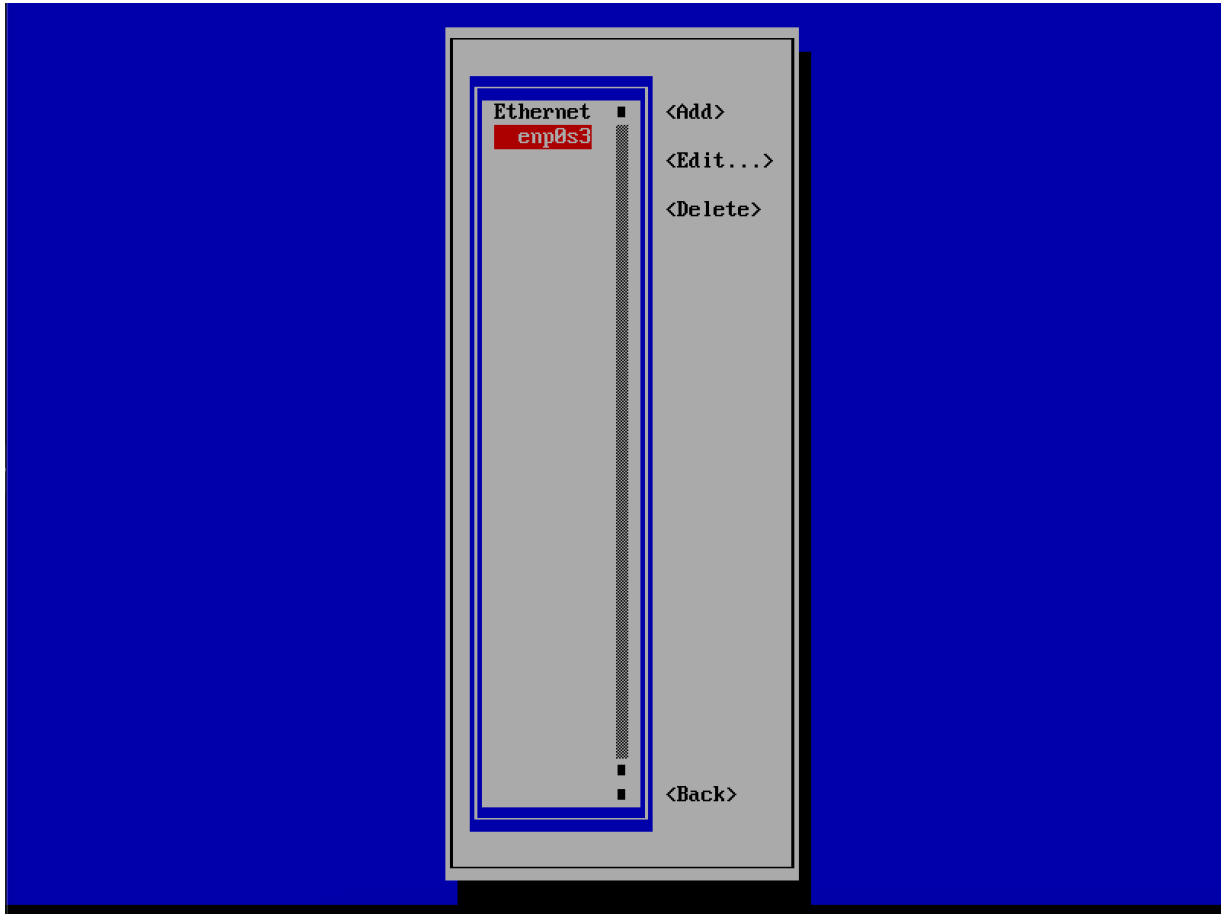
- NMTUI



4.1. Networking

4.1.1. Linux Networking

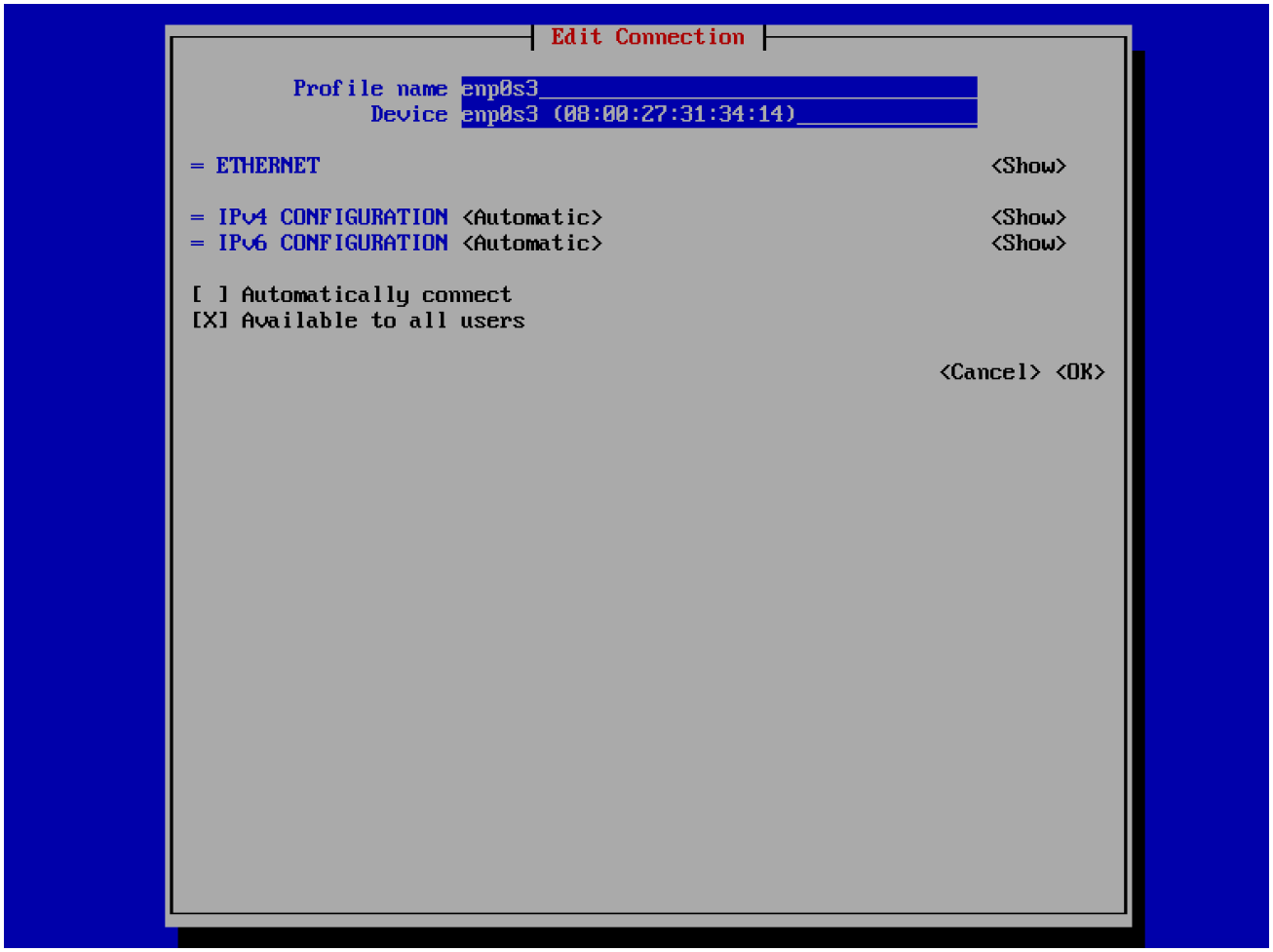
- NMTUI



4.1. Networking

4.1.1. Linux Networking

- NMTUI



4.1. Networking

4.1.1. Linux Networking

- Commands
  - You can use the following tools to configure the network in the command line interface,
    - ifconfig https://man7.org/linux/man-pages/man8/ifconfig.8.html
    - ip https://man7.org/linux/man-pages/man8/ip.8.html

Ref 1: Pg. (418)

4.1. Networking

4.1.1. Linux Networking

- Configurations files
  - Different Linux variants have different configurations files.
  - In CentOS you can find the configuration files related to networks in /etc/sysconfig/network-scripts/ directory.

```
[tharindu@localhost network-scripts]$ ls
ifcfg-enp0s3   ifdown-ipv6    ifdown-Team      ifup-eth    ifup-post      ifup-tunnel
ifcfg-lo       ifdown-isdn    ifdown-TeamPort  ifup-ippp   ifup-ppp       ifup-wireless
ifdown         ifdown-post    ifdown-tunnel    ifup-ipv6   ifup-routes    init.ipv6-global
ifdown-bnep    ifdown-ppp     ifup             ifup-isdn   ifup-sit       network-functions
ifdown-eth     ifdown-routes  ifup-aliases     ifup-plip   ifup-Team      network-functions-ipv6
ifdown-ippp    ifdown-sit     ifup-bnep        ifup-plusb  ifup-TeamPort
[tharindu@localhost network-scripts]$ _
```

  - Open the relevant file for the interface you want to configure. (in this machine the name of the network interface is “enp0s3”. Therefore the relavent file is “ifcfg-enp0s3”.

4.1. Networking

4.1.1. Linux Networking

- Configurations files…(cont)
  - Open ifcfg-enp0s3 using an editor.

```
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=dhcp
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=enp0s3
UUID=a0c7f2fb-db3f-4c30-b713-b1db816e8276
DEVICE=enp0s3
ONBOOT=no
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"ifcfg-enp0s3" [readonly] 15L, 281C
```

4.1. Networking

4.1.1. Linux Networking

- Configurations files…(cont)
  - Update the configuration file.

```
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=static
DEFROUTE=yes
IPADDR=192.168.10.100
NETMASK=255.255.255.0
GATEWAY=192.168.10.255
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=enp0s3
DEVICE=enp0s3
ONBOOT=yes
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"ifcfg-enp0s3" 17L, 309C
```

4.1. Networking

4.1.1. Linux Networking

- Configurations files…(cont)
  - Restart the network service: systemctl restart network
  - Verify by using *ip a* command (or using "ifconfig" command)

```
[root@localhost network-scripts]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
    link/ether 08:00:27:31:34:14 brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.100/24 brd 192.168.10.255 scope global enp0s3
       valid_lft forever preferred_lft forever
[root@localhost network-scripts]#
```

  - If it does not work, turn off the NetworkManager.

4.1. Networking

4.1.1. Linux Networking

- Ethtool - Use to get information related to the network adapter.

```
[tharindu@localhost ~]$ ethtool enp0s3
Settings for enp0s3:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supported pause frame use: No
        Supports auto-negotiation: Yes
        Supported FEC modes: Not reported
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Advertised pause frame use: No
        Advertised auto-negotiation: Yes
        Advertised FEC modes: Not reported
        Speed: 1000Mb/s
        Duplex: Full
        Port: Twisted Pair
        PHYAD: 0
        Transceiver: internal
        Auto-negotiation: on
        MDI-X: off (auto)
Cannot get wake-on-lan settings: Operation not permitted
        Current message level: 0x00000007 (7)
                               drv probe link
        Link detected: yes
```

Ref 1: Pg. (421-422)

4.1. Networking

4.1.2. Network Troubleshooting

- There are several tools we can use to troubleshoot issues in computer networks in Linux systems. Following are some of the basic tools,
  - Ping
  - Traceroute
  - Tcpdump
  - Netstat

Ref 1: Pg. (428)

4.1. Networking

4.1.2. Network Troubleshooting

- Ping
  - Use to check whether a host is alive.

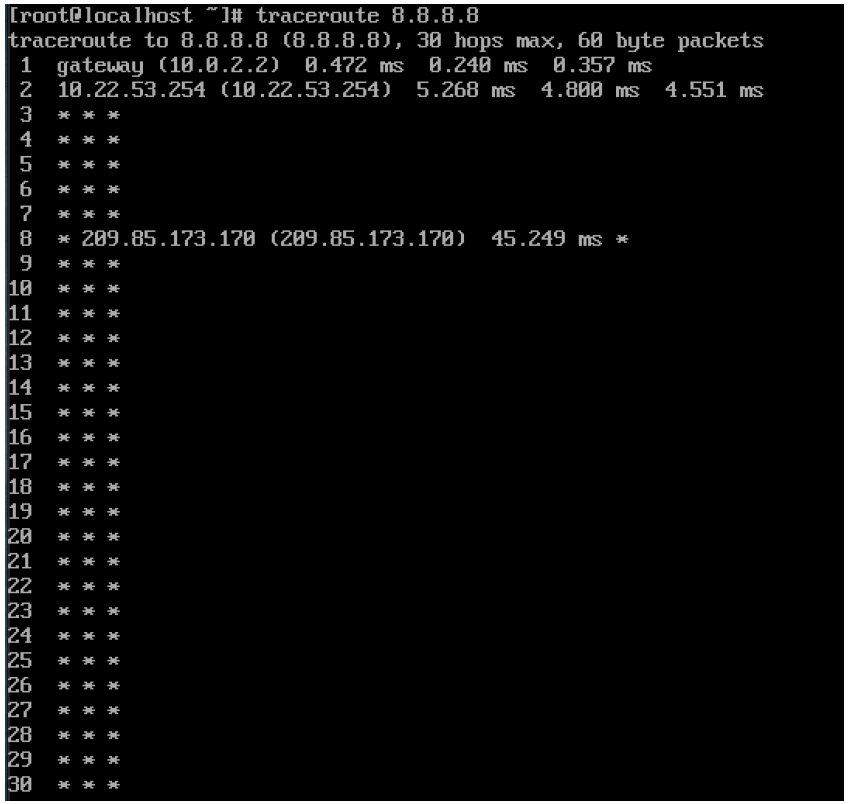
```
[root@localhost ~]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=63 time=51.8 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=63 time=51.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=63 time=52.2 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=63 time=49.8 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=63 time=45.4 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=63 time=48.7 ms
^C
--- 8.8.8.8 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5010ms
rtt min/avg/max/mdev = 45.497/49.987/52.250/2.365 ms
```

Ref 1: Pg. (429-431)

4.1. Networking

4.1.2. Network Troubleshooting

- Traceroute
  - Can use to find the gateways between the hosts.



If the tool is not available install using ***yum install traceroute*** command.

Ref 1: Pg. (431-434)

4.1. Networking

4.1.2. Network Troubleshooting

- tcpdump
  - Use to capture the network traffic flows through network interfaces.
  - Known as packet sniffing software.
  - Wireshark (GUI) is one of the most popular packet sniffing tool with more advanced features.
  - Tshark is the CLI version of Wireshark.

Ref 1: Pg. (434-436)

4.1. Networking

4.1.3. Network Monitoring

- Network monitoring tools helps you to get an oversight of the systems and the network.
- It collect data from number of sources and provide a summary of information which helps the administrators to get some idea about the ongoing trends of the network and it alter if there are any problems.
- Some tools,
  - SmokePing
  - iPerf
  - Cacti

Ref 1: Pg. (437-439)

4.1. Networking

4.1.4. Firewalls and NAT

- Linux iptables: rules, chains, and tables
  - iptables applies ordered “chains” of rules to network packets. Sets of chains make up “tables” and are used for handling specific kinds of traffic.
  - The default iptables table is named “filter”. Chains of rules in this table are used for packet-filtering network traffic. The filter table contains three default chains: FORWARD, INPUT, and OUTPUT. Each packet handled by the kernel is passed through exactly one of these chains.

Ref 1: Pg. (440-442)

4.1. Networking

4.1.4. Firewalls and NAT

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
INPUT_direct  all  --  anywhere             anywhere
INPUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
FORWARD_direct  all  --  anywhere             anywhere
FORWARD_IN_ZONES  all  --  anywhere             anywhere
FORWARD_OUT_ZONES  all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere             ctstate INVALID
REJECT     all  --  anywhere             anywhere             reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
OUTPUT_direct  all  --  anywhere             anywhere

Chain FORWARD_IN_ZONES (1 references)
target     prot opt source               destination
FWDI_public  all  --  anywhere             anywhere             [goto]
FWDI_public  all  --  anywhere             anywhere             [goto]

Chain FORWARD_OUT_ZONES (1 references)
target     prot opt source               destination
FWDO_public  all  --  anywhere             anywhere             [goto]
FWDO_public  all  --  anywhere             anywhere             [goto]
```

Ref 1: Pg. (440-)

4.1. Networking

4.1.4. Firewalls and NAT

- *iptables* firewall setup
  - Before you can use *iptables* as a firewall, you must enable IP forwarding and make sure that various *iptables* modules have been loaded into the kernel.
  - A Linux firewall is usually implemented as a series of *iptables* commands contained in an ***rc*** startup script.
  - Individual *iptables* commands usually take one of the following forms:
    - iptables -F chain-name
    - iptables -P chain-name target
    - iptables -A chain-name -i interface -j target

Ref 1: Pg. (442-445)

4.1. Networking

4.1.4. Firewalls and NAT

- *Linux Nat*
  - *iptables* implements NAT as well as packet filtering.
  - If you use NAT to let local hosts access the Internet, you must use a full complement of firewall filters.
  - To enable NAT, first you should enable pracket forwarding by changing the kernel variable /proc/sys/net/ipv4/ip_forward to 1
  - Then add the following kernel modules
    - $ sudo modprobe iptable_nat
    - $ sudo modprobe ip_conntrack
    - $ sudo modprobe ip_conntrack_ftp
  - Use iptables command to route packets using NAT.
    - iptables -t nat -A POSTROUTING -o enp0s3 -j SNAT --to 10.15.20.5

Ref 1: Pg. (442-445)

4.1. Networking

4.1.5. Cloud Networking

- Define the networking environment in the virtual servers.
- The combination of virtualization technology and programmable network switching equipment gives platform providers great flexibility to define the networking model
- Some of the examples of cloud platforms;
  - AWS's virtual private cloud (VPC)
  - Google Cloud Platform networking
  - DigitalOcean networking

https://aws.amazon.com/vpc/

https://cloud.google.com/products/networking/

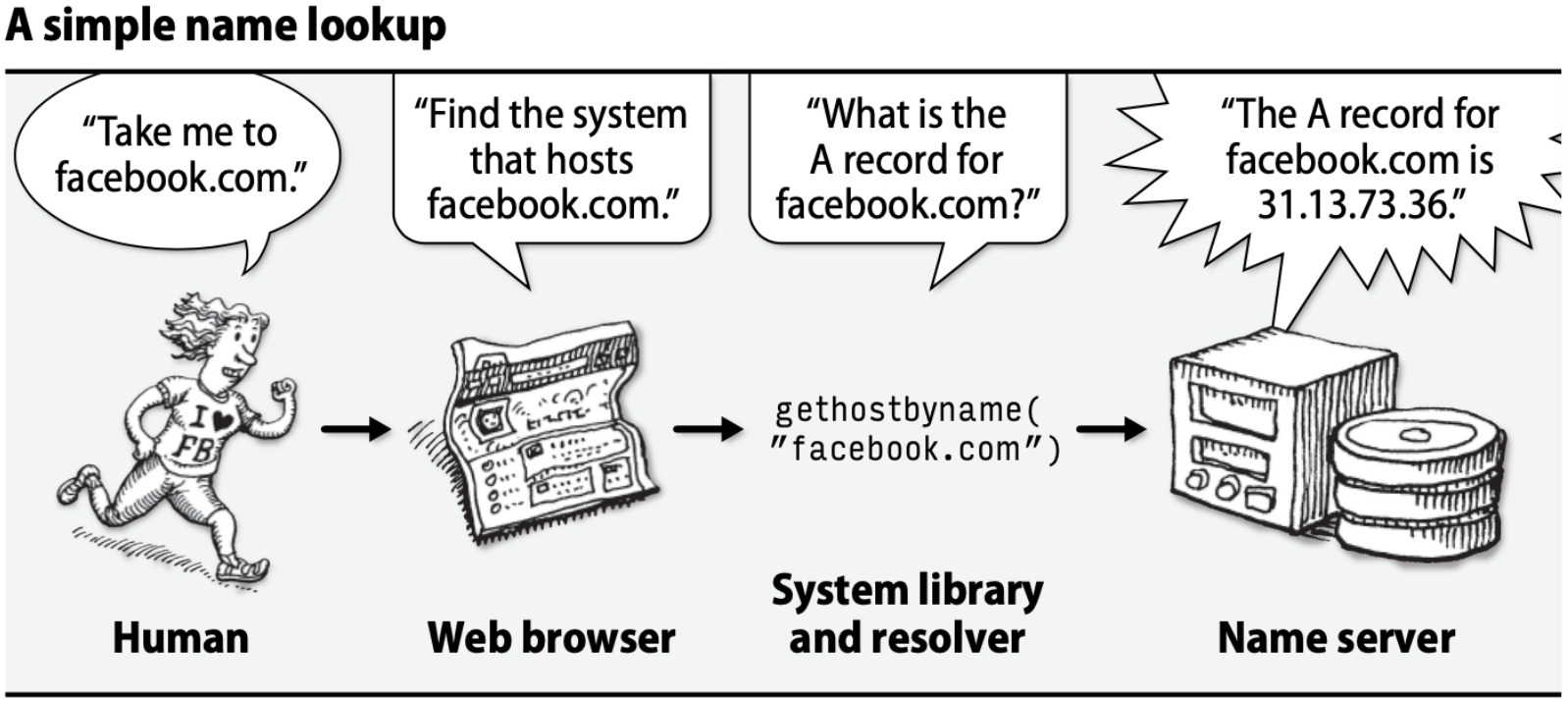
https://docs.digitalocean.com/products/networking/

Ref 1: Pg. (448-457)

4.2 The Domain Name System

4.2.1. DNS Architecture

- The need of Name Lookup
- DNS is a distributed hierarchical database
- Queries and responses



4.2 The Domain Name System

4.2.1. DNS Architecture

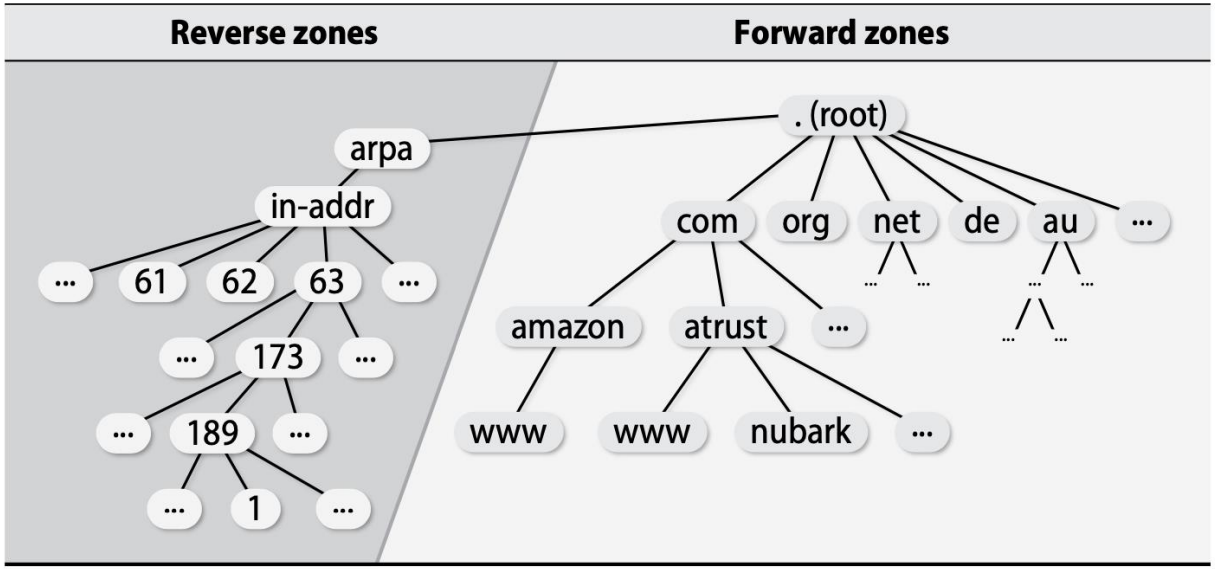
- DNS service providers
- Configuring a Linux system to resolve domain names
  - resolv.conf: client resolver configuration
  - nsswitch.conf: who do I ask for a name?
- How can we use /etc/hosts file to resolve names locally?

4.2 The Domain Name System

4.2.2. The DNS Namespace

- The DNS namespace is organized into a tree that contains both forward mappings and reverse mappings
- Forward mappings map hostnames to IP addresses (and other records) and reverse mappings map IP addresses to hostnames

DNS zone tree



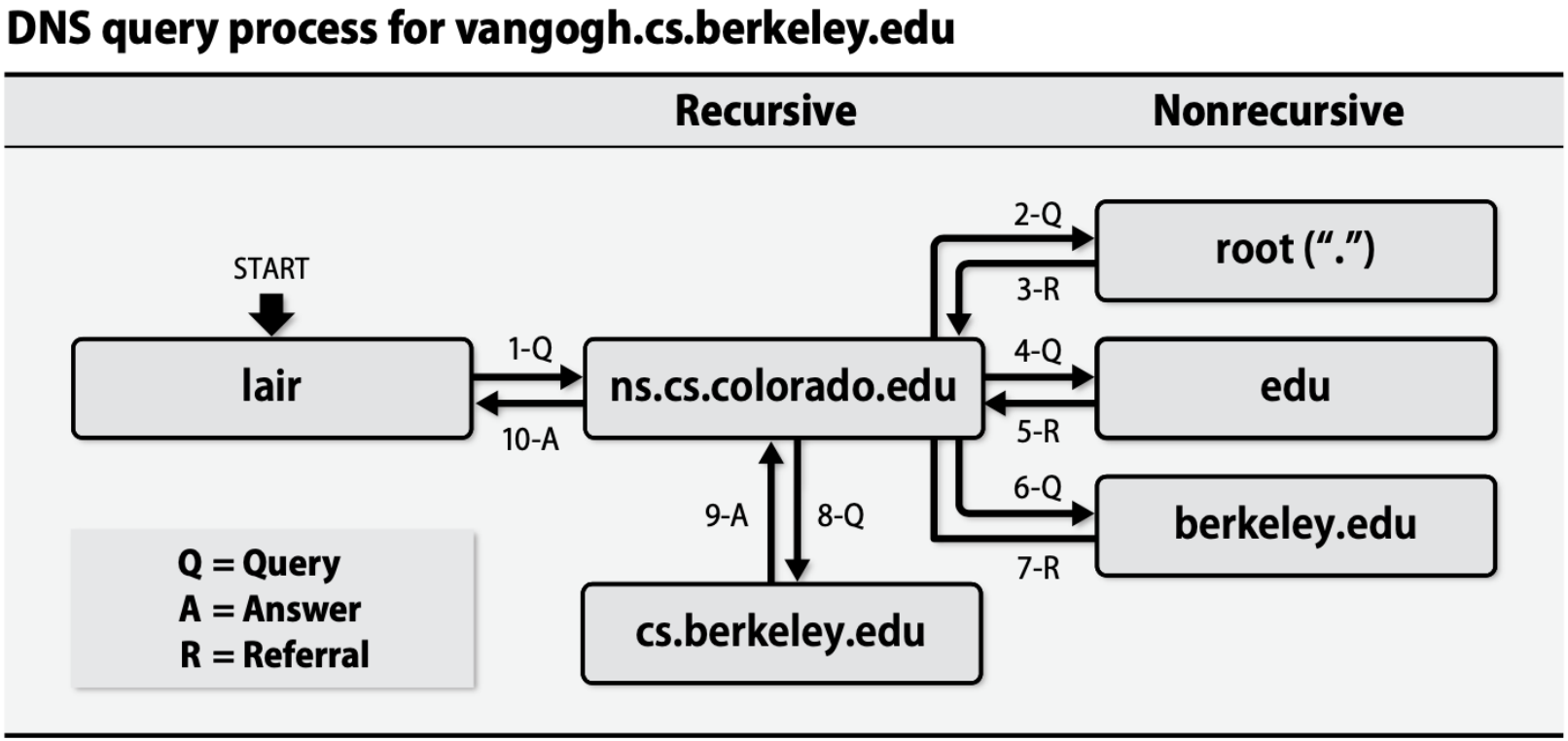
4.2 The Domain Name System

4.2.2. The DNS Namespace

- What is a fully qualified domain name?
  - www.ucsc.cmb.ac.lk.
  - Within the DNS system, the presence or absence of the trailing dot is of crucial importance as it represents the root server.
- Two types of top-level domains exist: country code domains (ccTLDs) and generic top-level domains (gTLDs)
- If you have a Domain Name Server and want to access the systems referred by that DNS globally, you have to register your domain name.
- If you have a Domain Name Server, you can create subdomains locally

4.2 The Domain Name System

4.2.3. How DNS works?



Source: Ref 01

4.2 The Domain Name System

4.2.3. How DNS works?

- **Authoritative and caching-only servers**
  - Master, slave, and caching-only servers are distinguished by two characteristics:
    - where the data comes from?
    - whether the server is authoritative for the domain?
  - Master – Slave relationship and zone transfers
- **Recursive and nonrecursive servers**
  - A client of a nonrecursive server must be prepared to accept and act on referrals
  - Any local name server listed in a client's resolv.conf file must be recursive

4.2 The Domain Name System

4.2.3. How DNS works?

- **Resource records**

```
nubark          IN      A       63.173.189.1
                IN      MX      10 mailserver.atrust.com.
```

Section 4.2.4 will have more details on this.

4.2 The Domain Name System

4.2.3. How DNS works?

- **Delegation**
  - Root servers
  - Top level domain servers
  - Authoritative domain servers
  - Delegation of authority to subdomain servers

4.2 The Domain Name System

4.2.3. How DNS works?

- **Cashing and efficiency**
  - TTL
  - The effect of negative cashing
- **Multiple answers and round robin DNS load balancing**
  - TTL and DNS load balancing

```
www             IN A      192.168.0.1
                IN A      192.168.0.2
                IN A      192.168.0.3
```

4.2 The Domain Name System

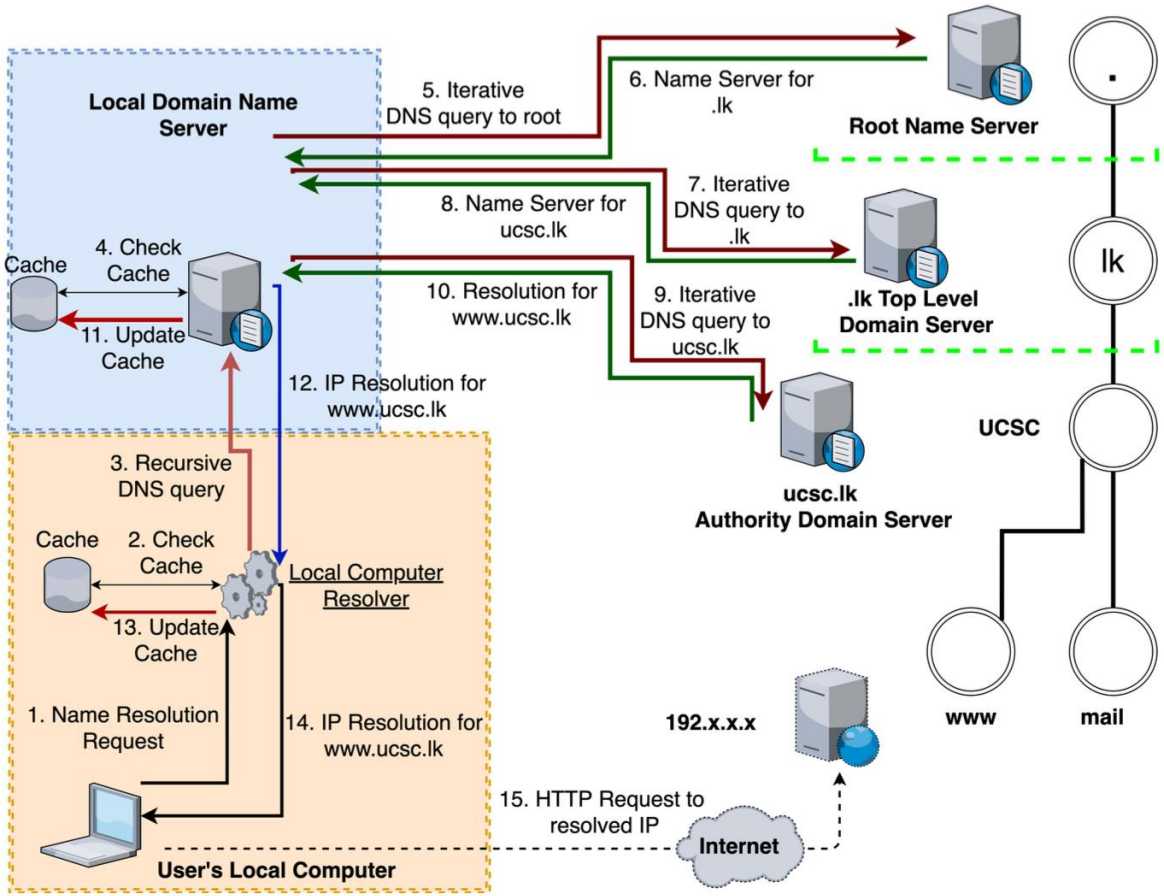
4.2.3. How DNS works?

- **Debugging with query tools**
  - Five command-line tools that query the DNS database are distributed with BIND:
    - **nslookup**
    - **dig**
    - **host**
    - **drilldelv**

Students should be able to use the tools and understand the output in detail

4.2 The Domain Name System

4.2.3. How DNS works?



4.2 The Domain Name System

4.2.4. The DNS Database

- **Parser commands in zone files**
  - $ORIGIN
  - $INCLUDE
  - $TTL

Students shall learn the usage and syntax of parser commands

4.2 The Domain Name System

4.2.4. The DNS Database

- **Resource Records (RR)**
  - Basic format of a resource record

```
[name] [ttl] [class] type data
```

Special characters in resource records

Character	Meaning
;	Introduces a comment
@	The current zone name
( )	Allows data to span lines
*	Wild card (*name* field only)[a]

Students shall learn the syntax of resource records

4.2 The Domain Name System

4.2.4. The DNS Database

- **Resource Records (RR)**

	Type	Name	Function
Zone	SOA	Start Of Authority	Defines a DNS zone
	NS	Name Server	Identifies servers, delegates subdomains
Basics	A	IPv4 Address	Name-to-address translation
	AAAA	IPv6 Address	Name-to-IPv6-address translation
	PTR	Pointer	Address-to-name translation
	MX	Mail Exchanger	Controls email routing
Security	DS	Delegation Signer	Hash of signed child zone's key-signing key
	DNSKEY	Public Key	Public key for a DNS name
	NSEC	Next Secure	Used with DNSSEC for negative answers
	NSEC3	Next Secure v3	Used with DNSSEC for negative answers
	RRSIG	Signature	Signed, authenticated resource record set
Optional	CNAME	Canonical Name	Nicknames or aliases for a host
	SRV	Service	Gives locations of a well-known service
	TXT	Text	Comments or untyped information

4.2 The Domain Name System

4.2.4. The DNS Database

- **Resource Record - Start of Authority (SOA) records**

```
; Start of authority record for atrust.com
atrust.com.      IN SOA    ns1.atrust.com. hostmaster.atrust.com. (
    2017110200     ; Serial number
    10800          ; Refresh  (3 hours)
    1200           ; Retry    (20 minutes)
    3600000        ; Expire   (40+ days)
    3600 )         ; Minimum  (1 hour)
```

4.2 The Domain Name System

4.2.4. The DNS Database

- **Resource Record**
  - SOA records
  - NS records
  - A records
  - AAAA records
  - PTR records
  - MX records
  - CNAME records
  - SRV records
  - TXT records

Please, refer the reference book for details of each resource record.

4.2 The Domain Name System

4.2.4. The DNS Database

- **Resource Record**
  - SPF records
  - DKIM records
  - DMARC records
  - DNSSEC records

Please, refer the reference book for details of each resource record.

4.2 The Domain Name System

4.2.5. BIND software configuration

- BIND is a Domain Name Server implementation for Linux systems.
- The BIND distribution has four major components
  - A name server daemon called named that answers queries
  - A resolver library that queries DNS servers on behalf of users
  - Command-line interfaces to DNS: **nslookup**, **dig**, and **host**
  - A program to remotely control named called **rndc**

4.2 The Domain Name System

4.2.5. BIND software configuration

- The `include` statement
- The `Options` statement
  - File locations
  - Name server identity
  - Zone synchronization
  - Query recursion
  - Cache memory use
  - IP port utilization
  - Forwarding
  - Permissions
  - Packet sizes
  - DNSSEC control
  - Statistics
  - Performance tuning

4.2 The Domain Name System

4.2.5. BIND software configuration

- The `acl` statement
- The `key` (TSIG) statement
- The `server` statement
- The `master` statement
- The `logging` statement
- The `statistics-channels` statement
- The `zone` statement
- The `control` statement for rndc
- The `view` statement

Configuring DNS with the configuration statements will help student to understand better.

4.2 The Domain Name System

4.2.6. Zone file updating

- Updating DNS data and DNS propagation.

- Zone transfer
  - DNS servers are synchronized through a mechanism called a zone transfer
  - zone transfers use the TCP protocol on port 53
  - Secondary server / Slave server of the authoritative server initiates the zone transfer
  - Please, refer to the reference book for details about the zone transfer steps.

4.3 Single Sign-On (SSO)

4.3.1. Core SSO Elements

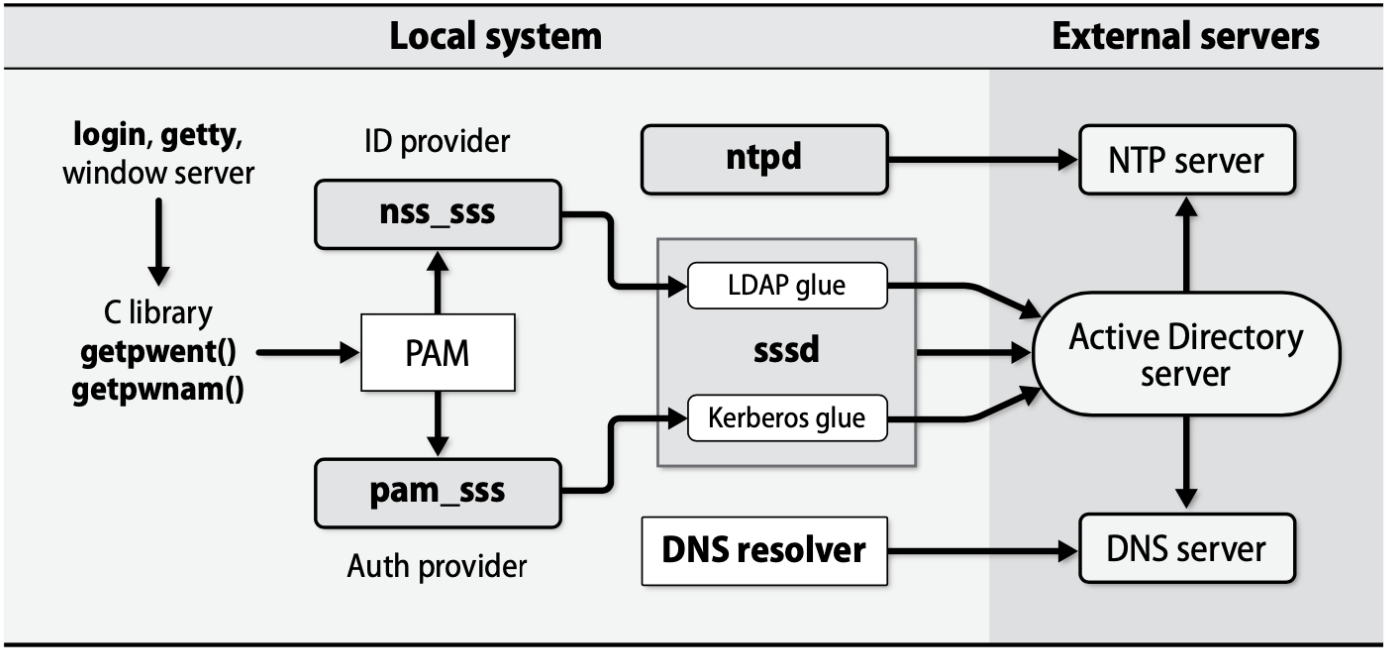
- A centralized directory store
- A tool for managing user information in the directory
- A mechanism for authenticating user identities
- Centralized-identity-and-authentication-aware versions of the library routines that look up user attributes

4.3 Single Sign-On (SSO)

4.3.1. Core SSO Elements

Exhibit A on the next page illustrates the high-level relationships of the various components in a typical configuration

SSO components



4.3 Single Sign-On (SSO)

4.3.2. Light-weight Directory Access Protocol (LDAP) Service

- Any kind of data that matches the following assumptions is a candidate for inclusion in the Directory.
  - Data objects are relatively small
  - The database will be widely replicated and cached
  - The information is attribute-based
  - Data are read often but written infrequently
  - Searching is a common operation

4.3 Single Sign-On (SSO)

4.3.2. Light-weight Directory Access Protocol (LDAP) Service

- Uses of LDAP
  - LDAP acts as a central repository for login names, passwords, and other account attributes.
  - LDAP can store additional directory information about users, such as phone numbers, home addresses, and office locations
  - Most mail systems can draw a large part of their routing information from LDAP
  - LDAP makes it easy for applications to authenticate users without having to worry about the exact details of account management
  - LDAP is well supported by common scripting languages such as Perl and Python through code libraries
  - LDAP is well supported as a public directory service

4.3 Single Sign-On (SSO)

4.3.2. Light-weight Directory Access Protocol (LDAP) Service

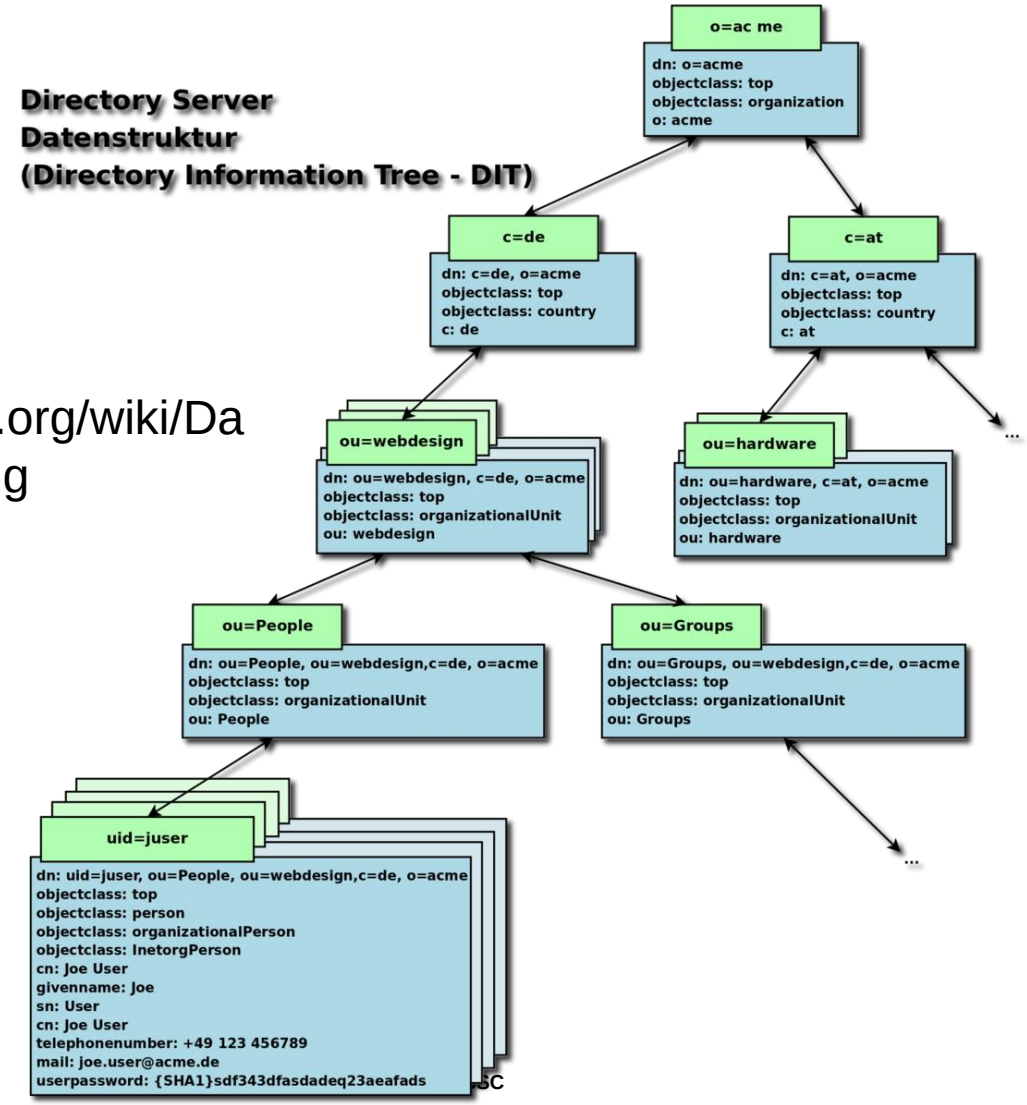
The structure of LDAP data

```
dn: uid=ghopper,ou=People,dc=navy,dc=mil
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
uid: ghopper
cn: Grace Hopper
userPassword: {crypt}$1$pZaGA2RL$MPDJoc0afuhHY6yk8HQFp0
loginShell: /bin/bash
uidNumber: 1202
gidNumber: 1202
homeDirectory: /home/ghopper
```

4.3 Single Sign-On (SSO)

4.3.2. Light-weight Directory Access Protocol (LDAP) Service

Ref:
https://de.wikipedia.org/wiki/Datei:Datenstruktur.png



4.3 Single Sign-On (SSO)

4.3.3. Using Directory Service for Login

- Implement and deploy a directory service
- Configure **sssd** and **pam** to authenticate users through LDAP

Students should implement this to get a better understanding over the topic.

4.4 Web Hosting

4.4.1. Hypertext Transfer Protocol (HTTP)

- In its simplest form, HTTP is a client/server, one-request/one-response protocol
- HTTP versions 1.0 and 1.1 are sent over the wire in plain text
- HTTP/2 moves from plain text to binary format in an effort to simplify parsing and improve network efficiency
    - Generic tools like telnet are no longer useful with HTTP/2.

4.4 Web Hosting

4.4.1. Hypertext Transfer Protocol (HTTP)

Uniform Resource Locators (URLs)

- A URL is an identifier that specifies how and where to access a resource.
- The address portion of a web URL allows quite a bit of interior structure. Here's the overall pattern:
- *scheme://[username:password@]hostname[:port][/path][?query][#anchor]*
- All the elements are optional except *scheme* and *hostname*.

4.4 Web Hosting

4.4.1. Hypertext Transfer Protocol (HTTP)

Structure of an HTTP transaction

- HTTP requests
- The first line of a request specifies an action for the server to perform. It consists of a request method (also known as the verb), a path on which to perform the action, and the HTTP version to use.
- Example: `GET /index.html HTTP/1.1`
- There are a number of HTTP request methods

Verb	Safe?	Purpose
GET	Yes	Retrieves the specified resource
HEAD	Yes	Like GET, but requests no payload; retrieves metadata only
DELETE	No	Deletes the specified resource
POST	No	Applies request data to the given resource
PUT	No	Similar to POST, but implies replacement of existing contents
OPTIONS	Yes	Shows what methods the server supports for the specified path

4.4 Web Hosting

4.4.1. Hypertext Transfer Protocol (HTTP)

Structure of an HTTP transaction

- HTTP response
- `HTTP/1.1 200 OK`
- The important part is the three-digit numeric status code. The phrase that follows it is a helpful English translation that software ignores.
- The first digit in the code determines its class

4.4 Web Hosting

4.4.1. Hypertext Transfer Protocol (HTTP)

Structure of an HTTP transaction

- HTTP response

HTTP response classes

Code	General indication	Examples
1xx	Request received; processing continues	101 Switching Protocols
2xx	Success	200 OK 201 Created
3xx	Further action needed	301 Moved Permanently 302 Found [a]
4xx	Unsatisfiable request	403 Forbidden 404 Not Found
5xx	Server or environment failure	503 Service Unavailable

4.4 Web Hosting

4.4.1. Hypertext Transfer Protocol (HTTP)

Structure of an HTTP transaction

- Headers and the message body
- Headers specify metadata about a request or response, such as whether to allow compression; what types of content are accepted, expected, or provided; and how intermediate caches should handle the data.
- Headers are separated from the message body by a blank line.
- For requests, the body can include parameters (for POST or PUT requests) or the contents of a file to upload.
- For responses, the message body is the payload of the resource being requested (e.g., HTML, image data, or query results).
- **Students are required to know the use of curl command on HTTP. Refer the main reference book.**

4.4 Web Hosting

4.4.1. Hypertext Transfer Protocol (HTTP)

More topics on HTTP

- TCP connection reuse
- HTTP over TLS
- Virtual hosts

4.4 Web Hosting

4.4.2. Web software basics

Partial list of HTTP server types

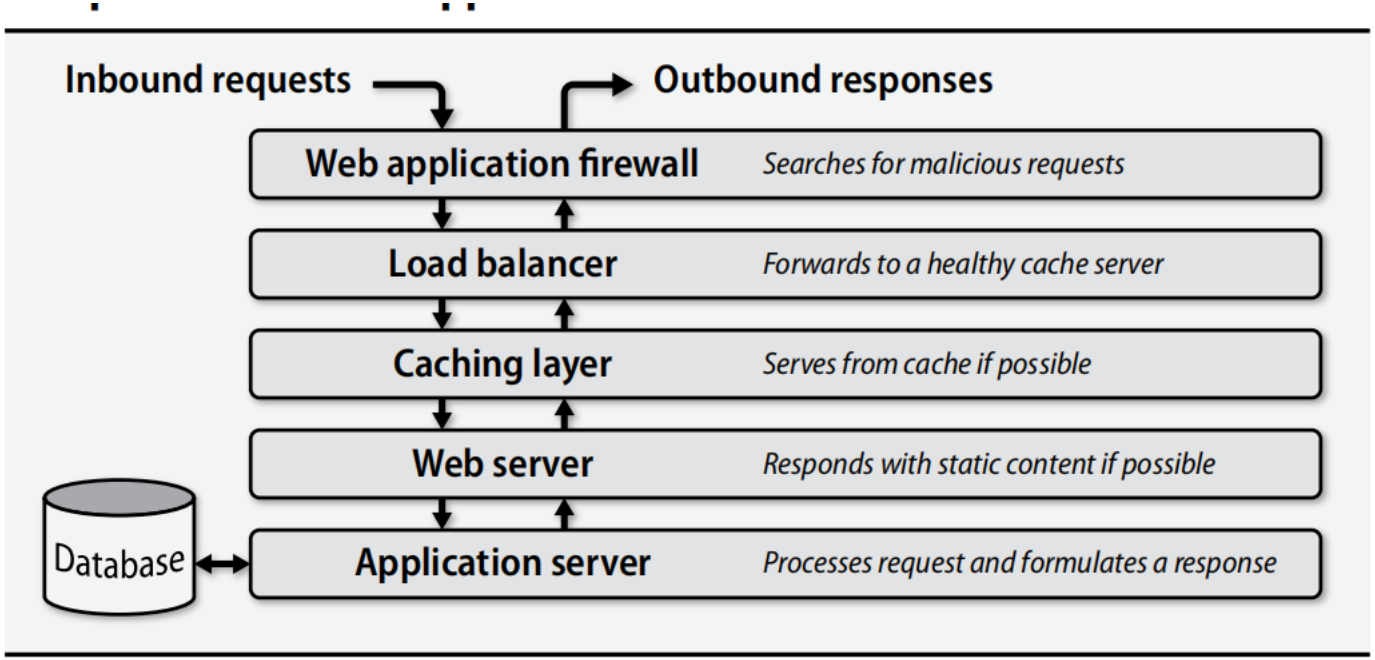
Type	Purpose	Examples
Application server	Runs web app code, interfaces to web servers	Unicorn, Tomcat
Cache	Speeds access to frequently requested content	Varnish, Squid
Load balancer	Relays requests to downstream systems	Pound, HAProxy
Web app firewall [a]	Inspects HTTP traffic for common attacks	ModSecurity
Web server	Serves static content, couples to other servers	Apache, NGINX

a. Often abbreviated WAF

4.4 Web Hosting

4.4.2. Web software basics

- Components of a web application stack



4.4 Web Hosting

4.4.2. Web software basics

Web servers and HTTP proxy software

- Most sites use web servers either to proxy HTTP connections to application servers or to serve static content directly
- A few of the features provided by web servers include:
  - Virtual hosts, allowing many sites to coexist peacefully within a single server
  - Handling of TLS connections
  - Configurable logging that tracks requests and responses
  - HTTP basic authentication
  - Routing to different downstream systems according to requested URLs
  - Execution of dynamic content through application servers

4.4 Web Hosting

4.4.2. Web software basics

Load balancers

- Load balancer is a system that manage the number of requests based on a defined algorithm to make sure no server is get overloaded.
- Load balancers solve many of the problems inherent in a single-system design:
  - Load balancers do not process requests but merely route them to other systems. As a result, they can handle many more concurrent requests than does a typical web server.
  - When a web server needs a software upgrade or has to be taken off-line for any other reason, it can easily be removed from the rotation.
  - If one of the servers experiences a problem, the health-check mechanism on the load balancer detects the problem and removes the errant system from the server pool until it becomes healthy again.

4.4 Web Hosting

4.4.2. Web software basics

Load balancers

- A few common load balancing algorithms are:
  - Round robin, in which requests are distributed among the active servers in a fixed rotation order
  - Load equalization, in which new requests go to the downstream server that's currently handling the smallest number of connections or requests
  - Partitioning, in which the load balancer selects a server according to a hash of the client's IP address. This method ensures that requests from the same client always reach the same web server.

4.4 Web Hosting

4.4.2. Web software basics

Caches

- Caches live between clients and web servers and store the results of the most frequent requests, sometimes in memory.
  - Browser caches
  - Proxy cache
  - Reverse proxy cache
- Cache problems
- Cache software

4.4 Web Hosting

4.4.2. Web software basics

- Content delivery networks
- Languages of the web
- Application programming interfaces (APIs)

4.4 Web Hosting

4.4.3. Web hosting in the cloud

- Build versus buy
- Platform-as-a-Service
- Static content hosting
- Serverless web applications

4.4 Web Hosting

4.4.4. Apache Configuration

- httpd in use
- httpd configuration logistics
- Virtual host configuration
- HTTP basic authentication
- Configuring TLS
- Running web applications within Apache
- Logging

4.4 Web Hosting

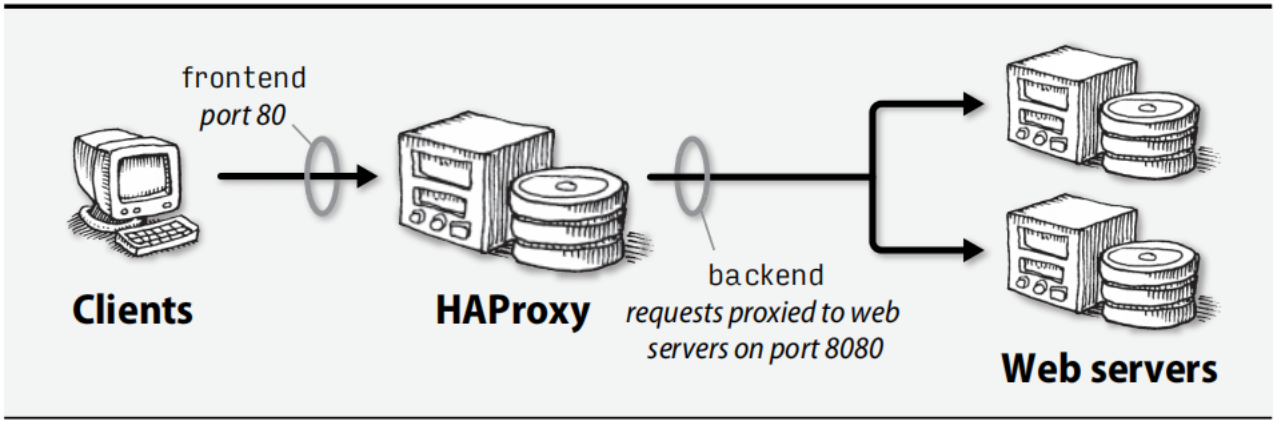
4.4.5. NGINX Configuration

- Installing and running NGINX
- Basic configurations of NGINX
- Configuring TLS for NGINX
- Load balancing with NGINX

4.4 Web Hosting

4.4.6. Load Balancing with HAProxy

HAProxy frontend and backend specifications



- Health checks
- Server statistics
- Sticky sessions
- TLS termination

Reference

- Ref 1. Evi Nemeth, Garth Snyder, Trent R. Hein, Ben Whaley and Dan Mackin “UNIX and Linux System Administration Handbook ” (5th Edition), Pearson Education, Inc., 2018.