Linux

Sectiunea 1
Sistem de fisiere

CUPRINS

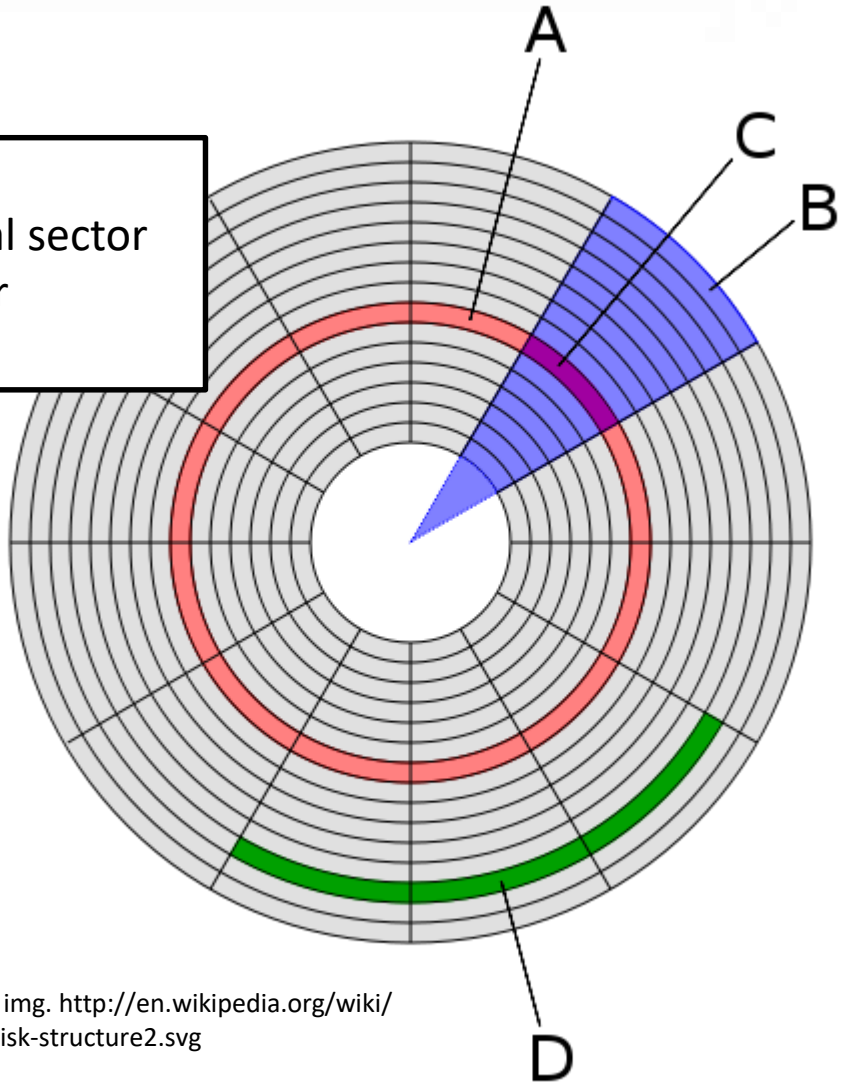
- sistem de fisiere Linux
- partitii
- comenzi utilizare fisiere

LINUX
SISTEMUL DE FISIERE – FILE SYSTEM

Sistem de fisiere
Disc

- un disc este alcatuit din:

(A) Track
(B) Geometrical sector
(C) Track sector
(D) Cluster



Sistemele de fisiere cel mai des intalnite in Linux:

ext , extended file system
ext2, the second extended file system.
ext3, the third extended file system.
ext4, the fourth extended file system.

Sursa img. http://en.wikipedia.org/wiki/File:Disk-structure2.svg

Partitionarea memorie

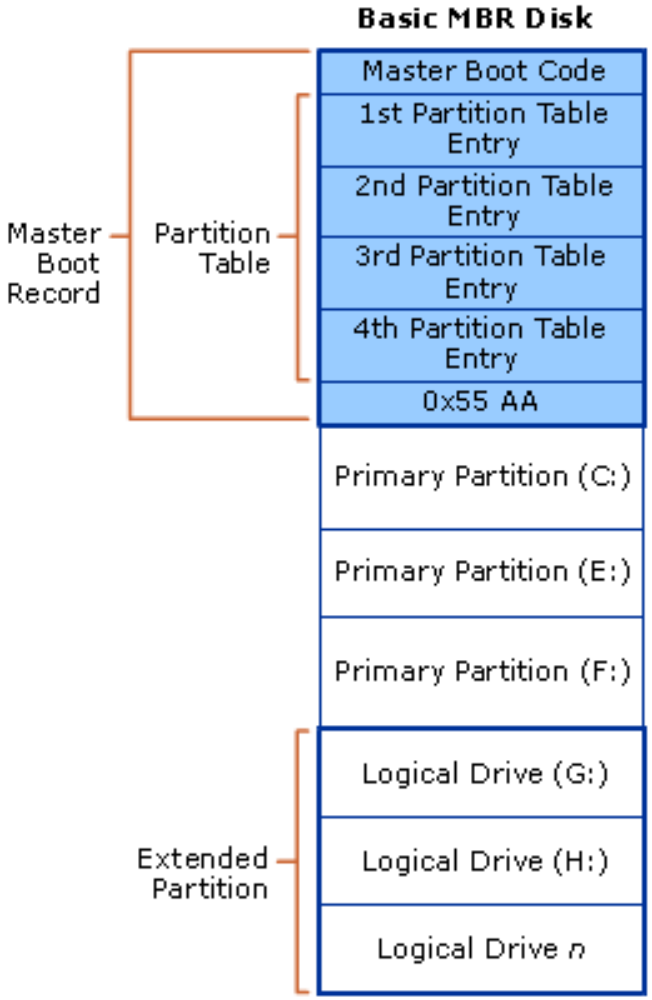
- **Partitii (**eng. **Partition)**
  - Pentru a instala un OS pe un disc este nevoie de impartirea discului in sectiuni numite partitii
  - Partiționarea memoriei împarte spațiul disponibil în secțiuni care pot fi accesate independent.
  - Intreaga memorie poate fi alocată unei partitii sau impartita intr-un numar de partiții
  - Exemple de partitii:
    - boot dual – cate o partitie pentru fiecare SO
    - menținerea unei partiții swap
    - separarea logică a datelor cum ar fi fișiere audio și video.

Partitionarea memorie

- **Tabel de partitii (**eng. **Partition Tables)**
  - Informațiile necesare sunt stocate într-o schemă de tabele de partiții
  - Contine o descriere a partitiilor disponibile pe disc
  - Standarde utilizate:
    - MBR (Master Boot Record)
      - cunoscut si ca standard asociat cu ms-dos
    - GPT (GUID Partition Table).

Partitii

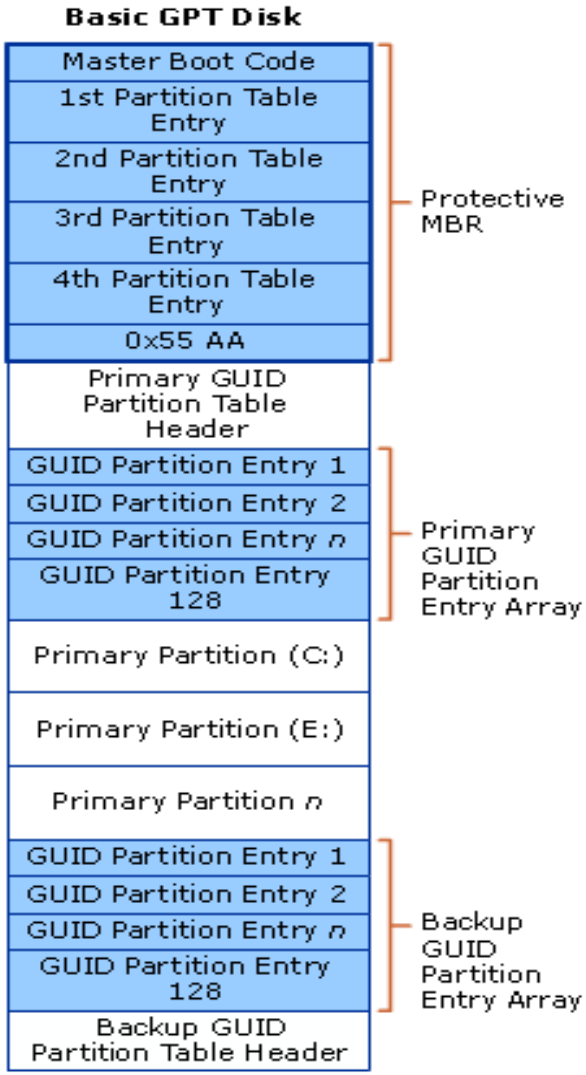
- **Master Boot Record (MBR)**
  - 3 tipuri de partitii: Primary, Extended, si Logical.
  - Maxim 4 partitii
  - Pentru mai mult de 4 partitii se creaza 3 partitii primare si maxim o partitie extinsa care va cuprinde un numar de partitii logice
  - Orice spatiu nealocat unei partitii este spatiu liber (dar nu poate fi utilizat de OS, pana nu este alocat.)



https://*msdn.microsoft.com/en-us/library/windows/desktop/aa363785(v=vs.85).aspx*

Partitii

- **GUID Partition Table (GPT)**
  - Compenseaza limitarile MBR
  - Posibilitatea adaugarii unui numar mai mare de 4 partitii primare
  - Partitii mai mari de 2 TB



https://*msdn.microsoft.com/en-us/library/windows/desktop/aa363785(v=vs.85).aspx*

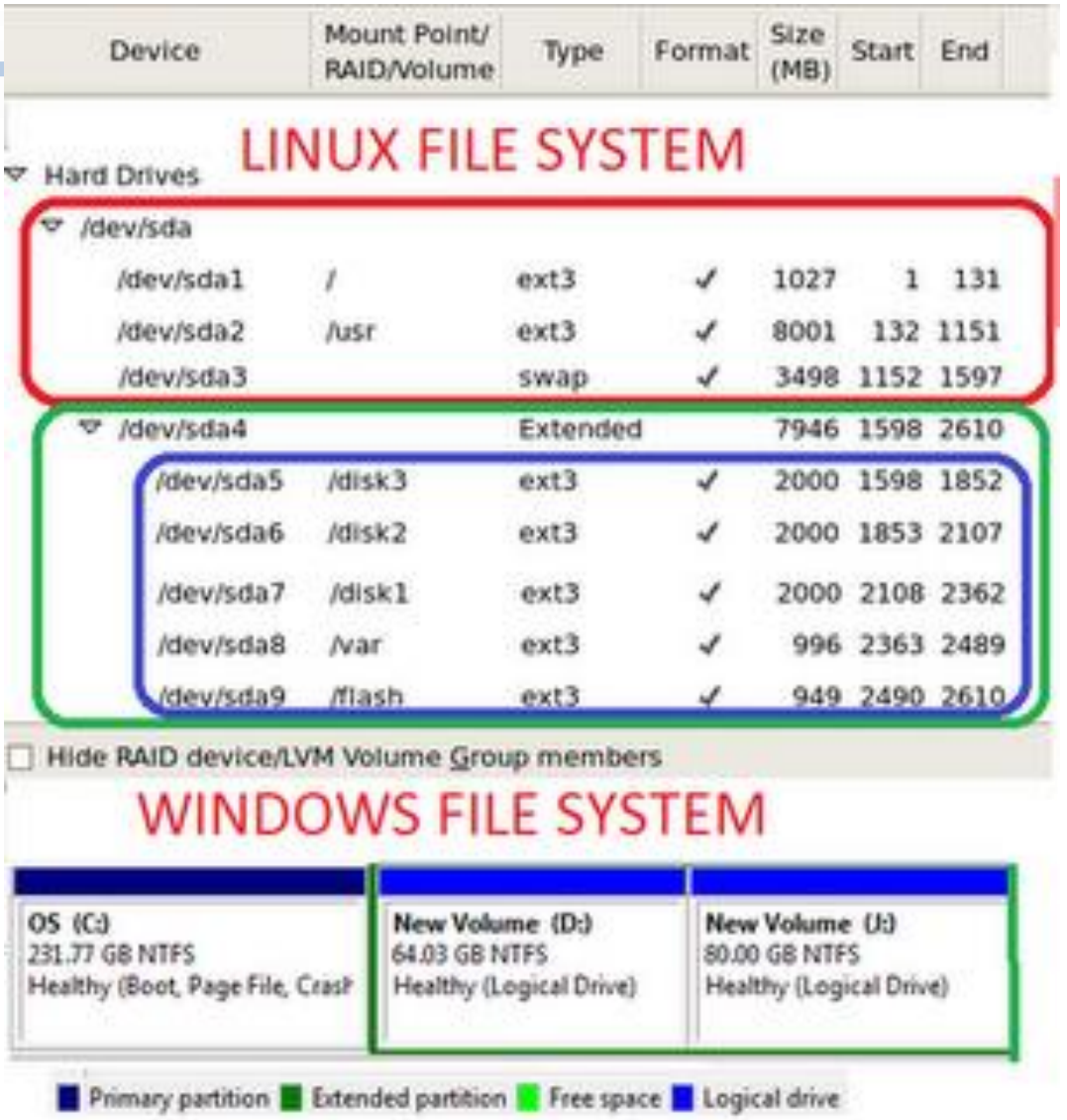
```
Disk /dev/sda: 320.1 GB, 320072933376 bytes
255 heads, 63 sectors/track, 38913 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xaa692010

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1        7681    61690880    7  HPFS/NTFS
/dev/sda2            7681       14182    52219904    7  HPFS/NTFS
/dev/sda3           14182       20556    51200000    7  HPFS/NTFS
/dev/sda4           20556       38913   147453953    f  W95 Ext'd (LBA)
/dev/sda5           20556       32030    92160000    7  HPFS/NTFS
/dev/sda6           34324       34770     3583999+  82  Linux swap / Solaris
/dev/sda7           34771       38913    33276928    7  HPFS/NTFS
/dev/sda8           32030       34323    18423808   83  Linux

Partition table entries are not in disk order
```

Sistem de fisiere

- **Sistem de fisiere (**eng. **File Systems)**
  - modalitate de stocare a informațiilor pe un computer
  - constă, de obicei, dintr-un arborele de directoare utilizat pentru a organiza fișiere (organizare ierarhica) .
  - o partitie trebuie formatata inainte sa poata gestiona date
  - prin formatare se asociaza un sistem de fisiere
  - fiecare partiție pot avea un sistem de fișiere diferit, dacă se dorește.

Sistem de fisiere

- Sisteme de fisiere des intalnite (linux)
  - **Ext** - “Extended file system” versiuni ext2 -ext4
  - BtrFS - B-Tree File System
  - XFS
  - JFS - “Journaled File System”
  - ZFS - ZFSonLinux



Sistem de fisiere

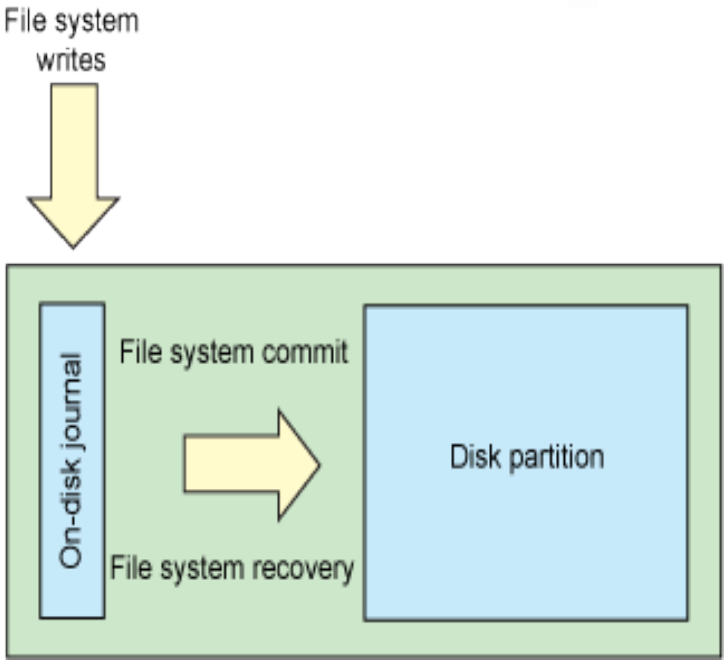
- Linux – conventie notare
  - “dev” – prescurtare de la “device”
  - “sd” – prescurtare de la SCSI mass-storage driver. (Small Computer System Interface.)
    - Primul disc detectat este denumit sda (discul 0), al doilea disc sdb (discul 1) ...
  - “tmpfs” – partitie temporara gasduita in memoria principala – RAM
  - “swap” - sectiune salvata pe disc utilizata ca memorie virtuala (memoria principala este formata din memorie fizica (RAM) si memorie virtuala swap)

Sistem de fisiere ext4

- Sistemul de fisiere ext4
  - sistem de fisiere de tip jurnal (se retin operatiile efectuate)
  - adresare de 48-biti pentru un bloc
  - dimensiunea maxima a unui fisier poate varia intre 16 GB si 16 TB
  - dimensiunea maxima a unui sistem de fisiere ext4 este de 1 EB (exabyte). 1 EB = 1024 PB (petabyte). 1 PB = 1024 TB (terabyte).
  - Un director poate contine maxim 64,000 subdirectoare

Sistem de fisiere jurnalizare

- Sistem de fișiere cu jurnalizare
  - menține un fișier special numit jurnal care este utilizat pentru a repara orice neconcordanță care apare ca urmare a erori (ex: închideri necorespunzătoare a unui computer).
  - inregistreaza date despre fișiere și directoare în jurnalul salvat pe disc înainte de fiecare operatie.
  - in caz de eroare sistemul va citi jurnalul, care poate fi rulat până la cel mai recent punct de coerență a datelor.



Sursa: https://www.ibm.com/developerworks/library/l-journaling-filesystems/index.html

Mount

- In sistemul de operare Linux fisiere si directoare sunt atasate pentru fiecare dispozitiv I/O.
  - Atasarea unui director catre un dispozitiv I/O este cunoscuta ca “mounting”
  - Directorul este cunoscut ca “mount point”
  - Accesarea dispozitivului se face prin intermediul directorului atasat
  - Dispozitivele se gasesc definite de obicei in directorul /dev

Mount

- Comanda mount
  - fdisk –l -vizualizati sistemele de fisiere
  - mount /dev/fd0
  - mount /dev/cdrom
  - umount /mnt/floppy
  - umount /dev/fd0

Sistem de fisiere Linux
Comenzi

- Comanda ls
  - ls -li test
  - ls -i /etc/passwd

```
upbvm063:/ # ls -i
  4401 aquota.group  210913 bin      352 dev   16225 home
  4400 aquota.user        2 boot  592177 etc    8113 lib
```

Sistem de fisiere Linux
Comenzi

- Comanda df
  - este folosita pentru a afisa spațiu liber pe disc
  - df –i
  - df –h

```
upbvm063:/ # df -i
Filesystem            Inodes   IUsed   IFree IUse% Mounted on
/dev/sda2            1695408  134978 1560430    8% /
udev                  506627     753  505874    1% /dev
/dev/sda1              24192      42   24150    1% /boot
upbvm063:/ # df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              26G  3.5G   21G  15% /
udev                  2.0G  112K  2.0G   1% /dev
/dev/sda1              92M   20M   68M  23% /boot
```

dimensiune sistem de fisiere

spatiu disponibil sistem de fisiere

Comenzi legate de sistemul de fisiere

root@host-6-1:~

```
[root@host-6-1 ~]# ls -li test
18284181 -rw-r--r--. 1 root root 69 Nov 12 22:13 test
[root@host-6-1 ~]# ls -i test
18284181 test
[root@host-6-1 ~]# df
Filesystem                   1K-blocks    Used Available Use% Mounted on
/dev/mapper/centos_lvm-root    6813696 4100336   2713360  61% /
devtmpfs                        241940       0    241940   0% /dev
tmpfs                           250952      80    250872   1% /dev/shm
tmpfs                           250952    4780    246172   2% /run
tmpfs                           250952       0    250952   0% /sys/fs/cgroup
/dev/sda1                       508588  136588    372000  27% /boot
[root@host-6-1 ~]# df -i
Filesystem                  Inodes  IUsed   IFree IUse% Mounted on
/dev/mapper/centos_lvm-root 6823936 155265 6668671    3% /
devtmpfs                      60485    353   60132    1% /dev
tmpfs                         62738      5   62733    1% /dev/shm
tmpfs                         62738    446   62292    1% /run
tmpfs                         62738     13   62725    1% /sys/fs/cgroup
/dev/sda1                    512000    336  511664    1% /boot
[root@host-6-1 ~]#
```

Informatii despre disc

Comenzi legate de sistemul de fisiere

root@host-6-1:~

```
[root@host-6-1 ~]# df -h
Filesystem                Size  Used Avail Use% Mounted on
/dev/mapper/centos_lvm-root  6.5G  4.0G  2.6G  61% /
devtmpfs                  237M     0  237M   0% /dev
tmpfs                     246M   80K  245M   1% /dev/shm
tmpfs                     246M  4.7M  241M   2% /run
tmpfs                     246M     0  246M   0% /sys/fs/cgroup
/dev/sda1                 497M  134M  364M  27% /boot
[root@host-6-1 ~]#
```

Comenzi

Extras din man grep & man awk

- grep
  - -globally search a regular expression and print
  - v, --invert-match .
    - Invert the sense of matching, to select non-matching lines
  - -E, --extended-regexp
    - Interpret PATTERN as an extended regular expression (see below).
  - if the first character of the list is the caret ^ then it matches any character not in the list

Script verificare memorie

- Explicatie

Rezultatul comenzii

```
[root@host-7-65 ~]# df -H
Filesystem                Size  Used Avail Use% Mounted on
/dev/mapper/centos_lvm-root  7.0G  4.2G  2.8G  61% /
devtmpfs                  248M     0  248M   0% /dev
tmpfs                     257M   82k  257M   1% /dev/shm
tmpfs                     257M  4.9M  253M   2% /run
tmpfs                     257M     0  257M   0% /sys/fs/cgroup
/dev/sda1                 521M  140M  381M  27% /boot
```

```
[root@host-7-65 ~]# df -H | grep -vE '^Filesystem|tmpfs|cdrom' > output
[root@host-7-65 ~]# cat output
/dev/mapper/centos_lvm-root  7.0G  4.2G  2.8G  61% /
/dev/sda1                 521M  140M  381M  27% /boot
```

Elimin liniile care contin …

```
[root@host-7-65 ~]# df -H | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 " " $1 }' >  output
[root@host-7-65 ~]# cat output
61% /dev/mapper/centos_lvm-root
27% /dev/sda1
```

Afisez doar coloana 5 si 1

```
[root@host-1-81 ~]# fdisk -l

Disk /dev/sda: 7516 MB, 7516192768 bytes, 14680064 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000cd7f9

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048     1026047      512000   83  Linux
/dev/sda2         1026048    14680063     6827008   8e  Linux LVM

Disk /dev/sdb: 2155 MB, 2155023360 bytes, 4209030 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0xe79dc10f

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1            2048     4209029     2103491   82  Linux swap / Solaris

Disk /dev/mapper/centos_lvm-root: 6987 MB, 6987710464 bytes, 13647872 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Primul disc sda

Al doilea disc sdb

LVM

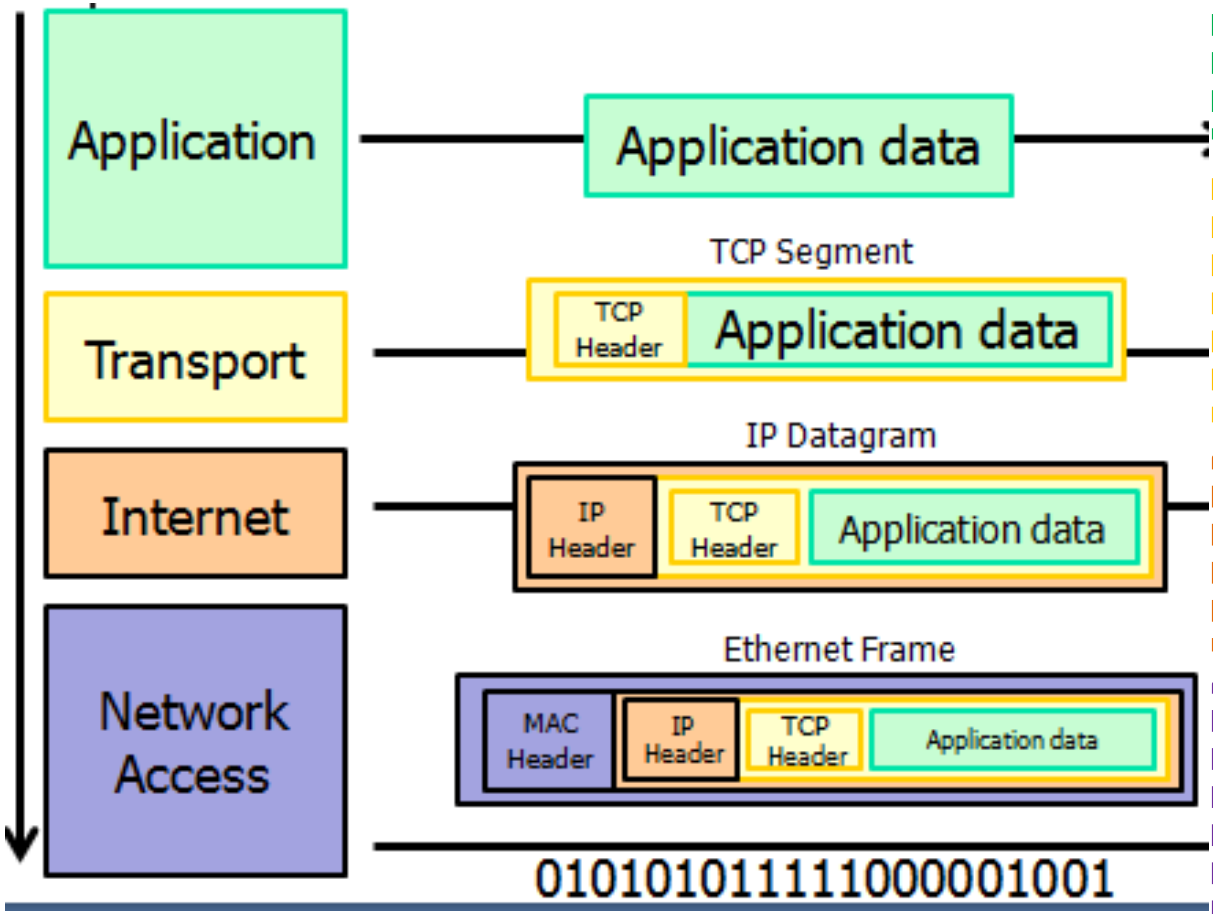
Sectiunea 2
Conexiunea in retea

CUPRINS

- Retele de calculatoare
  - TCP/IP
  - Nivelul 1 – Cablu de retea / Placa de retea
  - Nivelul 2 – Internet
  - Nivelul 3 - Transport

SISTEME DE OPERARE
RETELE DE CALCULATOARE

TCP/IP



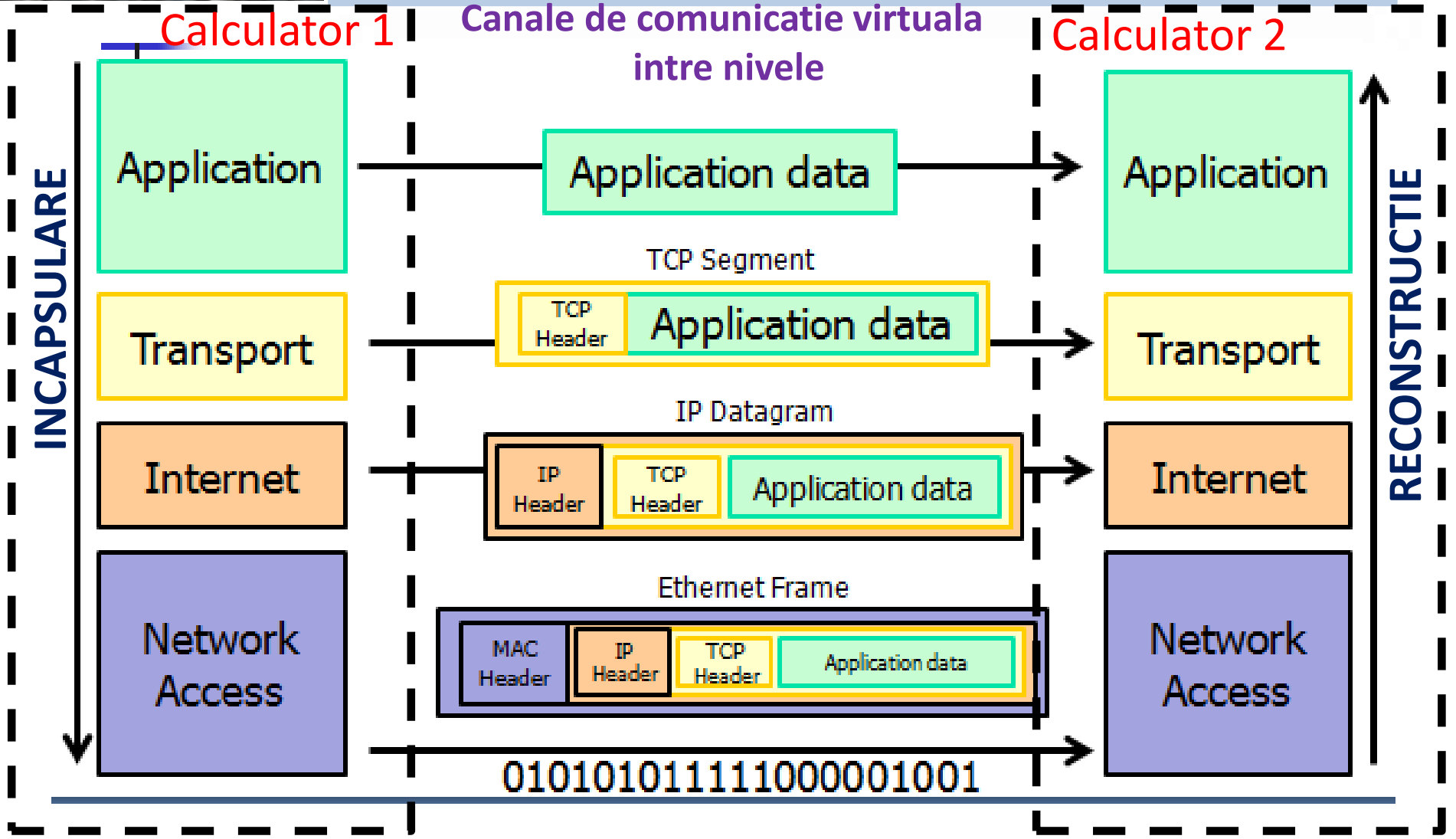
Nivelul aplicatie genereaza datele ce vor fi transmise

Nivelul transport defineste protocolul folosit pentru transmisie – Identificare: port TCP.

Nivelul internet gestioneaza comunicatia prin internet – Identificare: adresa IP

Nivelul acces retea gestioneaza comunicatia prin tipul de media: cablu, fibra, unde. - Identificare: adresa MAC

TCP/IP


Calculator 1
Canale de comunicatie virtuala intre nivele
Calculator 2
INCAPSULARE
RECONSTRUCTIE
Application
Application data
Application
TCP Segment
Transport
TCP Header
Application data
Transport
IP Datagram
Internet
IP Header
TCP Header
Application data
Internet
Ethernet Frame
Network Access
MAC Header
IP Header
TCP Header
Application data
Network Access
0101010111110000001001

NIVELUL 1 – ACCES RETEA

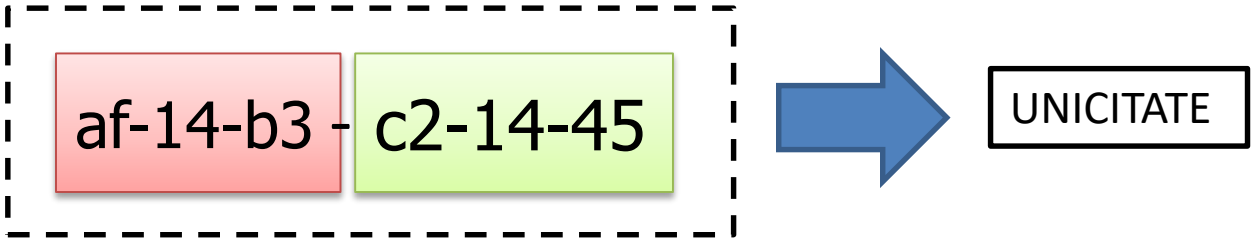
NIVEL 1 – Acces retea

- Nivelul acces retea se imparte in:
  - Nivel fizic
  - Legatura de date

- Tipuri de medii de transmisie
  - Perechi rasucite (Twisted Pair) – format din 8 fire de Cupru
  - Cablu Coaxial – format dintr-un conductor de Cupru
  - Fibra optica
  - Wi -Fi

MAC - Media Access Control

Fiecare adresa MAC este formata din doua parti:



- Prima jumatate a adresei (af-14-b3) alcatuieste **OUI - Organizationally Unique Identifier**. Aceast cod este unic pentru fiecare producator de placi de retea.
- A doua jumatate denumita "**Device ID**" reprezinta un cod unic alocat de catre producator .

NIC - Network Interface Card

- Adresa de MAC este continuta intr-un cip pe placa de retea;

- Componente fizice:
  - Circuitul Rx(receive)
  - Circuitul Tx (transmit)
  - Ethernet Controller
- Este cunoscut si sub numele de LAN adapter.

NIVELUL 2 – INTERNET

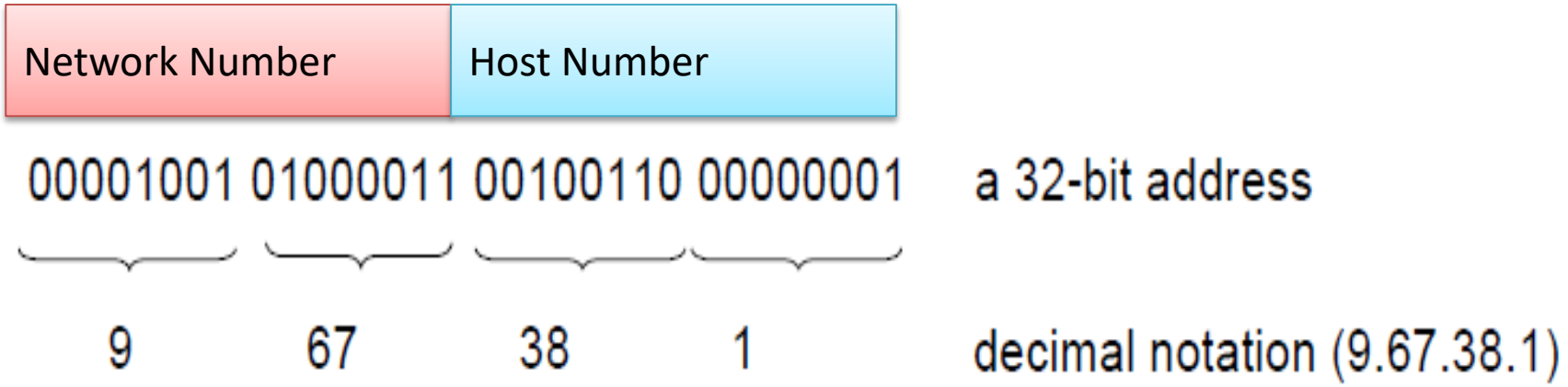
- ADRESA IP
- CONFIGURARE RETEA
- DIAGNOSTICARE RETEA

Adresa IP

Adresele de IP sunt utilizate in cadrul protocolului IP (Internet Protocol) pentru a identifica o interfata.

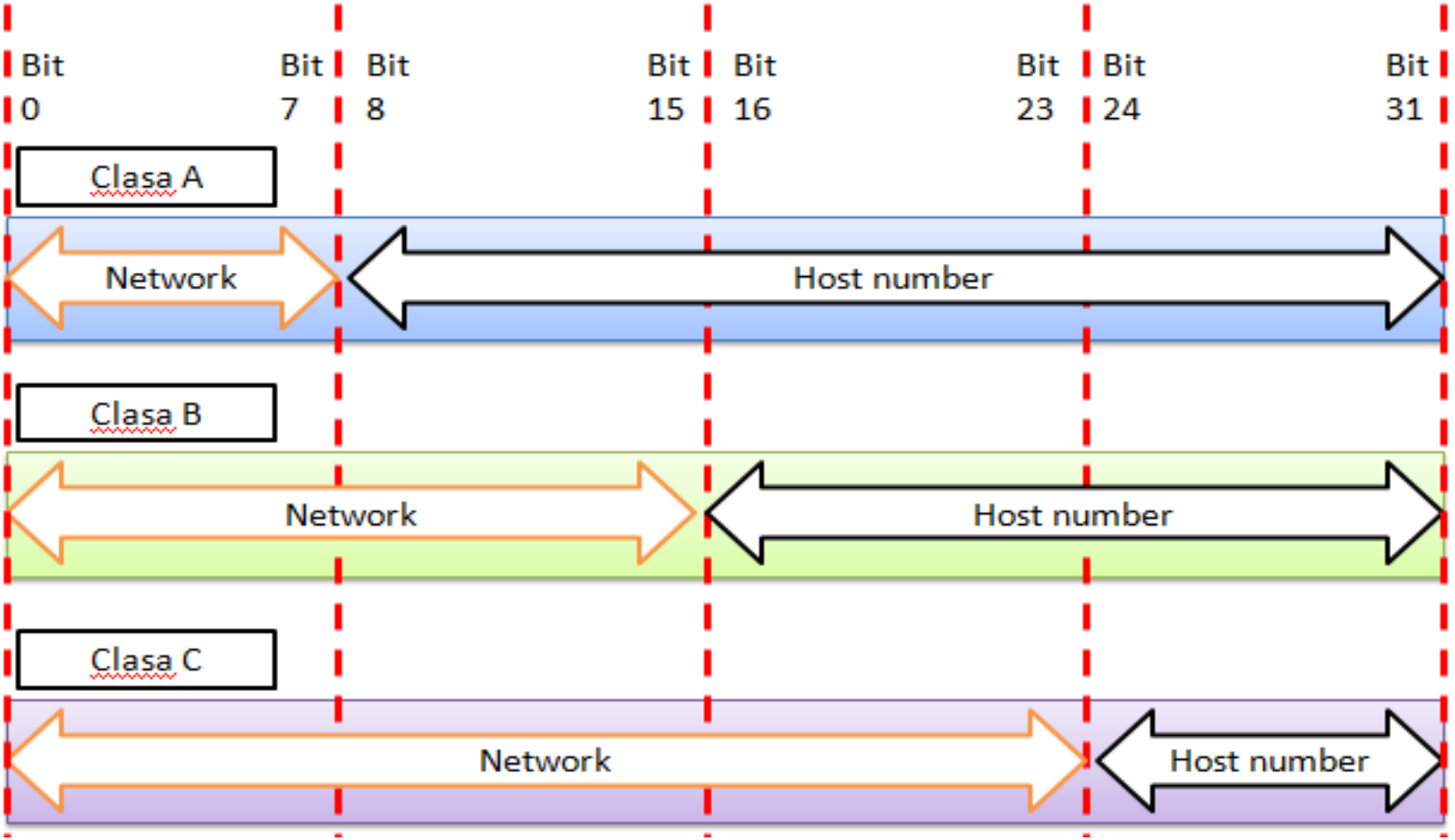
Adresele IP se reprezinta pe 32-biti.

- Formata din 2 parti:
  - o parte care denumeste reteaua (network number)
  - o parte care denumeste statiile de lucru (host)



Clase de adrese

A se vedea materiale de curs retele 1



Adrese IP rezervate

- Adrese IP cu acelasi “network number” formează o retea.
  - **prima adresa IP dintr-o retea este adresa retelei (network address)** (se foloseste pentru a denumi intreaga retea)
  - **ultima adresa IP dintr-o retea reprezinta adresa de broadcast** (se foloseste pentru a comunica un mesaj catre toate calculatoarele din retea).
- **Loopback**: Reteaua de Clasa A 127.0.0.0.
  - Aceasta adresa este atasata interfetelor care proceseaza date in sistemul local, interfata virtuala care simuleaza accesul la retea.
  - Aceasta interfata nu are acces la nivelul fizic de retea.

- Adrese rezervate (private) folosite pentru retele locale fara a asigura accesul la internet:
  - De la 10.0.0.0 la 10.255.255.255 (10.0.0.0/8)
  - De la 172.16.0.0 la 172.31.255.255 (172.16.0.0/14)
  - De la 192.168.0.0 la 192.168.255.255 (192.168.0.0/16)

Adrese IP rezervate

Exemplu:

- Pentru o adresa de clasa C: **192.168.0.0**
  - 192.168.0 reprezinta network number (primii 24 biti) iar 0 reprezinta host number (ultimii 8 biti)
  - **192.168.0.0** este adresa retelei (network address)
  - **192.168.0.1 - 192.168.0.254** reprezinta adrese utile (adrese host), ce pot fi alocate
  - de obicei prima sau ultima adresa utila este alocata pentru “default gateway”
  - **192.168.0.255** reprezinta adresa broadcast

COMENZI:

CONFIGURARE ACCES RETEA

DIAGNOSTICARE CONEXIUNE RETEA

Configurare acces la retea

- In Linux configurarea accesului la retea se poate face, uzual, in 3 moduri:
  - configurare completand fisiere de configurare
  - configurare folosind comenzi shell
  - configurare folosind interfata vizuala

CONFIGURARE ACCES LA RETEA:

CONFIGURARE EDITAND FISIERE DE CONFIGURARE

Configurare acces la retea folosind fisiere - slide 1

- Configurarea parametrilor de retea
  - se poate face folosind scripturile continute in fisierele de mai jos
  - locatia si structura fiserelor difera in functie de distributie;
  - exemplele de mai departe sunt prezentate pentru
    - **1. SUSE** + RedHat/CentOS/Fedora
    - 2. Ubuntu/Debian/Linux Mint

Configurare acces la retea folosind fisiere - slide 1

- Fisierele se gasesc in directorul
  - /etc/sysconfig/network
- Pentru configurare interfetelor de tip ethernet se folosesc fiserele:
  - /etc/sysconfig/network-scripts/ifcfg-eth*
  - /etc/sysconfig/network-scripts/ifcfg.template
- Pentru configurare protocol DHCP:
  - /etc/sysconfig/network/dhcp

* reprezinta numarul interfetei: 0,1,2 …

fisierul poate fi folosit ca model pentru generarea fisierelor de mai sus

root@host-6-1:/etc/sysconfig/network-scripts

```
[root@host-6-1 sysconfig]# cd /etc/sysconfig/network
-bash: cd: /etc/sysconfig/network: Not a directory
[root@host-6-1 sysconfig]# cd /etc/sysconfig/network-scripts/
[root@host-6-1 network-scripts]# ls
ifcfg-ens3    ifdown-ppp       ifup-eth     ifup-sit
ifcfg-lo      ifdown-routes    ifup-ippp    ifup-Team
ifdown        ifdown-sit       ifup-ipv6    ifup-TeamPort
ifdown-bnep   ifdown-Team      ifup-isdn    ifup-tunnel
ifdown-eth    ifdown-TeamPort  ifup-plip    ifup-wireless
```

Configurare acces la retea folosind fisiere – slide 2

- Configurare fisier ifcfg-eth*

```
BOOTPROTO='static'
BROADCAST=''
ETHTOOL_OPTIONS=''
IPADDR='10.10.1.2'
MTU=''
NAME='Broadcom NetXtreme II BCM5708 Gigabit Ethernet'
NETMASK='255.255.255.0'
NETWORK=''
REMOTE_IPADDR=''
STARTMODE='auto'
UNIQUE='JNkJ.swy66v437tA'
USERCONTROL='no'
_nm_name='bus-pci-0000:03:00.0'
```

Tip protocol:
static
dhcp
none

Adresa IP

Masca retea

MTU
maximum
transmission unit
Ethernet = 1500

Configurare acces la retea folosind fisiere - slide 3

- Semnificatii ale campurilor din fisierul ifcfg-eth*
- BOOTPROTO=<*protocol*>
  - *Variante:* none — static ; dhcp —protocol DHCP ....
- BROADCAST=<*address*>,
  - <*address*> = adresa broadcast .
- DEVICE=<*name*>,
  - <*name*> = numele dispozitivului.
- IPADDR=<*address*>,
  - <*address*> = adresa IP.
- NETMASK=<*mask*>,
  - <*mask*> = masca de retea.
- NETWORK=<*address*>,
  - <*address*> = adresa de retea.
- ONBOOT=<*answer*>,
  - <*answer*> = yes —activ la boot / no
- USERCTL=<*answer*>,
  - <*answer*> = true — toti utilizatorii pot controla dispozivul / false — doar root

Configurarea retelei folosind fisiere

root@host-6-1:/etc/sysconfig/network-scripts

```
 GNU nano 2.3.1                File: ifcfg-ens3

HWADDR="00:0C:11:00:13:90"
TYPE="Ethernet"
BOOTPROTO="dhcp"
DEFROUTE="yes"
PEERDNS="yes"
PEERROUTES="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_PEERDNS="yes"
IPV6_PEERROUTES="yes"
IPV6_FAILURE_FATAL="no"
NAME="ens3"
UUID="71697b55-cf9d-42c6-ba4d-bb0c01ae34d9"
ONBOOT="yes"

                        [ Read 17 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

Exemplu:
Adresa preluata prin protocolul DHCP

Configurarea retelei folosind fisiere

- Exemplu:
  - In prompt se deschide fisierul de configurare folosind un editor de text:
    - Exemplu: nano /etc/sysconfig/network-scripts/ifcfg-eth0
  - Se configureaza conform exemplului:

DEVICE="eth0"
BOOTPROTO=static
ONBOOT=yes
TYPE="Ethernet"
IPADDR=192.168.50.2
NAME="System eth0"
HWADDR=00:0C:29:28:FD:4C
GATEWAY=192.168.50.1

Configurare acces la retea folosind fisiere

- Pentru configurare default gateway
  - /etc/sysconfig/network/routes
  - Se introduce o linie de tipul: “default 10.10.1.1”

- Echipamentele de nivel 1 (Media Access + Data Link)
  - folosesc pentru transmiterea (gasirea destinatarului) pachetelor doar adresa MAC
  - Exemplu: switch
- Echipamentele de nivel 2 (IP)
  - folosesc pentru transmiterea (gasirea destinatarului) pachetelor doar adresa IP
  - Exemplu: router

Configurare acces la retea folosind fisiere - slide 1

Pentru UBUNTU

- Cale fisier configurare:
  - /etc/network/interfaces
- Model de configurare:

```
auto eth0
iface eth0 inet static
address 192.168.50.2
netmask 255.255.255.0
gateway 192.168.50.1
```

CONFIGURARE ACCES LA RETEA:

CONFIGURARE ACCES LA RETEA FOLOSIND COMENZI

Configurare acces la retea folosind comenzi

Comanda **ifconfig**

- Comanda **ifconfig**:
  - salveaza configurarea temporar
  - Exemple:
    - ifconfig
    - ifconfig -a
    - ifconfig eth0
    - ifconfig eth0 up
    - ifconfig eth0 down
    - ifconfig eth0 192.168.2.2 netmask 255.255.255.0

Afiseaza informatii despre interfete, respectiv interfata eth0

Porneste si opreste interfata eth0

Configureaza interfata eth0 cu adresa 192.168.2.2 si masca de retea 255.255.255.0

Configurare acces la retea folosind comenzi

Comanda ifconfig

```
upbvm063:~ # ifconfig
eth0      Link encap:Ethernet  HWaddr 00:50:56
          inet addr:10.160.0.92  Bcast:10.160.255.255  Mask:255.255.0.0
          inet6 addr: fe80::250:56ff:fe00:9622/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3129878 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:266263361 (253.9 Mb)  TX bytes:578 (578.0 b)
          Interrupt:19 Base address:0x2000

eth1      Link encap:Ethernet  HWaddr 00:50:56:
          inet addr:141.85.204.93  Bcast:141.85.207.255  Mask:255.255.252.0
          inet6 addr: fe80::250:56ff:fe00:9623/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:13704728 errors:0 dropped:0 overruns:0 frame:0
          TX packets:549636 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:959615655 (915.1 Mb)  TX bytes:81602368 (77.8 Mb)
          Interrupt:16 Base address:0x2040

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:100 (100.0 b)  TX bytes:100 (100.0 b)
```

Interfata

Adresa MAC

Masca de retea

Adresa IP

Pachete trimise (TX) si primite (RX)

eth0, eth1 = interfata ethernet 0 si 1
lo = loopback interface

Configurarea retelei folosind comenzi

Comanda ifconfig

root@host-6-1:/etc/sysconfig/network-scripts

```
[root@host-6-1 network-scripts]# ifconfig
ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.6.1  netmask 255.255.255.248  broadcast 10.0.6.7
        inet6 fe80::20c:1ff:fe00:690  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:01:00:06:90  txqueuelen 1000  (Ethernet)
        RX packets 1347  bytes 138768 (135.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 904  bytes 174334 (170.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 0  (Local Loopback)
        RX packets 6  bytes 560 (560.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6  bytes 560 (560.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@host-6-1 network-scripts]#
```

Configurare acces la retea folosind comenzi

Comanda **ip**

- Comanda **ip**:
  - comanda folosita la configurarea permanaenta a conexiunii la retea
    - ip addr show
    - ip a
    - ip addr add 10.1.1.2/24 dev eth0
    - ip route add default via 192.168.1.254
  - ifup / ifdown – comenzi folosite pentru pornirea / oprirea interfetelor

Afisaza adresele ip

Afisaza configuratia de retea

Configureaza adresa 10.1.1.2 cu masca 255.255.255.0 (/24)

Stabilirea unei rute de tip “default gateway”

Configurare acces la retea folosind comenzi

Comanda ip

```
upbvm063:~ # ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 brd 127.255.255.255 scope host lo
    inet 127.0.0.2/8 brd 127.255.255.255 scope host secondary lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
    link/ether 00:50:56:          brd ff:ff:ff:ff:ff:ff
    inet 10.160.0.92/16 brd 10.160.255.255 scope global eth0
    inet6 fe80::250:56ff:fe00:9622/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN qlen 1000
    link/ether 00:50:56:          brd ff:ff:ff:ff:ff:ff
    inet 141.85.204.93/22 brd 141.85.207.255 scope global eth1
    inet6 fe80::250:56ff:fe00:9623/64 scope link
       valid_lft forever preferred_lft forever
```

Interfata

Adresa MAC

Masca de retea

Adresa IP

MTU = Maximum Transition Unit
- dimensiunea cea mai mare a segmentului de date ce poate fi transferat prin protocolul ethernet

Sintaxa: ip a

Configurare acces la retea folosind comenzi

Comanda **ip**

root@host-6-1:/etc/sysconfig/network-scripts

```
[root@host-6-1 network-scripts]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    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: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP ql
en 1000
    link/ether 00:0c:01:00:06:90 brd ff:ff:ff:ff:ff:ff
    inet 10.0.6.1/29 brd 10.0.6.7 scope global dynamic ens3
       valid_lft 860579sec preferred_lft 860579sec
    inet6 fe80::20c:1ff:fe00:690/64 scope link
       valid_lft forever preferred_lft forever
[root@host-6-1 network-scripts]#
```

Configurare acces la retea folosind comenzi - exemple

```
upbvm063:~ # ip route
141.85.204.0/22 dev eth1  proto kernel  scope link  src 141.85.204.93
10.160.0.0/16 dev eth0  scope link
169.254.0.0/16 dev eth0  scope link
127.0.0.0/8 dev lo  scope link
default via 141.85.204.1 dev eth1
```

Stabilirea unei rute de tip “default gateway”

Pasi configurare acces retea folosind comenzi

Se vor configura urmatoarele:
IP: 192.168.0.2
Subnet Mask: 255.255.255.0
Gateway: 192.168.0.1

- Pasi configurare:
  - Pas 0: Identificare placa retea
    - ifconfig
    - ip a

Configurare acces la retea folosind comenzi

  - Pas 1: Configurare adresa IP
    - ifconfig eth0 192.168.0.2 netmask 255.255.255.0
    - SAU
    - ip addr add 192.168.0.2 /24 dev eth0
  - Pas 2: Configurare gateway
    - ip route add default via 192.168.0.1

CONFIGURARE ACCES LA RETEA:

DIAGNOSTICAREA CONEXIUNII

Diagnosticarea conexiunii

- Comanda ping determina existenta unei conexiuni si ofera informatii despre calitatea conexiunii
  - ping 192.168.0.1
- Comanda traceroute determina numarul de noduri prin care va trece un pachet pentru a ajunge de la sursa la destinatie
  - traceroute www.uso.aii.pub.ro
  - traceroute 192.168.0.1
- Comanda whois afisaza informatii despre o adresa de ip publica (proprietar, locatie)
  - whois 192.168.0.1
- Comanda arp afisaza tabela arp (corespondenta adresa IP - adresa MAC)
  - arp -a

Diagnosticarea conexiunii comanda: ping

```
upbvm063:~ # ping 86.55.202.1
PING 86.55.202.1 (86.55.202.1) 56(84) bytes of data.
From 130.117.1.109: icmp_seq=1 Time to live exceeded
From 130.117.1.109 icmp_seq=1 Time to live exceeded
From 130.117.1.109 icmp_seq=2 Time to live exceeded
From 130.117.1.117 icmp_seq=3 Time to live exceeded
From 130.117.1.117 icmp_seq=4 Time to live exceeded
From 130.117.1.117 icmp_seq=5 Time to live exceeded
^XFrom 130.117.1.109 icmp_seq=6 Time to live exceeded
^C
--- 86.55.202.1 ping statistics ---
7 packets transmitted, 0 received, +7 errors, 100% packet loss, time 6021ms

upbvm063:~ # ping 141.85.204.1
PING 141.85.204.1 (141.85.204.1) 56(84) bytes of data.
64 bytes from 141.85.204.1: icmp_seq=1 ttl=255 time=1.98 ms
64 bytes from 141.85.204.1: icmp_seq=2 ttl=255 time=1.98 ms
64 bytes from 141.85.204.1: icmp_seq=3 ttl=255 time=1.99 ms
64 bytes from 141.85.204.1: icmp_seq=4 ttl=255 time=1.97 ms
64 bytes from 141.85.204.1: icmp_seq=5 ttl=255 time=1.96 ms
64 bytes from 141.85.204.1: icmp_seq=6 ttl=255 time=2.03 ms
^C
--- 141.85.204.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5017ms
rtt min/avg/max/mdev = 1.963/1.990/2.035/0.056 ms
```

TTL = Time To Live
- numarul de noduri prin care poate sa treaca un pachet pana sa fie sters

Lipsa conexiune

timp de raspuns – masura a calitatii conexiunii

numar de pachete primite ca raspuns – masura a calitatii conexiunii

Conexiune buna

Diagnosticarea conexiunii comanda: ping

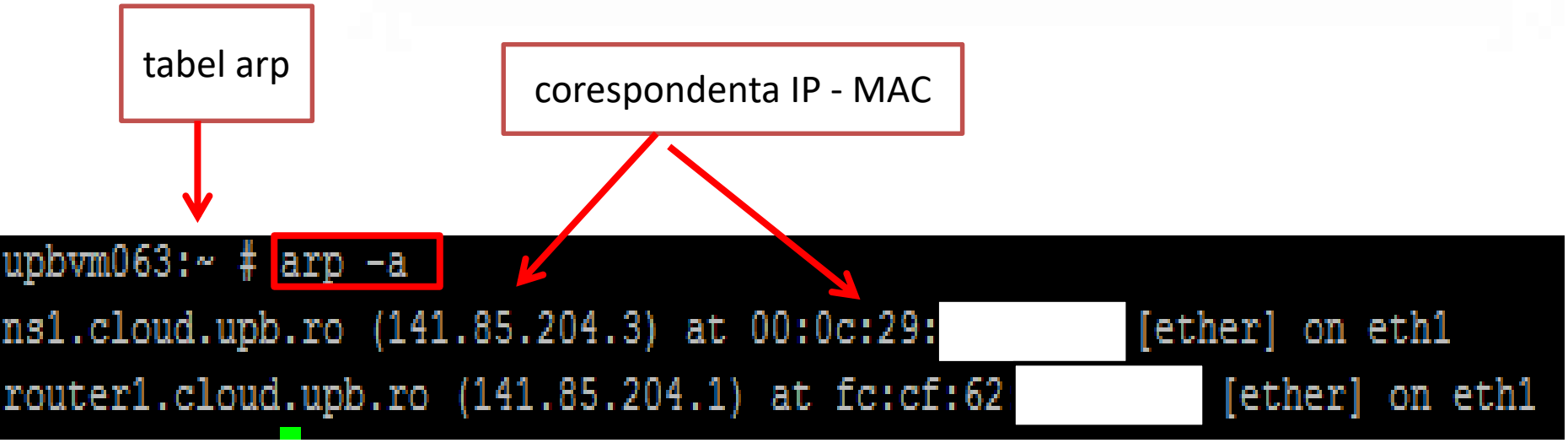
```
root@host-6-1:/etc/sysconfig/network-scripts
[root@host-6-1 network-scripts]# ping www.acs.pub.ro
PING www.acs.pub.ro (141.85.227.151) 56(84) bytes of data.
^C
--- www.acs.pub.ro ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 5999ms

[root@host-6-1 network-scripts]# ping localhost
PING localhost (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.043 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.046 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.063 ms
^C
--- localhost ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.043/0.050/0.063/0.012 ms
[root@host-6-1 network-scripts]# traceroute www.acs.pub.ro
traceroute to www.acs.pub.ro (141.85.227.151), 30 hops max, 60 byte packets
 1  host-6-6.linuxzoo.net (10.0.6.6)  0.136 ms  0.102 ms  3.011 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  *^C
```

Nu raspunde la comanda ping.

Raspunde la comanda ping.

Raspuns la comanda traceroute.

Diagnosticarea conexiunii comanda: arp



- **Comanda arp**
  - editeaza sau afișează memoria cache alocata pentru stocarea datelor despre nodurile vecine dintr-o rețea IPv4.

Diagnosticare conexiune retea comanda: traceroute

```
upbvm063:~ # traceroute www.google.ro
traceroute to www.google.ro (173.194.113.119), 30 hops max, 40 byte packets using UDP
 1  router1.cloud.upb.ro (141.85.204.1)  1.943 ms   1.915 ms   1.890 ms
 2  r-bb5-e0.Bucharest.roedu.net (141.85.254.16)  15.489 ms   9.825 ms   1.934 ms
 3  37.128.225.193 (37.128.225.193)  0.437 ms   0.443 ms   0.590 ms
 4  vl-4000.core1.buc.roedu.net (37.128.232.137)  1.175 ms   0.866 ms   0.894 ms
 5  te-3-4.core2.nat.roedu.net (37.128.239.49)  0.976 ms   0.836 ms   0.952 ms
 6  te-4-3.br1.nat.roedu.net (37.128.239.5)  16.968 ms * *
 7  roedunet.rt1.buc.ro.geant.net (62.40.125.137)  0.804 ms   0.770 ms   0.745 ms
 8  ae4.mx1.bud.hu.geant.net (62.40.112.193)  24.019 ms   24.048 ms   24.027 ms
 9  ae0.mx2.bra.sk.geant.net (62.40.98.111)  26.648 ms   26.618 ms   26.631 ms
10  ae1.mx1.pra.cz.geant.net (62.40.98.54)  30.773 ms   30.762 ms   30.742 ms
11  ae2.mx1.fra.de.geant.net (62.40.98.53)  41.231 ms   38.596 ms   37.191 ms
12  ae4.rt1.fra.de.geant.net (62.40.98.135)  37.309 ms   37.176 ms   37.177 ms
13  google-gw.rt1.fra.de.geant.net (62.40.125.202)  37.272 ms   37.268 ms   37.253 ms
14  209.85.240.64 (209.85.240.64)  37.774 ms   39.843 ms   37.852 ms
15  209.85.243.233 (209.85.243.233)  39.741 ms   37.996 ms   37.993 ms
16  fra02s22-in-f23.1e100.net (173.194.113.119)  37.698 ms   37.683 ms   37.704 ms
```

nod administrat de Internet Service Provider: ROEDU

numarul de noduri de la sursa la www.google.ro

detalii despre noduri si timpul de raspuns

Diagnosticare conexiune retea comanda:netstat

- Comenzi
  - Comanda netstat
    - netstat –i
    - netsat –s

root@host-6-1:/etc/sysconfig/network-scripts

```
[root@host-6-1 network-scripts]# netstat -i
Kernel Interface table
Iface       MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
ens3       1500     1952      0      0 0           1343      0      0      0 BMRU
lo        65536       12      0      0 0             12      0      0      0 LRU
[root@host-6-1 network-scripts]# netstat -s
Ip:
    1949 total packets received
    0 forwarded
    0 incoming packets discarded
    1932 incoming packets delivered
    1295 requests sent out
    32 dropped because of missing route
Icmp:
    9 ICMP messages received
    0 input ICMP message failed.
    ICMP input histogram:
        timeout in transit: 3
        echo requests: 3
        echo replies: 3
    32 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        echo request: 29
```

Utilitare retea

- Comanda netstat
  - netstat –i
  - netsat –s
  - netstat – v
  - netsata –c
  - netstat –ta
  - netsat proto tcp
  - netsata --tcp --numeric

Configurare router

- Sistemul de Operare Linux poate fi configurat ca router (packet forward)
- Pentru configurare:
  - Se poate modifica un paramatru cu comanda:
    - echo 1 > /proc/sys/net/ipv4/ip_forward
  - Se poate modifica un fisier:
    - /etc/sysconfig/sysctl
    - se modifica linia dupa cum urmeaza IP_FORWARD=yes
- Masquerade – traducerea de adrese
  - NAT (Network Address Translation)
  - traduce o adresa publica intr-o adresa privata

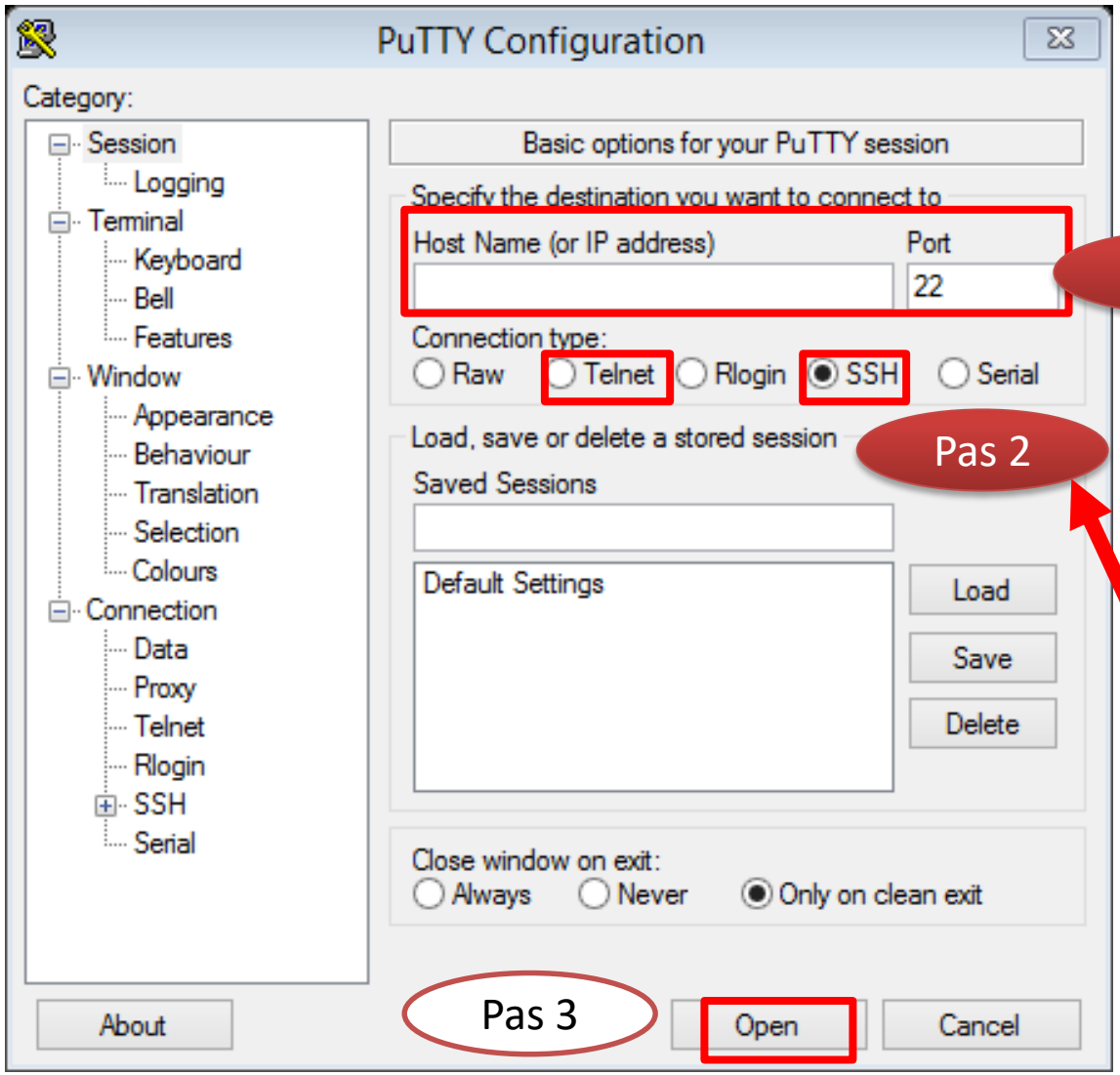
LINUX
NIVEL 3 – TRANSPORT
PROTOCOALE DE CONECTARE LA DISTANTA

Protocoale de conectare la distanta

- SSH — Secure Shell
  - Stabileste o conexiune criptata intre calculatorul sursa si cel destinatie pentru accesarea interpretorului de comenzi
  - Exemplu: ssh 192.168.0.3
- telnet
  - Stabileste o conexiune intre calculatorul sursa si cel destinatie pentru accesarea interpretorului de comenzi. Conexiunea nu este criptata
  - Exemplu: telnet 192.168.0.3

Utilitar PuTTY

Utilitar pentru conexiunea la distanta prin telnet sau SSH pentru SO windows



Se introduce adresa de IP sau adresa URL si se selecteaza portul

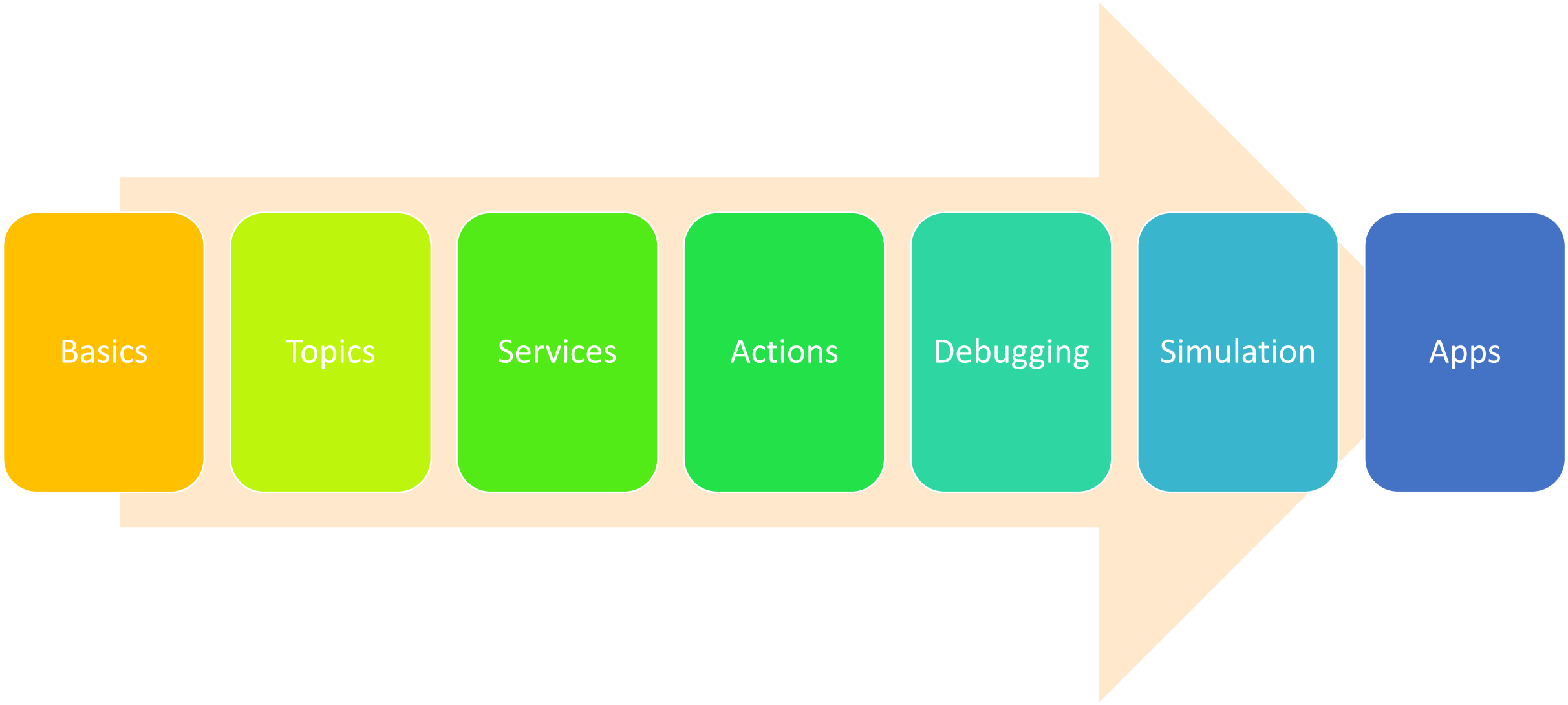
Se selecteaza SSH sau Telnet

Laborator

- ROS Noetic
  - nano ~/.bashrc
- sudo apt install turtlebot3 / turtlebot3-simulations
  - export TURTLEBOT3_MODEL=burger
- roslaunch turtlebot3_teleop

K2. Basics

Deconstruction



Basics - checkpoint

! roslaunch turtlebot3_gazebo turtlebot3_stage_4.launch

- roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch

- roslaunch <package_name> <launch_file>

Basics - packages

- launch folder: Contains launch files
- src folder: Source files (cpp, python)
- CMakeLists.txt: List of cmake rules for compilation
- package.xml: Package information and dependencies

- roscd <package_name>
- ls

Basics – launch files

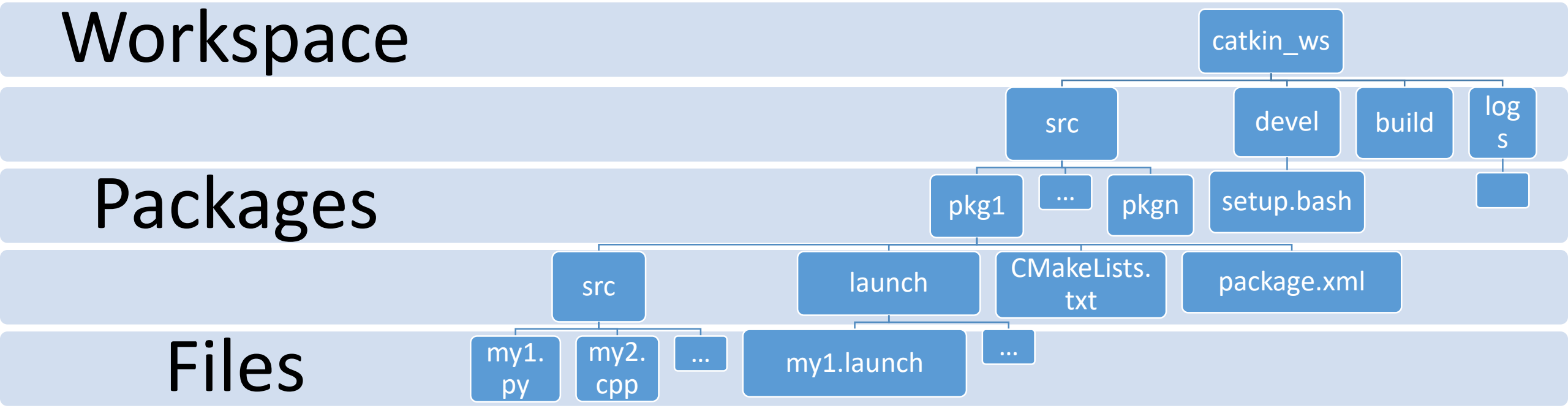
- roscd turtlebot3_teleop
- cd launch
- cat keyboard3_teleop_key.launch

```
<launch>
<arg name="model" default="$(env TURTLEBOT3_MODEL)" doc="model type [burger, waffle, waffle_pi]"/>
<param name="model" value="$(arg model)"/>
<!-- turtlebot3_teleop_key already has its own built in velocity smoother -->
<node pkg="turtlebot3_teleop" type="turtlebot3_teleop_key" name="turtlebot3_teleop_keyboard"  output="screen">
</node>
</launch>
```

```
<node pkg="turtlebot3_teleop" type="turtlebot3_teleop_key"
name="turtlebot3_teleop_keyboard"  output="screen"> </node>
```

- pkg="package_name" # Name of the package that contains the code of the ROS program to execute
- type="python_file_name.py" # Name of the program file that we want to execute
- name="node_name" # Name of the ROS node that will launch our Python file
- output="type_of_output" # Through which channel you will print the output of the Python file

Basics – workspaces



Basics - create

- roscd && pwd

- cd ~
- mkdir catkin_ws
- mkdir src && cd src
- catkin_create_pkg <package_name> <package_dependecies>
- catkin_create_pkg my_package rospy

Basics - build

- catkin init
- python3-catkin-tools? build-essential?
- catkin clean
- catkin build

- nano ~/.bashrc
- source correctly

- https://robotics.stackexchange.com/questions/16604/ros-catkin-make-vs-catkin-build

- rospack list
- rospack list | grep my_package
- roscd my_package
- rospack profile

- cd src
- touch simple.py && nano simple.py

```
#! /usr/bin/env python
# This line specifies the interpreter used , every Python file needs to start with this line at the top.
import rospy  # Import rospy, which is a Python library for ROS
rospy.init_node(‘Friends') # Initiate a node called ObiWan
print(“We were on a break!!") # A simple Python print
```

- rosrun my_package simple.py
- chmod? / ls -la

- roscd my_package
- mkdir launch
- touch launch/my_launch_file.launch
- nano launch/my_launch_file.launch
- write launch file

- roslaunch my_package my_launch_file.launch

Basics - nodes

- rosnode list
- simple.py:

```
#! /usr/bin/env python
import rospy
rospy.init_node(“Friends")
rate = rospy.Rate(2)              # We create a Rate object of 2Hz
while not rospy.is_shutdown():    # Endless loop until Ctrl + C
   print(“How you doooin?")
   rate.sleep()                   # We sleep the needed time to maintain the Rate fixed above
# This program creates an endless loop that repeats itself 2 times per second (2Hz) until somebody presses Ctrl + C in the Shell
```

- roslaunch my_package my_launch_file.launch

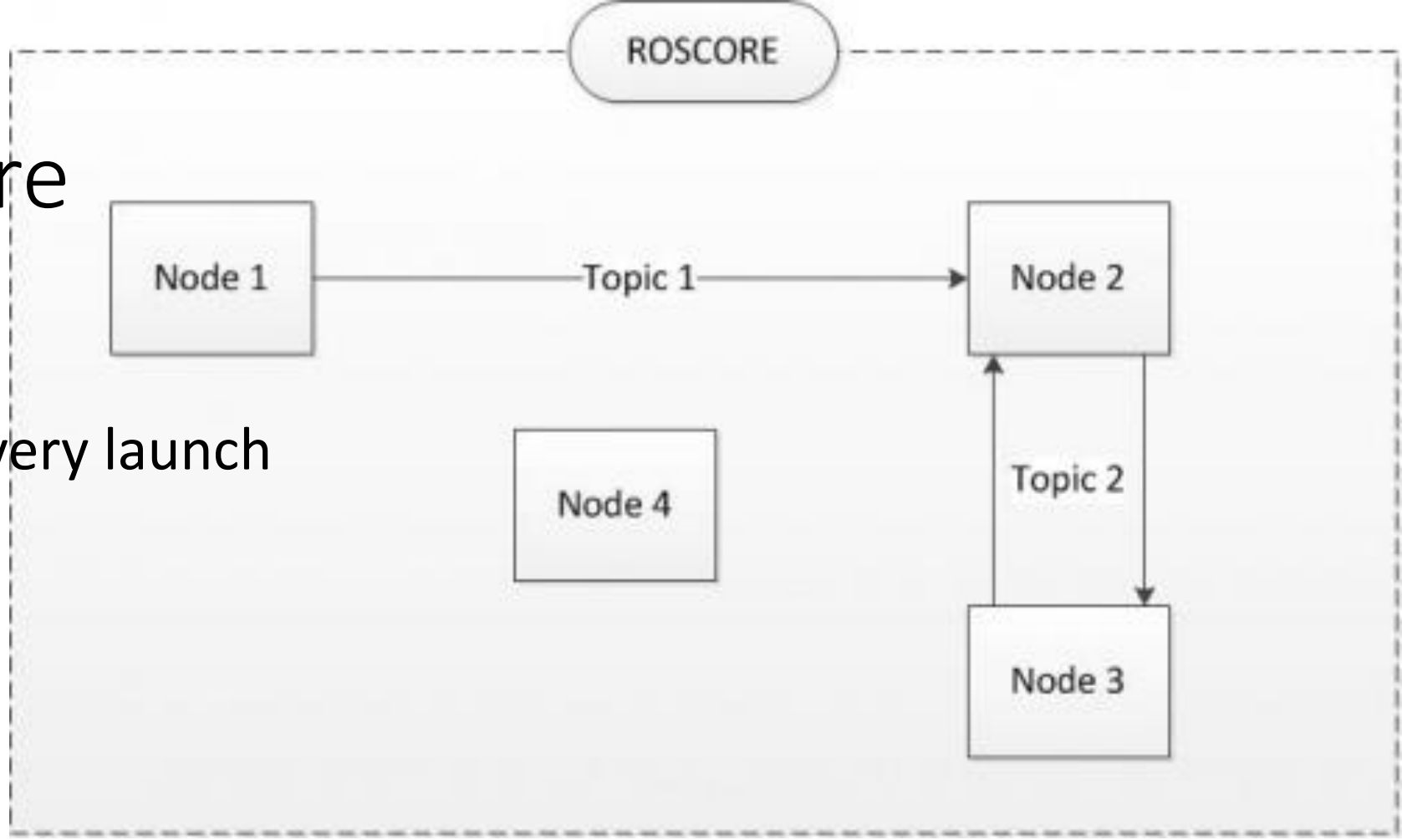
Basics

- rosnode list
- rosnode info <node_name>
- rosnode info /Friends

- rosparam list
- rosparam get <parameter_name>
- rosparam set <parameter_name> <value>

Basics - ROScore

- roscore
- automatically with every launch



- export | grep ROS

K3. Git

Git - General

- Version control software (VCS)
- Manage projects with **Repositories**
- **Clone** a project to work on a local copy
- Control and track changes with **Staging** and **Committing**
- **Branch** and **Merge** to allow for work on different parts and versions of a project
- **Pull** the latest version of the project to a local copy
- **Push** local updates to the main project

Git - Use

- Initialize Git on a folder, making it a **Repository**
- Git now creates a hidden folder to keep track of changes in that folder
- When a file is changed, added or deleted, it is considered **modified**
- You select the modified files you want to **Stage**
- The **Staged** files are **Committed**, which prompts Git to store a **permanent** snapshot of the files
- Git allows you to see the full history of every commit.
- You can revert back to any previous commit.
- Git does not store a separate copy of every file in every commit, but keeps track of changes made in each commit!

Git - Github

- Git is not the same as GitHub.
- GitHub makes tools that use Git.
- Alternatives: Bitbucket, GitLab

Git - Configure

- sudo apt install git
- git config --global user.name “Me_of_course"
- git config --global user.email bb8@ros.org
- git config --global core.editor <nano / gedit / vim>

- GitHub account + Team repo
- readme.md

Git – Working with Repos

- git commit
- git commit –m “…”

- git push <remote>

- git merge
  - conflicts?
- git pull <remote>
  - pull = fetch + merge
  - always at start

Git - Commands

- git status
- staged: git add .
- git log
  - --pretty=oneline
  - --since=2.weeks
  - --graph
- git diff
- git help --all

Git - Branches

- git branch <new_direction>
- git checkout <some_branch_or_commit>
  - -b creates new branch
- git status

Git - Contribute

- fork a repo
- git clone <repo> <local name>
- pull requests

Git - Advanced

- .gitignore
- GitHub SSH instructions

- git reset <commit>
- git stash
- git commit --amend

K3. Topics

Publishers

Topics

- Publishers
- Messages
- Subscribers

Topics - Publishers

- cd ~/catkin_ws/src
- catkin_create_pkg topics_pkg rospy std_msgs
- cd topics_pkg/src
- touch simple_topic_publisher.py
- chmod +x simple_topic_publisher.py
- nano simple_topics_publisher.py
- cd ~/catkin_ws
- catkin build
- source devel/setup.bash

Topics - Publishers

- simple_topics_publisher.py

```
#! /usr/bin/env python

import rospy                                                    # Import the Python library for ROS
from std_msgs.msg import Int32                                  # Import the Int32 message from the std_msgs package

rospy.init_node('topic_publisher')                              # Initiate a Node named 'topic_publisher'
pub = rospy.Publisher('/counter', Int32, queue_size=1)          # Create a Publisher object, that will publish on the /counter topic
                                                                # messages of type Int32

rate = rospy.Rate(2)                                            # Set a publish rate of 2 Hz
count = Int32()                                                 # Create a var of type Int32
count.data = 0                                                  # Initialize 'count' variable

while not rospy.is_shutdown():                                  # Create a loop that will go until someone stops the program execution
  pub.publish(count)                                            # Publish the message within the 'count' variable
  count.data += 1                                               # Increment 'count' variable
  rate.sleep()                                                  # Make sure the publish rate maintains at 2 Hz
```

Topics - Publishers

- rosrun topics_pkg simple_topic_publisher.py
- rostopic list | grep '/counter‘
- rostopic info /counter
- rostopic echo /counter

- simple_topics_publisher.py #comments

Topics - Publishers

- rostopic list
- rostopic echo <topic_name> -n1
- rostopic info <topic_name>
- rostopic -h

Topics - Messages

- rosmsg show <message>
- rosmsg info <message>
- rosmsg show std_msgs/Int32
- roscd std_msgs/msg/ && ls

- launch file
  - simple_topics_publisher.py
    - topic
    - message

K4. Topics

Messages & Subscribers

Topics - Subscribers

- cd ~/catkin_ws/src/topics_pkg/src
- touch simple_topic_subscriber.py
- chmod +x simple_topic_subscriber.py
- nano simple_topics_subscriber.py
- cd ~/catkin_ws
- catkin build
- source devel/setup.bash

Topics - Subscribers

- simple_topics_publisher.py

```
#! /usr/bin/env python

import rospy
from std_msgs.msg import Int32

def callback(msg):                          # Define a function called 'callback' that receives a parameter named 'msg'
    print (msg.data)                        # Print the value 'data' inside the 'msg' parameter

rospy.init_node('topic_subscriber')                 # Initiate a Node called 'topic_subscriber'
sub = rospy.Subscriber('/counter', Int32, callback)     # Create a Subscriber object that will listen to the /counter
                                            # topic and will call the 'callback' function each time it reads
                                            # something from the topic
rospy.spin()                                    # Create a loop that will keep the program in execution
```

Topics - Subscribers

- ??
- rostopic echo /counter
- rostopic pub <topic_name> <message_type> <value>
- read odometry
  - topic
  - message

Topics - Messages

- Custom Messages -> Age
- roscd topics_pkg
- mkdir msg
- touch msg/Age.msg
- nano msg/Age.msg

float32 years

float32 months

float32 days

Topics - Messages

- CMakeLists.txt
  - find_package() rospy, std_msgs, message_generation
  - add_message_files() Files, Age.msg
  - generate_messages() Dependencies, std_msgs
  - catkin_package() Catkin_Depends rospy message_runtime
- package.xml

<build_depend>message_generation</build_depend>

<build_export_depend>message_runtime</build_export_depend>

<exec_depend>message_runtime</exec_depend>

Topics - Messages

- roscd; cd ..
- catkin build
- source devel/setup.bash

- rosmsg show Age
- rosmsg list | grep Age

- Age publisher

K5. Services

Clients

Services

- Topics Services & Actions
- Services (synchronous) vs Actions (asynchronous)

Simulations

- cd ~/simulations_ws
- mkdir src && cd src
- git clone git@github.com:bianca-ghi/ROS_Sims.git
- catkin init
- catkin build
- nano ~/.bashrc

Services – launch & shell

- roslaunch turtlebot_demo start_demo.launch
- different shells! (and the & operator)
- rosnode list
- rosservice list

- rosservice info /name_of_your_service
- rosservice info /move_in_square

Services – launch & shell

- roscd turtlebot_demo/ launch
- cat start_demo.launch
- <include file="$(find your_pkg)/launch/whatever.launch"/>

- rosservice call /the_service_name TAB-TAB
- rosservice call /move_in_square [TAB]+[TAB]

Services

- cd ~/catkin_ws/src
- catkin_create_pkg services_pkg rospy
- cd services_pkg/src
- touch simple_service_client.py
- chmod +x simple_service_client.py
- gedit simple_service_client.py
- cd ~/catkin_ws
- catkin build
- source ~/.bashrc (why?)

simple_service_client.py

```
#! /usr/bin/env python
import rospy
# Import the service message used by the service /trajectory_by_name
from turtlebot_make_square.srv import MoveInSquare, MoveInSquareRequest
import sys

# Initialise a ROS node with the name service_client
rospy.init_node('service_client')
# Wait for the service client /move_in_square to be running
rospy.wait_for_service('/move_in_square')
# Create the connection to the service
move_in_square_service = rospy.ServiceProxy('/move_in_square', MoveInSqare)
# Create an object of type MoveInSquareRequest
move_in_square_object = MoveInSquareRequest()

# Send through the connection the name of the request
result = move_in_square_service(move_in_square_object)
# Print the result given by the service called
print(result)
```

Services - clients

- roslaunch turtlebot_make_square start_service.launch
- rosrun services_pkg simple_service_client.py

Services - messages

- rosservice info /name_of_the_service
- rossrv show name_of_the_package/Name_of_Service_message
- roservice: live interaction with service servers (eq. rostopic)
- rossrv: offline service definitions (eq. rosmsg)
- rossrv show turtlebot_make_square/MoveInSquare
- Service message: *.srv | Topic message: *.msg
- srv directory | msg directory
- roscd turtlebot_make_square; ls srv

Services - messages

- Service message:

REQUEST

RESPONSE

- {} is possible
- **REQUEST**: how you call a service
- **RESPONSE**: how service will respond

Services - messages

- 3 message objects are created when a Service message is compiled
- MyServiceMessage: used for creating a connection to the service server

```
move_in_square_service = rospy.ServiceProxy('/move_in_square', MoveInSquare)
```

- MyServiceMessageRequest: used for creating a request to send to the server

```
move_in_square_object = MoveInSquareRequest()
# Send through the connection the request
result = move_in_square_service(move_in_square_object)
```

- MyServiceMessageResponse: used for sending a response from the server back to the client, when the service ends

server-side (next course)

Services - practice

- create launch file (my_turtlebot_demo.launch), that starts the /move_direction service
- this service is launched by the launch file turtlebot_move_start_service.launch, which is in the package turtlebot_move
- get information on what type of service message the /move_direction service uses
- make turtlebot move following a direction, which is specified as a string; two options: 'straight' and 'right'
- modify the previous code, which called the /move_in_square service, to now call the /move_direction service instead
- modify the launch file, so that now it also launches the python code just created

K6. Services

Messages & Servers

Services - servers

- cd ~/catkin_ws/src/services_pkg/src
- touch simple_service_server.py
- chmod +x simple_service_server.py
- gedit simple_service_server.py
- cd ~/catkin_ws
- catkin build
- source ~/.bashrc

simple_service_server.py

```
#! /usr/bin/env python

import rospy
# you import the service message python classes generated from Empty.srv
from std_srvs.srv import Empty, EmptyResponse

def my_callback(request):
    print("My_callback has been called")
    # the service Response class, in this case EmptyResponse
    return EmptyResponse()

rospy.init_node('service_server')
# create the Service called my_service with the defined callback
my_service = rospy.Service('/my_service', Empty , my_callback)
# maintain the service open
rospy.spin()
```

Services - servers

- rosrun services_pkg simple_service_server.py
- rosservice list
- rosservice call /my_service [TAB]+[TAB]
- create service: /move_tb_in_circle: {} -> {}
  - launch file
- create client
  - separate launch file

Services - messages

- cd ~/catkin_ws/src/services_pkg
- mkdir srv && cd srv
- gedit CustomServMess.srv

```
int32 duration      # The time (in seconds) during which BB-8 will keep moving in circles

---

bool success        # Did it achieve it?
```

CMakeLists.txt

```
find_package(catkin REQUIRED COMPONENTS
  std_msgs
  message_generation)

add_service_files(
  FILES
  CustomServMess.srv)

generate_messages(
  DEPENDENCIES
  std_msgs)

catkin_package(
      CATKIN_DEPENDS
      rospy)
```

package.xml

<build_depend>message_generation</build_depend>

<build_export_depend>message_runtime</
build_export_depend>

<exec_depend>message_runtime</exec_depend>

Services - messages

- cd ~/catkin_ws
- catkin build
- source ~/.bashrc
- rossrv list | grep CustomServMess
- create service: /move_tb_custom: (int32) duration -> (bool) success
  - launch file
- create client
  - launch both from single launch file