Network. Software. Services

Basic Network Configuration. Software and Services Management

SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.bg

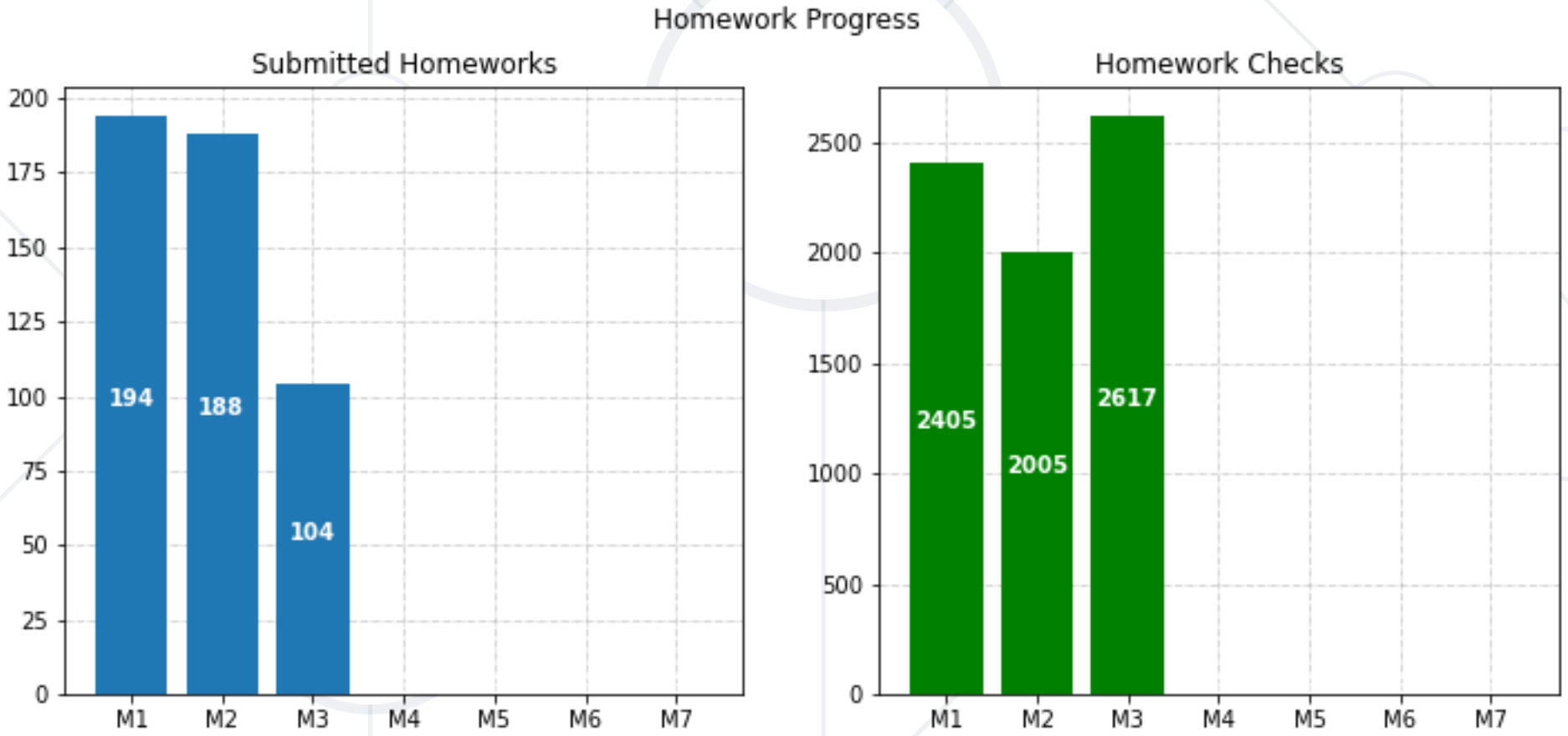
Have a Question?

Software University

sli.do

#LSA

Homework Progress



Solutions for **M3** can be submitted until **23:59:59** on **27.03.2025**

Solutions for **M4** can be submitted until **23:59:59** on **03.04.2025**

Quick Overview

Previous Module (M3)

What We Covered

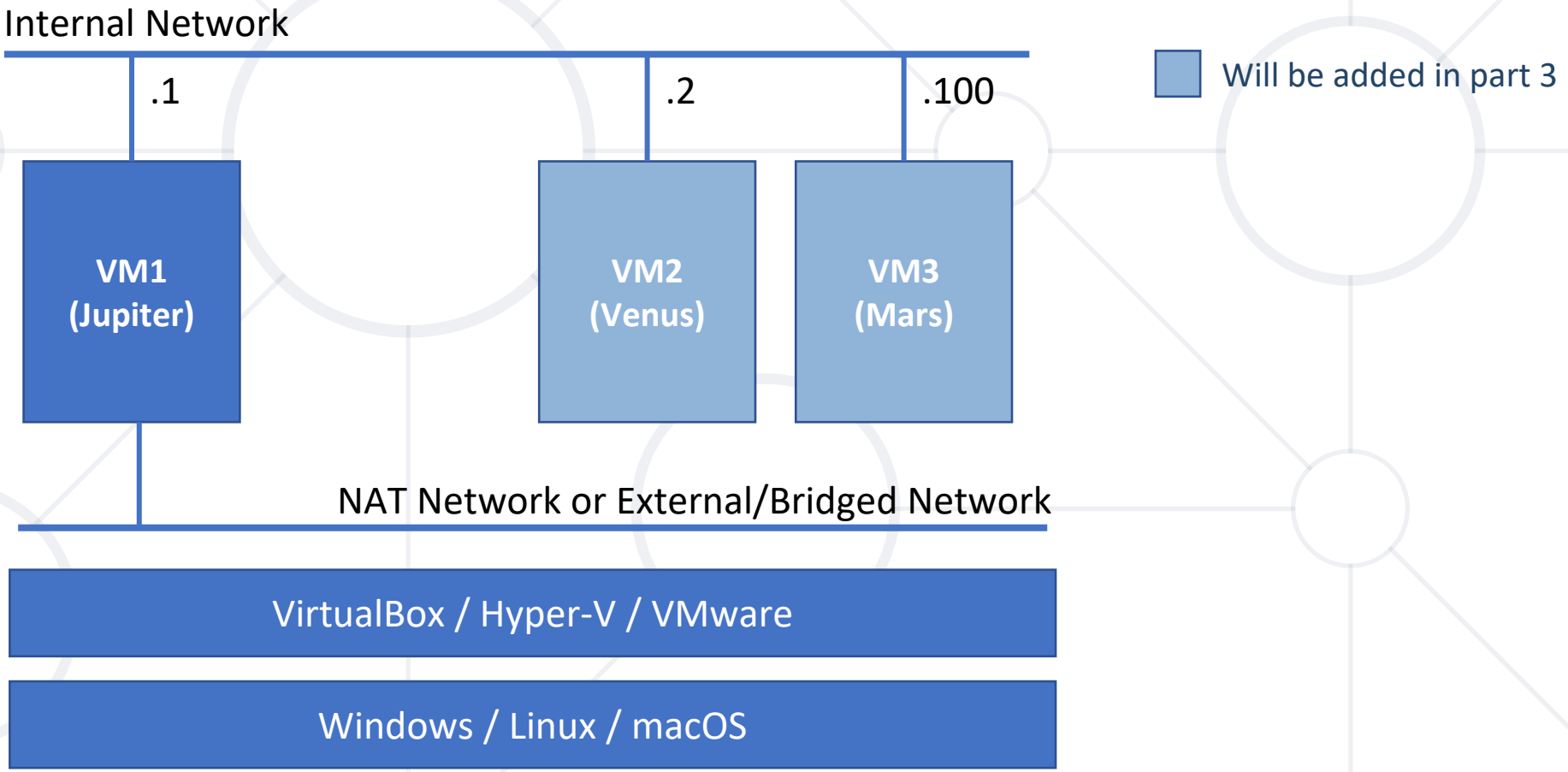
1. Working with Input / Output streams
2. Executing command sequences
3. Controlling the environment
4. Working with text editors and files
5. Using simple regular expressions
6. Searching for files
7. SUDO management

This Module (M4)

Topics and Lab Infrastructure

Table of Contents

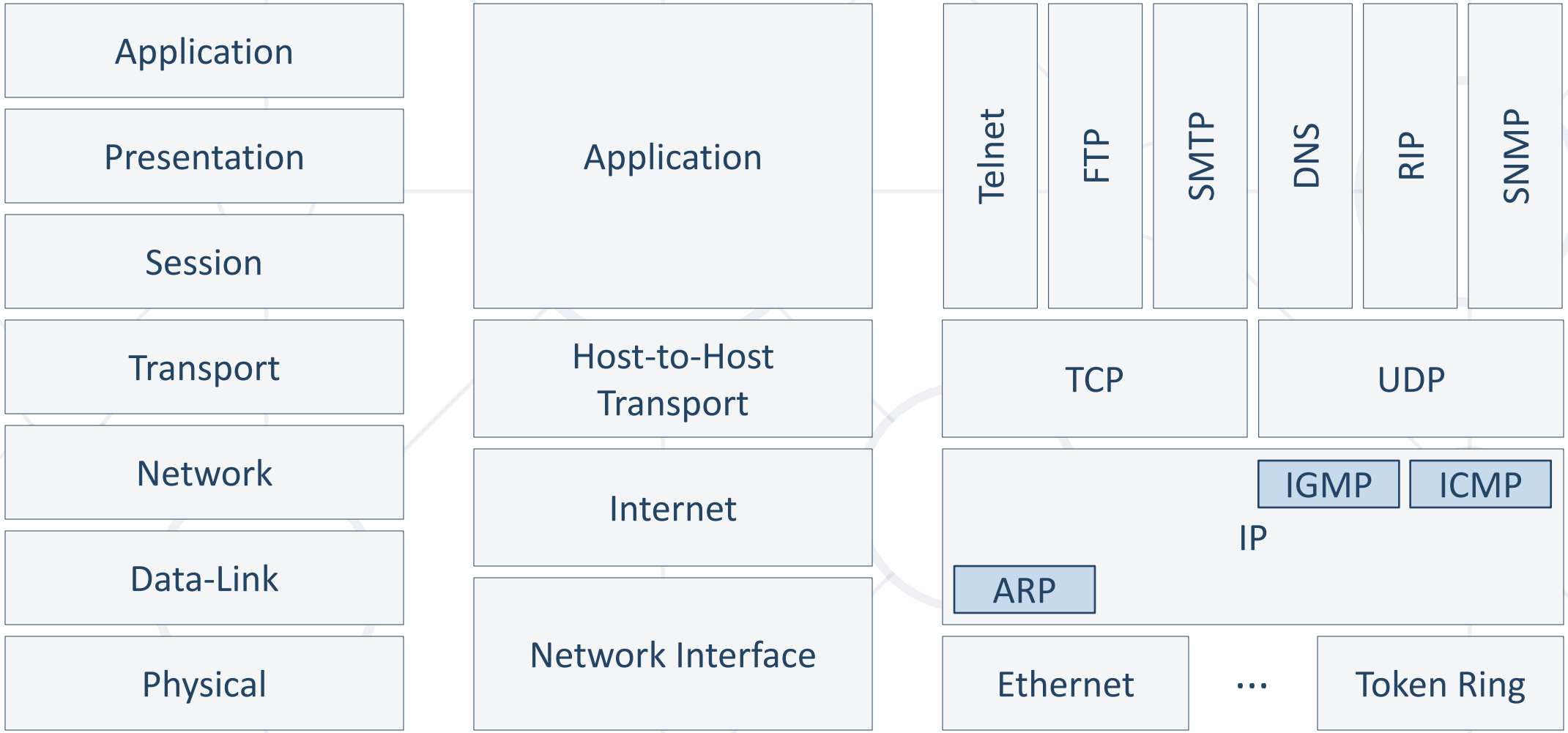
Lab Infrastructure



Network Fundamentals

Basics. OSI vs TCP. IP Addressing

Reference Network Models



Networking Protocols*

- **IP** (Internet Protocol)
  - Handles addressing and communication between devices
- **TCP** (Transmission Control Protocol)
  - It complements IP and focuses on the transport of data packages
- **UDP** (User Datagram Protocol)
  - It is like TCP, but it is connectionless, no error checking
- **ICMP** (Internet Control Message Protocol)
  - Networking devices such as routers are using it

* Brief list of protocols that most likely we will be exposed to.

Port Numbers

- Used to identify a network service (on top of **tcp** or **udp**)
- Network services registry in **`/etc/services`**
- Classification
  - **Well-known** (or system) ports: **0 - 1023**
  - **Registered** (or user) ports: **1024 - 49151**
  - **Dynamic** (and / or private) ports: **49152 - 65535**
- Some of them are
  - **22/tcp** - SSH, **53/udp** - DNS, **80/tcp** - HTTP, **110/tcp** - POP3, **123/tcp** - NTP, **143/tcp** - IMAP

IP General Information

- Main terms
  - **IP address** - either **static** or **dynamic**
  - **Network mask** - also known as **subnet mask**, it marks the border between two networks or subnets
  - **Gateway address** - it connects **two networks**
  - **Broadcast address** - used for **communication to all hosts** on a network. It is the **last address** of a subnet, and it is same for all
- Two versions - **IPv4** (**4,3 Billion**) and **IPv6** (**340 Undecillion**)

IPv4 Address Rules

- **32 bits** grouped in **4 octets** of 8 bits, which is equal to **4 Bytes**
- Written in **binary** or **decimal format** separated by **dots**
- Ranging from **0.0.0.0** to **255.255.255.255**
- Divided in two parts – **network** and **host part**
- Calculation can be made bin-to-dec and dec-to-bin

192.168.200.156

11000000.10101000.11001000.10011100

Value	128	64	32	16	8	4	2	1
Bit #	8	7	6	5	4	3	2	1
Weight	7	6	5	4	3	2	1	0

IPv4 Address Classes and Ranges

Software University

- Five address classes

Class	Leading Bits	Start	End	Default mask	CIDR Notation	Network Bytes	Host Bytes
Class A	0	0.0.0.0	127.255.255.255	255.0.0.0	/8	1	3
Class B	10	128.0.0.0	191.255.255.255	255.255.0.0	/16	2	2
Class C	110	192.0.0.0	223.255.255.255	255.255.255.0	/24	3	1
Class D	1110	224.0.0.0	239.255.255.255	n/a	n/a	n/a	n/a
Class E	1111	240.0.0.0	255.255.255.255	n/a	n/a	n/a	n/a

- **Class D** and **E** are reserved and are **not for public usage**
- **CIDR** = **Classless Inter-Domain Routing**, a method for allocating IP addresses

Special IPv4 Addresses

- **Three private** (non-routable) addresses ranges
- Private address are usually **used with custom masks**

Block	Start	End	Default mask	Notation	Addresses
24 bit	10.0.0.0	10.255.255.255	255.0.0.0	/8	16 777 216
20 bit	172.16.0.0	172.31.255.255	255.240.0.0	/12	1 048 576
16 bit	192.168.0.0	192.168.255.255	255.255.0.0	/16	65 536

Special IPv4 Addresses

- 127.0.0.0/8 is **reserved for loopback**
- Network address (all host bits are set to 0), can not be assigned
  - For 192.168.1.100/24 it is 192.168.1.0
- Broadcast address (all host bits are set to 1), can not be assigned
  - For 192.168.1.100/24 it is 192.168.1.255

IPv4 Address Exercise (Standard Mask)

- Class C address
  - IP 192.168.23.48/24
- Result
  - Network mask: 255.255.255.0
  - Network: 192.168.23.0
  - Broadcast: 192.168.23.255
  - Hosts: 254
    - $(2^{(32-24)} - 2 => 2^8 - 2 => 256 - 2)$

IPv4 Address Exercise (Non-Standard Mask)

- Class C address
  - IP 192.168.23.48/27
- Result
  - Network mask: 255.255.255.224
  - Network: 192.168.23.32
  - Broadcast: 192.168.23.63
  - Hosts: 30
    - $(2^{(32-27)} - 2 \Rightarrow 2^5 - 2 \Rightarrow 32 - 2)$

Network Device Naming

In Modern Linux Distributions

Naming Schemes

- **Traditional**
  - eth[0123...], wlan[0123...], ... - **eth0**
  - Advantage: **Easy to read**
  - Disadvantage: **Unpredictable**
- **Current**
  - Could include type, location, etc. - **enp0s3**
  - Advantage: **Predictable**
  - Disadvantage: **Hard to read**

Naming Rules

- Name Prefixes
  - **en** for **Ethernet**
  - **wl** for wireless LAN (**WLAN**)
  - **ww** for wireless wide area network (**WWAN**)
- Name Types

PCI Bus #0

enp0s3

Ethernet **Slot #3**

Format	Description
o<*index*>	On-board device index number
s<*slot*>[**f**<*function*>][**d**<*dev_id*>]	Hotplug slot index number
x<*MAC*>	MAC address
p<*bus*>**s**<*slot*>[**f**<*function*>][**d**<*dev_id*>]	PCI geographical location
p<*bus*>**s**<*slot*>[**f**<*function*>][**u**<*port*>][…][**c**<*config*>][**i**<*interface*>]	USB port number chain

Network Stack

Configuration and Testing

General Information

- Managed by **Network Manager**, **SystemD Network** or **Wicked**
- Common Tools
  - Management – (old) **ifconfig**, (old) **route**, (new) **ip**
  - Testing – **ping**, **arp**, **arping**, **traceroute**, **tracepath**, **nmap**
  - Monitoring – (old) **netstat**, (new) **ss**, **tcpdump**

General Information

- Common Files
  - Network name information – `/etc/networks`
  - Resolver configuration file – `/etc/resolv.conf`
  - Static name resolution with IP-name pairs – `/etc/hosts`
  - Name Service Switch configuration – `/etc/nsswitch.conf`

Option #1 

Software University

- Managed by **Network Manager**
- Service is **NetworkManager.service**
- Management Tools
  - **nmcli**
  - **nmtui**
- Configuration Files
  - **`/etc/NetworkManager/*`**
  - **`/etc/sysconfig/network-scripts/*`**

Option #2

Software University

- Managed by **SystemD Network (networkd)**
- Service is **systemd-networkd.service**
- Management Tools
  - **networkctl**
- Configuration Files
  - **`/etc/network/*`**
  - **`/etc/systemd/network/*`**

Option #3 

Software University

- Managed by **Wicked**
- Services are **wicked.service** and **wickedd.service**
- Management Tools
  - **wicked**
- Configuration Files
  - **`/etc/wicked/*`**
  - **`/etc/sysconfig/network/*`**

Netplan

Software University

- Network configuration abstraction
- **Introduced** in **Ubuntu 16.04**, but **default** since **Ubuntu 17.10**
- Supports **NetworkManager** and **networkd**
- Network structure is described in **YAML**
- Configuration Files
  - `/etc/netplan/*`

ip

- Description
  - Show / manipulate routing, devices, policy routing and tunnels
- Show information about links, addresses, and routes

```
[root@host ~]# ip {link|address|route|...}
```

- Add/remove address to/from an interface

```
[root@host ~]# ip addr add 192.168.1.44/32 dev
enp3s0 label enp3s0:0
[root@host ~]# ip addr del 192.168.1.44/32 dev
enp3s0
```

nmcli

- Description
  - Command line tool for controlling **NetworkManager**
- Example

```
[user@host ~]$ nmcli
...
[user@host ~]$ nmcli device show
...
[user@host ~]$ nmcli connection show
...
```

nmtui

Software University

- Description
  - Text user interface for controlling **NetworkManager**
- Example

```
# general utility
[user@host ~]$ nmtui
# specialized sub-utilities
[user@host ~]$ nmtui-connect
[user@host ~]$ nmtui-edit
[user@host ~]$ nmtui-hostname
```

wicked

- Description
  - Command line tool for controlling **wickedd**
- Example

```
[user@host ~]$ sudo wicked show eth0
eth0          up
    link:    #2, state up, mtu 1500
    type:    ethernet, hwaddr 08:00:27:e3:1b:45
    config:  compat:suse:/etc/sysconfig/network/ifcfg-eth0
    leases:  ipv4 dhcp granted
    addr:    ipv4 10.0.2.15/24 [dhcp]
    route:   ipv4 default via 10.0.2.2 proto dhcp
```

networkctl

- Description
  - Query the status of network links managed by **networkd**
- Example

```
[user@host ~]$ networkctl list
IDX LINK     TYPE         OPERATIONAL    SETUP
  1 lo       loopback     carrier        unmanaged
  2 enp0s3   ether        routable       configured

2 links listed.
[user@host ~]$
```

ping

- Description
  - Send ICMP **ECHO_REQUEST** to network hosts
- Example

```
[user@host ~]$ ping -c 4 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=128 time=0.76 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=128 time=0.56 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=128 time=0.54 ms
64 bytes from 192.168.1.1: icmp_seq=4 ttl=128 time=0.63 ms
...
[user@host ~]$
```

arp

- Description
  - Manipulate or display the system **ARP cache**
- Example

```
[user@host ~]$ arp
Address         Hwtype  Hwadress           Flags Mask  Iface
10.0.2.2        ether   52:54:00:12:35:02  C           eth0
...
[user@host ~]$ sudo arp -i eth0 -d 10.0.2.2
...
[user@host ~]$ sudo arp -i eth0 -s 10.0.2.2 52:54:00:12:35:02
...
```

arping

- Description
  - Send **ARP REQUEST** to a neighbor host
- Example

```
[user@host ~]$ arping -c 4 192.168.1.1
ARPING 192.168.1.1 from 192.168.1.100 enp0s3
Unicast reply from 192.168.1.1 [00:1B:FC:02:16:9E] 1.321ms
Unicast reply from 192.168.1.1 [00:1B:FC:02:16:9E] 1.271ms
Unicast reply from 192.168.1.1 [00:1B:FC:02:16:9E] 1.205ms
...
[user@host ~]$
```

Services Control

Manage Services with systemctl

systemctl

- Purpose
  - System service manager
- Syntax

```
systemctl [OPTIONS] COMMAND [NAME]
```

- List all active services

```
[root@host ~]# systemctl list-units --type=service
--state=active
...
```

Common Systemctl Scenarios

Software University

- Start a service

```
[root@host ~]# systemctl start sshd.service
```

- Stop a service

```
[root@host ~]# systemctl stop sshd.service
```

- Reload a service

```
[root@host ~]# systemctl reload sshd.service
```

- Restart a service

```
[root@host ~]# systemctl restart sshd.service
```

Common Systemctl Scenarios

Software University

- Show runtime status of a service

```
[user@host ~]$ systemctl status sshd.service
```

- Show properties of a service

```
[user@host ~]$ systemctl show sshd.service
```

- Enable a service

```
[root@host ~]# systemctl enable sshd.service
```

- Disable a service

```
[root@host ~]# systemctl disable sshd.service
```

Practice

Software Management

Applications. Libraries. Packages

Applications and Libraries

- Software Components
  - Binaries
  - Libraries
  - Configuration
  - Documentation
  - Data

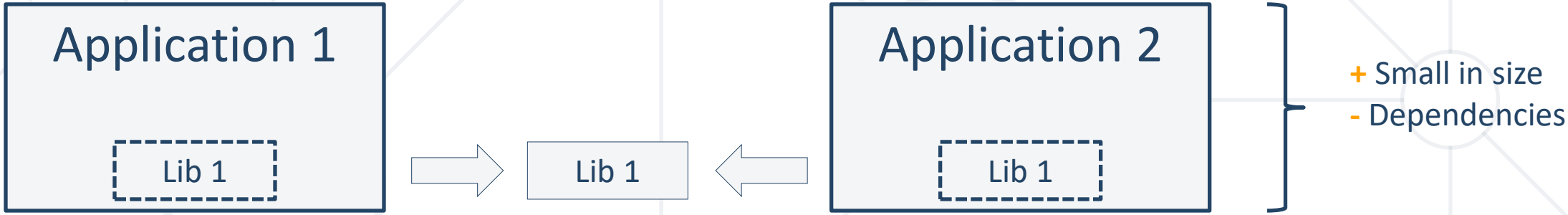
Usually are stored in different locations

- **Problem One**: **Hard to manage** all files and locations, **update and keep track of changes**

Static vs Dynamic Linking

- Static Linking

- Dynamic Linking



- **Problem Two: More Space & No Dependency Issues** OR **Less Space & Dependency Hell**

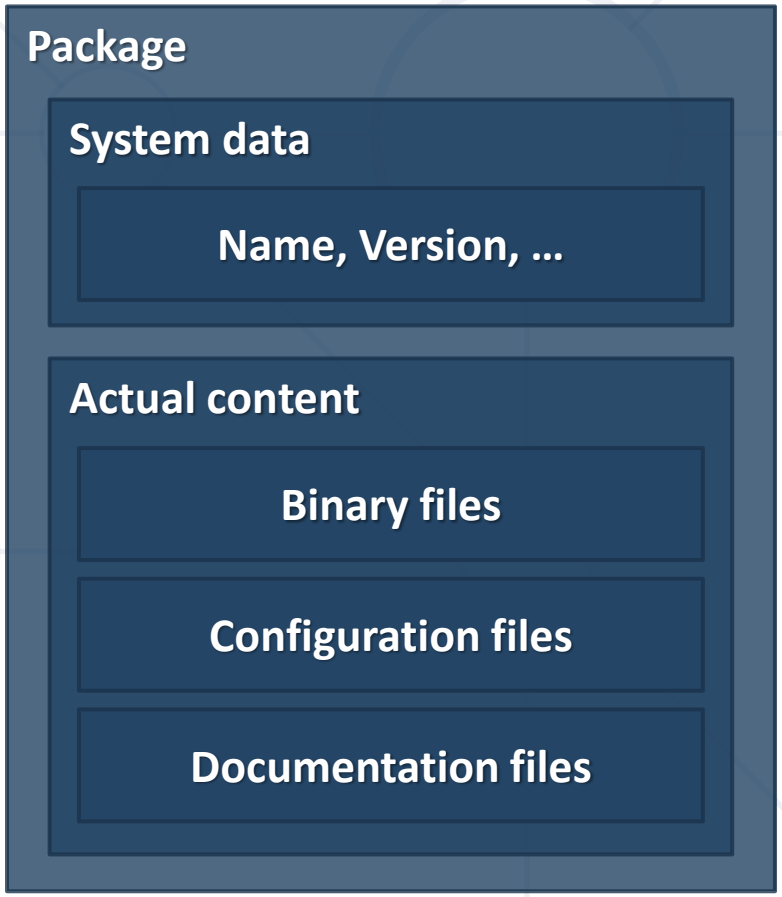
Packages Are the Solution

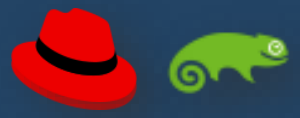
- Packages and Packaging Systems
  - Dependency tracking
  - Easier installation and update
- Two* main and widespread formats
  - **rpm - Red Hat packages**
  - **deb - Debian packages**

* There are others as well

Packages Are the Solution

- Packages are available
  - Locally as individual files
  - Through local or remote repositories (package catalogs)



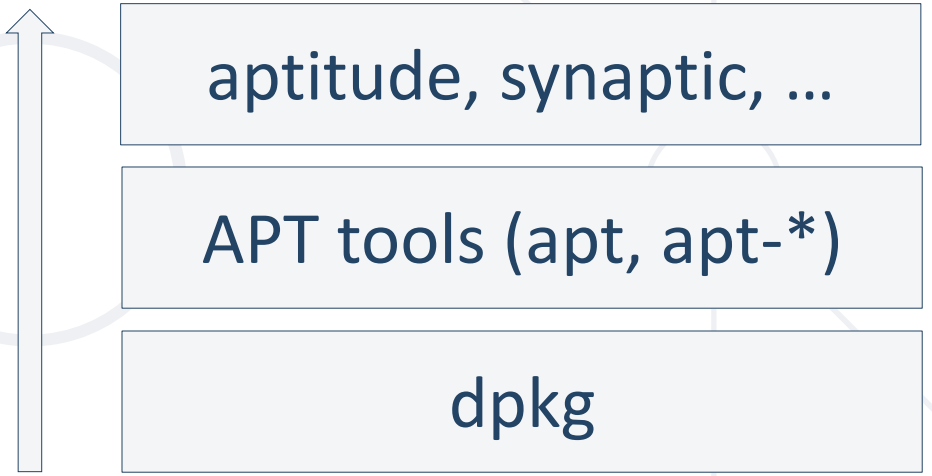
rpm 

Software University

- Stands for
  - Package format
  - Package database
  - Package management tool
- With **rpm** we can **install**, **update**, and **remove local** packages
- Different set of tools is used to work with **repositories**
  - **Fedora** (Red Hat) family is using **YUM** and **DNF**
  - **openSUSE** (SUSE) family is using **Zypper**

deb

- **dpkg** is the package management tool
  - It is used to **install**, **update**, and **remove local** packages
- For working with repositories - **searching** and **downloading**
  - Pure text based - **APT tools**
  - Pseudo graphical - **aptitude**
  - Graphical - **synaptic**



ldd

- Description
  - Print shared object dependencies
- Example

```
[user@host ~]$ ldd /bin/ls
 linux-vdso.so.1 =>  (0x00007ffda9378000)
 libselinux.so.1 => /lib64/libselinux.so.1 (0x00007f644da11000)
 libcap.so.2 => /lib64/libcap.so.2 (0x00007f644d80c000)
 libacl.so.1 => /lib64/libacl.so.1 (0x00007f644d602000)
 libc.so.6 => /lib64/libc.so.6 (0x00007f644d241000)
 libpcre.so.1 => /lib64/libpcre.so.1 (0x00007f644cfe0000)
 libdl.so.2 => /lib64/libdl.so.2 (0x00007f644cddb000)
```

Package Management

Local Packages and RPM

RPM Database

Software University

- Collection of **Berkeley database** files
- Data is located in **/var/lib/rpm**
- In case of **corruption**, it can be **fixed** by the **root**

```
[root@host ~]# rpm --rebuilddb
...
[root@host ~]#
```

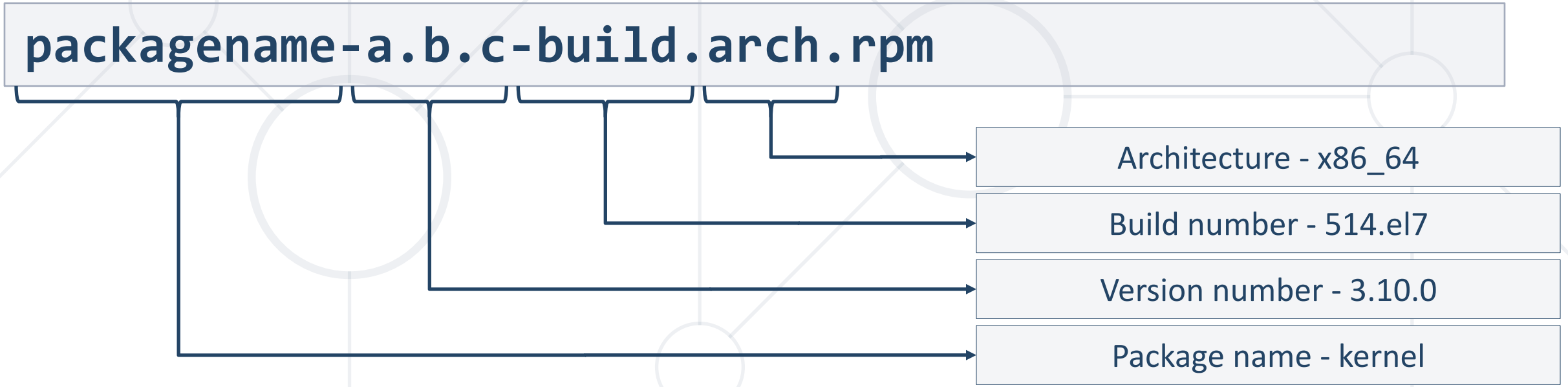
- **Only one** user can access the database at a time
- It is implemented through **lock file** **/var/lib/rpm/.rpm.lock**

RPM Package Files

- **Source** and **binary**
- **Full**, **patch**, and **delta**
- Typically, they consist of
  - **Compressed** binary files
  - **Name and version** of the package's software
  - **Build date and host** on which it was built
  - **Description** of the package and its purpose
  - **Checksums and dependencies** required

RPM Package Files

- When asking for information
  - For **installed** packages - it comes from the database
  - For **not installed** packages - it comes from packages themselves
- Package naming



rpm Command

- **Installing** packages
- **Upgrading** packages
- **Removing** and uninstalling packages
- **Querying** the RPM database for information
- **Verifying** the package file
- **Checking** installed files

Common rpm Scenarios

- Query for all installed packages

```
[user@host ~]$ rpm -qa
```

- Query for a specific installed package

```
[user@host ~]$ rpm -q kernel
```

- Query local, but not installed package

```
[user@host ~]$ rpm -qip package.rpm
```

- List all files in a package

```
[user@host ~]$ rpm -ql vim-minimal
```

Common rpm Scenarios

- Query for file dependency

```
[user@host ~]$ rpm -qf /etc/fstab
```

- Find package configuration files

```
[user@host ~]$ rpm -qc openssh
```

- Install specific package

```
[root@host ~]# rpm -ivh package.rpm
```

- Install all packages in current folder

```
[root@host ~]# rpm -ivh *.rpm
```

Common rpm Scenarios

Software University

- Upgrade (and install) all packages from files in folder

```
[root@host ~]# rpm -Uv *.rpm
```

- Freshen (upgrade if installed) packages

```
[root@host ~]# rpm -Fvh *.rpm
```

- Remove specific package

```
[root@host ~]# rpm -e package
```

- Remove specific package with specific version

```
[root@host ~]# rpm -e package-3.17.0-200.el7
```

Package Management

Repositories and YUM/DNF

YUM/DNF

Software University

- **Yellowdog Updater Modified**
- Used by most of the **Red Hat** based distributions
- Related files
  - Main configuration `/etc/yum.conf`
  - Repositories `/etc/yum.repos.d/*.repo`
  - Cache files `/var/cache/yum`
  - Log file `/var/log/yum.log`
- Replaced by **DNF** (**dandified yum**) in **Fedora 22+** and **CentOS 8+**

Common yum/dnf Scenarios

- List all available packages in the enabled repositories

```
[user@host ~]$ yum list
```

- Search for specific package

```
[user@host ~]$ yum list httpd
```

- Search for all packages starting with a word

```
[user@host ~]$ yum list http*
```

- Search for a concept

```
[user@host ~]$ yum search "web server"
```

* **dnf** can be used instead of **yum** in above examples

Common yum/dnf Scenarios

Software University

- Search package details for a given string

```
[user@host ~]$ yum search http
```

- Display details about a package

```
[user@host ~]$ yum info httpd
```

- List available groups

```
[user@host ~]$ yum groups list
```

- Search what package provides the given value

```
[user@host ~]$ yum provides httpd
```

* **dnf** can be used instead of **yum** in above examples

Common yum/dnf Scenarios

Software University

- Install a package

```
[root@host ~]# yum install httpd
```

- Install two packages without confirmation

```
[root@host ~]# yum -y install httpd php
```

- Install group of packages

```
[root@host ~]# yum groups install "GNOME Desktop"
```

- Update the whole system

```
[root@host ~]# yum upgrade
```

* **dnf** can be used instead of **yum** in above examples

Common yum/dnf Scenarios

Software University

- Update specific package

```
[root@host ~]# yum upgrade httpd
```

- Remove cached data

```
[root@host ~]# yum clean
```

- Remove installed package

```
[root@host ~]# yum erase httpd
```

- Remove installed group

```
[root@host ~]# yum groups erase "GNOME Desktop"
```

* **dnf** can be used instead of **yum** in above examples

Install EPEL Repository

- Extra Packages for **Enterprise Linux** (EPEL)
  - Open source and free community-based repository project from Fedora team
- Install EPEL repository

```
[root@host ~]# yum install epel-release
...
[root@host ~]# yum repolist
...
epel/x86_64     Extra Packages for Enterprise ...
...
```

Package Management

Repositories and zypper

zypper

Software University

- Used by **openSUSE** based distributions
- CLI (**zypper**) and package manager (**ZYpp** or **libzypp**)
- Related files
  - Configuration `/etc/zypp/zypper.conf` and `/etc/zypp/zypp.conf`
  - Repositories `/etc/zypp/repos.d/*.repo`
  - Cache files `/var/cache/zypp`
  - Log file `/var/log/zypper.log` and `/var/log/zypp/history`

Common Zypper Scenarios

Software University

- List all available packages in the enabled repositories

```
[user@host ~]$ zypper packages
```

- Shows details for a specific package

```
[user@host ~]$ zypper info wget
```

- Search for all packages that have a httpd in their name

```
[user@host ~]$ zypper search http
```

- Search packages by specific word

```
[user@host ~]$ zypper what-provides httpd
```

Common Zypper Scenarios

- Install a package

```
[root@host ~]# zypper install nano
```

- Remove a package

```
[root@host ~]# zypper remove nano
```

- Update installed packages

```
[root@host ~]# zypper update
```

- Perform a distribution upgrade

```
[root@host ~]# zypper dist-upgrade
```

Common Zypper Scenarios

Software University

- List all defined repositories

```
[user@host ~]$ zypper repos
```

- Add a new repository

```
[root@host ~]# zypper addrepo -r repository.repo
```

- Remove specified repository

```
[root@host ~]# zypper removerepo 15
```

- Refresh all repositories

```
[root@host ~]# zypper refresh
```

Common Zypper Scenarios

Software University

- List all defined patterns

```
[user@host ~]$ zypper patterns
```

- Install a pattern

```
[root@host ~]# zypper install -t pattern lxde
```

- Remove a pattern

```
[root@host ~]# zypper remove -t pattern lxde
```

- Clean both metadata and package caches

```
[root@host ~]# zypper clean --all
```

Package Management

Local Packages and dpkg

DEB Database

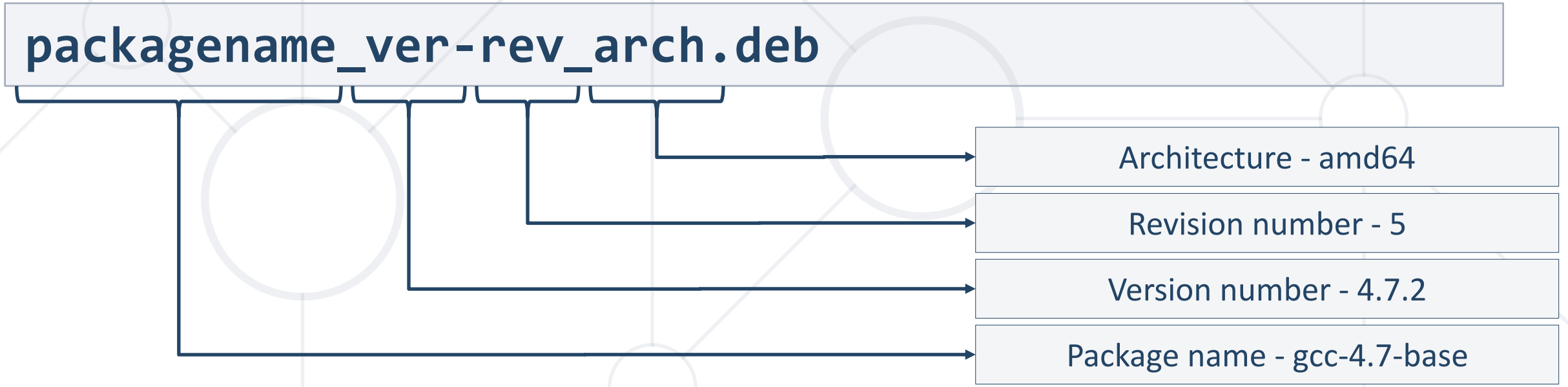
- Collection of files and folders
- Data is located in `/var/lib/dpkg`
- In case of **corruption**, it can be **fixed** by following a procedure
- **Only one** user can write to the database at a time
- It is implemented through **lock file** `/var/lib/dpkg/lock`

DEB Package Files

- **Source** and **binary**
- **ar** archive file
- Three parts
  - Header - **debian-binary**
  - Control Archive - **control.tar.**{**gzip**, **xz**}
  - Data Archive - data.tar.{**gzip**, **bzip2**, **lzma**, **xz**}

DEB Package Files

- When asking for information
  - For **installed** packages - it comes from the database
  - For **uninstalled** packages - it comes from the packages themselves
- Package naming



Common dpkg Scenarios

Software University

- List currently installed packages

```
[user@host ~]$ dpkg --list
```

- Print information about installed package

```
[user@host ~]$ dpkg --status wget
```

- List installed files associated with a package

```
[user@host ~]$ dpkg --listfiles wget
```

- Locate the package that own a file

```
[user@host ~]$ dpkg --search /etc/bash.bashrc
```

Common dpkg Scenarios

Software University

- Install specific package

```
[user@host ~]$ sudo dpkg --install package.deb
```

- Remove a package but leave the configuration files

```
[user@host ~]$ sudo dpkg --remove package
```

- Remove a package including the configuration files

```
[user@host ~]$ sudo dpkg --purge package
```

- Reconfigure an installed package

```
[user@host ~]$ sudo dpkg --configure package
```

Package Management

Repositories and apt/apt-*

apt

- Command line **package manager**
- Used by **Debian**-based distributions
- Handles **search**, **install**, **remove**, and **update** of packages
- Offers options more suitable for interactive use
- Configuration is stored in **/etc/apt/***

Common apt Scenarios

- List all available packages based on package names

```
[user@host ~]$ apt list apache* php*
```

- Install a package

```
[user@host ~]$ sudo apt install apache2 php
```

- Remove a package

```
[user@host ~]$ sudo apt remove apache2
```

- Update list of available packages

```
[user@host ~]$ sudo apt update
```

Common apt Scenarios

Software University

- Get list of packages with available upgrades

```
[user@host ~]$ apt list --upgradable
```

- Show package details

```
[user@host ~]$ apt show apache2
```

- Upgrade the system

```
[user@host ~]$ sudo apt upgrade
```

- Edit the source information file

```
[user@host ~]$ sudo apt edit-sources
```

apt-* Tools

Software University

- Used by **Debian**-based distributions
- It runs on top of **APT**
- It is a lower-level tool compared to **aptitude** and **synaptics**
- The package includes tools for every task. Some of them are:
  - Installation, update, and removal of packages - **apt-get**
  - Package searching - **apt-cache**
  - Key management - **apt-key**
  - Configuration management - **apt-config**

Common apt-* Scenarios

Software University

- List cache statistics

```
[user@host ~]$ apt-cache stats
```

- Search for package

```
[user@host ~]$ apt-cache search wget
```

- List stored keys

```
[user@host ~]$ apt-key list
```

- Add key

```
[user@host ~]$ sudo apt-key add docker-ce.gpg
```

Common apt-* Scenarios

Software University

- Remove installed key

```
[user@host ~]$ sudo apt-key remove 0EBFCD88
```

- List configured repositories and their priorities

```
[user@host ~]$ apt-cache policy
```

- Add repository

```
[user@host ~]$ sudo apt-add-repository http://repo.url.com
```

- Remove repository

```
[user@host ~]$ sudo apt-add-repository -r http://repo.url.com
```

Common apt-* Scenarios

- Synchronize package index files with their sources (repositories)

```
[user@host ~]$ sudo apt-get update
```

- Install the newest versions of all currently installed packages

```
[user@host ~]$ sudo apt-get upgrade
```

- Install package

```
[user@host ~]$ sudo apt-get install httpd
```

- Remove package with all configuration files

```
[user@host ~]$ sudo apt-get purge httpd
```

Software Management

Alternative (Universal) Package Formats

A Brief Comparison*

Software University

Feature	Traditional Packages	AppImage	Flatpak	Snap
Portability	Low	High	High	Medium/High
Installation Required	Yes	No	Yes (no system changes)	Yes
Sandboxing	No	No	Yes	Yes
Dependency Management	Excellent	Self-Contained	Runtimes	Self-Contained
Update Mechanism	System Package Manager	Manual/3rd Party Tools	Automatic	Automatic
Package Size	Smallest	Largest	Medium/Large	Medium/Large
System Integration	Deep	Minimal (additional steps)	Partial/Improving	Partial/Improving
Performance	Best	Good	Almost Native	Can be slower/Improving
Centralization	Distribution Repositories	Not exactly	Flathub (and others)	Snap Store (by Canonical)

AppImage: https://appimage.org/ and https://appimage.github.io/apps/

Flatpak: https://flatpack.org and https://flathub.org

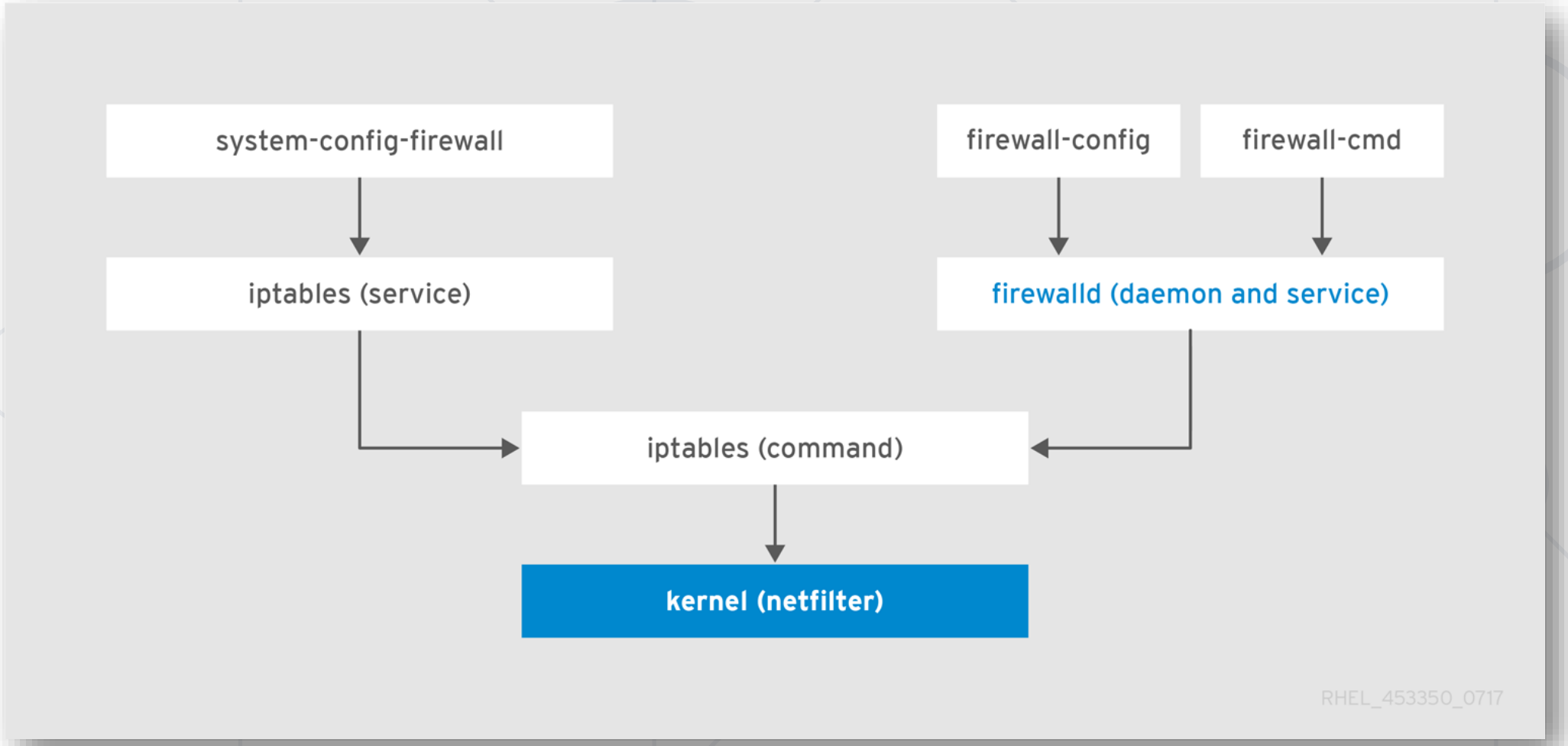
Snap: https://snapcraft.io/

Practice

Basic Network Services

Firewall

Linux Firewall



https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Security_Guide/sec-Using_Firewalls.html

Firewall #1

- **firewalld**
  - Dynamically managed firewall with support for **zones**
- Managed through
  - Command line interface - **firewall-cmd**
  - Special offline tool - **firewall-offline-cmd**
  - Graphical application - **firewall-config**
- Configuration files:
  - `/etc/firewalld/*` and `/usr/lib/firewalld/*`

Common firewall-cmd Commands

Software University

- Check if firewall is running

```
[user@host ~]$ sudo firewall-cmd --state
```

- Get firewall zones

```
[user@host ~]$ firewall-cmd --get-zones
```

- Check the default zone

```
[user@host ~]$ firewall-cmd --get-default-zone
```

- Set default zone

```
[user@host ~]$ sudo firewall-cmd --set-default-zone=internal
```

Common firewall-cmd Commands

Software University

- Get firewall services

```
[user@host ~]$ firewall-cmd --get-services
```

- Add firewall service*

```
[user@host ~]$ sudo firewall-cmd --add-service=http --permanent
```

- Add firewall port*

```
[user@host ~]$ sudo firewall-cmd --add-port=22/tcp --permanent
```

- Reload firewall configuration

```
[user@host ~]$ sudo firewall-cmd --reload
```

* To remove service or port, change the **add** prefix to **remove**. Equal signs (=) could be skipped.

Firewall #2

Software University

- **ufw**
  - The **Uncomplicated Firewall** is a frontend for **iptables**
  - ufw provides a **framework** for managing **netfilter**
  - and **command-line** interface for manipulating the firewall
- Managed through
  - Command line interface - **ufw**
- Configuration files:
  - **`/etc/default/ufw`** and **`/etc/ufw/*`**

Common ufw Commands

- Check if firewall is running

```
[user@host ~]$ sudo ufw status
```

- Show extended firewall status

```
[user@host ~]$ sudo ufw status verbose
```

- Report ports that are in listening state tcp or open state for udp

```
[user@host ~]$ sudo ufw show listening
```

- Complete firewall report

```
[user@host ~]$ sudo ufw show raw
```

Common ufw Commands

- List allowed applications

```
[user@host ~]$ sudo ufw app list
```

- Allow port rule*

```
[user@host ~]$ sudo ufw allow 80/tcp
```

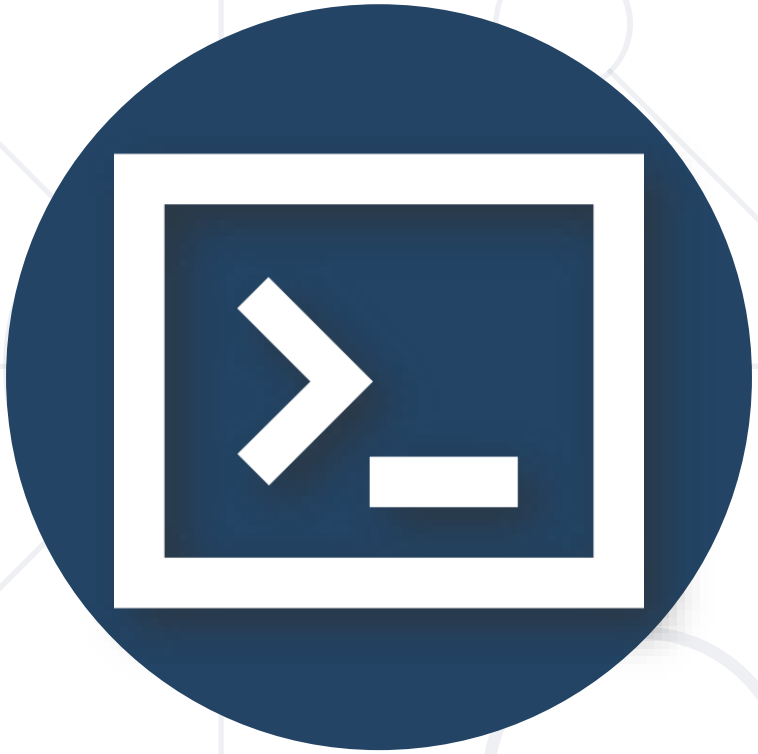
- Allow application rule*

```
[user@host ~]$ sudo ufw allow OpenSSH comment 'allow ssh app'
```

- Reload firewall configuration

```
[user@host ~]$ sudo ufw reload
```

* To remove port or application, use the **delete** command



Secure Shell (SSH)

Introduction

Secure Shell (SSH) *

- Provides **secure way** to access computer over an unsecured network
- By default, it is **installed** on **Red Hat** and **openSUSE**
- By default, it is **started** on **Red Hat**
- Package
  - **openssh-server**
  - **openssh**

* The actual status (installed/not installed and running/not running) may vary between distributions and their versions

Secure Shell (SSH)

- Configuration:
  - **Server** - `/etc/ssh/sshd_config`
  - **Client** - `/etc/ssh/ssh_config`
- Listens on **TCP** port **22**
- Popular clients for Windows – PuTTY, KiTTY, WinSCP, OpenSSH
- Our task: **Explore SSH connectivity**

Dynamic Host Configuration Protocol

Introduction

Dynamic Host Configuration Protocol

- Commonly referred to as **DHCP**
- Dynamically sets **network configuration** for connected devices
- By **default**, only the client is **installed**
- Package name
  - **dhcp-server**
  - **isc-dhcp-server**

Dynamic Host Configuration Protocol

- Configuration:
  - **Red Hat** & **Debian**: **`/etc/dhcp/dhcpd.conf`**
  - **openSUSE**:
    - **`/etc/sysconfig/dhcpd`**
    - **`/etc/dhcpd.conf`**
- Listens on **UDP** port **67**
- Our task: **Configure client-server environment**

Practice

Summary

- We know that there are multiple network models
- There are also two addressing schemes IPv4 and IPv6
- Modern Linux support multiple network device naming schemes
- There are at least three software solutions for network management

Summary

- deb and rpm are the two most popular packaging formats
- Each one has its own set of tools both low and high level, cli and gui based
- Debian family is dpkg as low-level tool and apt tools as high level
- Fedora family is using rpm as low-level tool and yum or dnf as high level
- openSUSE family is using combination of rpm and zypper

Resources

- RPM packages
  - https://en.wikipedia.org/wiki/Rpm_(software)
- DEB packages
  - https://en.wikipedia.org/wiki/Deb_(file_format)
  - https://www.debian.org/doc/manuals/debian-faq/ch-pkg_basics.en.html
- Netplan Documentation
  - https://wiki.ubuntu.com/Netplan
  - https://wiki.ubuntu.com/MigratingToNetplan

Resources

- Ubuntu Uncomplicated Firewall
  - *https://wiki.ubuntu.com/UncomplicatedFirewall*
  - *https://help.ubuntu.com/lts/serverguide/firewall.html*
- Red Hat Networking and Security (incl. Firewall) Documentation
  - *https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/networking_guide/*
  - *https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/security_guide/index*
- openSUSE Leap Documentation
  - *https://doc.opensuse.org/*

Questions?

SoftUni

Software University

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

СофтУни БУДИТЕЛ

Finance Academy

SoftUni Diamond Partners

Software University

SmartIT

Coca-Cola HBC Bulgaria

SUPER HOSTING .BG

createX

Postbank

top EMPLOYER България Bulgaria 2024 FOR A BETTER WORLD OF WORK

DRAFT KINGS

THE CROWN IS YOURS

VIVACOM

INDEAVR

Serving the high achievers

AMBITIONED

PHAR VISION

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg