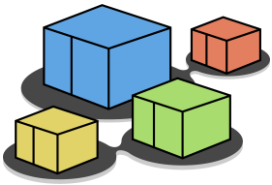


Kathará

Kathará lab

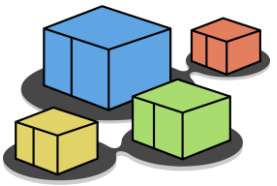
one bridge

Version	2.0
Author(s)	L. Ariemma, T. Caiazzzi, G. Di Battista
E-mail	contact@kathara.org
Web	http://www.kathara.org/
Description	One bridge and four computers are connected; the learning features of the bridge are experienced

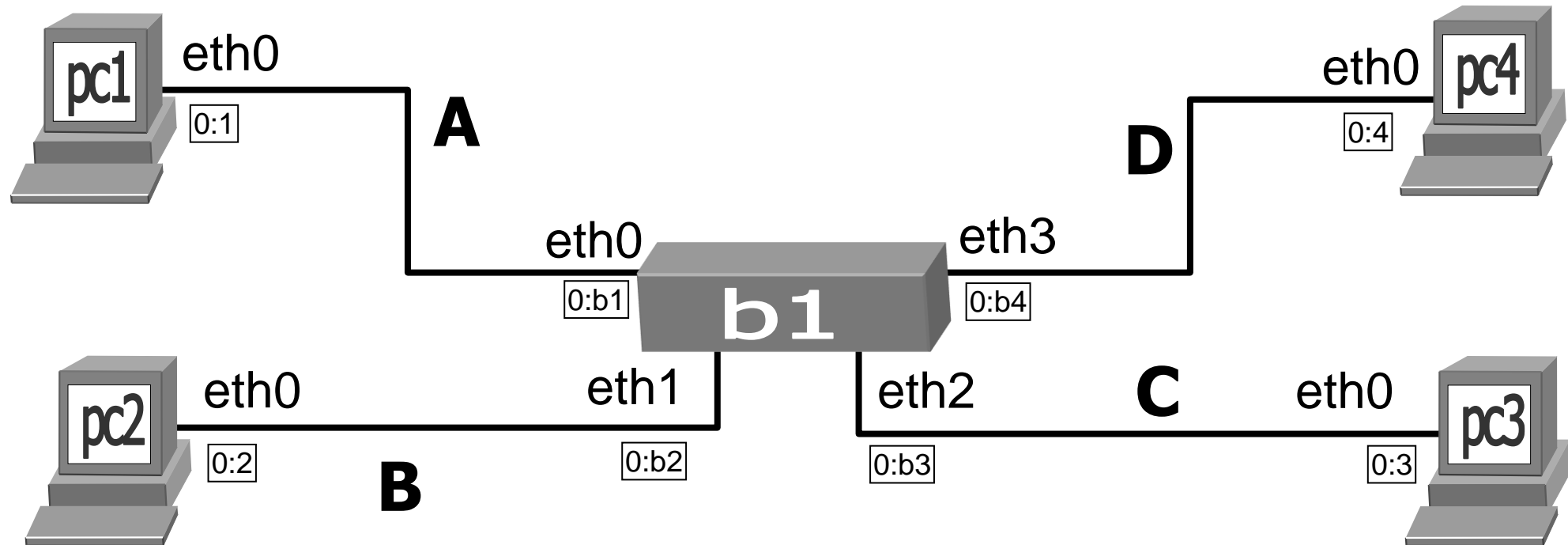


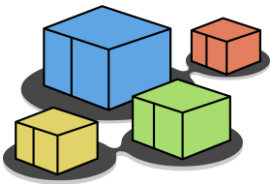
copyright notice

- All the pages/slides in this presentation, including but not limited to, images, photos, animations, videos, sounds, music, and text (hereby referred to as “material”) are protected by copyright.
- This material, with the exception of some multimedia elements licensed by other organizations, is property of the authors and/or organizations appearing in the first slide.
- This material, or its parts, can be reproduced and used for didactical purposes within universities and schools, provided that this happens for non-profit purposes.
- Information contained in this material cannot be used within network design projects or other products of any kind.
- Any other use is prohibited, unless explicitly authorized by the authors on the basis of an explicit agreement.
- The authors assume no responsibility about this material and provide this material “as is”, with no implicit or explicit warranty about the correctness and completeness of its contents, which may be subject to changes.
- This copyright notice must always be redistributed together with the material, or its portions.



network topology





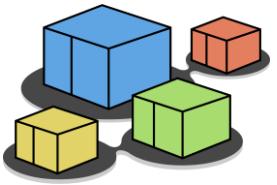
the lab.conf

lab.conf - part 1

```
pc1[0]="A/00:00:00:00:00:01"  
pc1[image]="kathara/base"  
pc1[ipv6]="false"  
  
pc2[0]="B/00:00:00:00:00:02"  
pc2[image]="kathara/base"  
pc2[ipv6]="false"  
  
pc3[0]="C/00:00:00:00:00:03"  
pc3[image]="kathara/base"  
pc3[ipv6]="false"  
  
pc4[0]="D/00:00:00:00:00:04"  
pc4[image]="kathara/base"  
pc4[ipv6]="false"
```

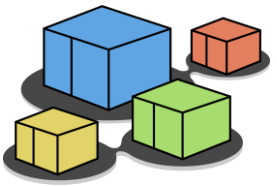
lab.conf - part 2

```
b1[0]="A/00:00:00:00:00:b1"  
b1[1]="B/00:00:00:00:00:b2"  
b1[2]="C/00:00:00:00:00:b3"  
b1[3]="D/00:00:00:00:00:b4"  
b1[image]="kathara/base"  
b1[ipv6]="false"
```



how to create a bridge

- we want device b1 to be a bridge
- Linux has an already available software bridge
 - we will use it
 - there are different utilities that manages different parts of the bridge
 - ip link
 - brctl
 - bridge
 - ...



how to create a bridge

- the command that lets b1 to be a bridge follows
 - the name of the device remains b1, the internal name we give to the bridge is mainbridge

```
root@b1:~$ ip link add name mainbridge type bridge
```

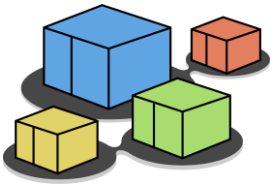
the internal
name of the
bridge

the type of the
newly created
object



remarks

- b1 is a name of the Kathará device
- mainbridge is the name of a bridge inside b1



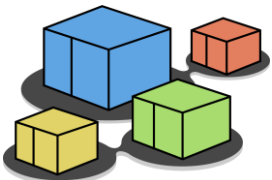
connect interfaces to a bridge

- the bridge needs then to be connected to some network interfaces
- connecting an interface to a bridge is called "enslaving"
 - it is done setting the bridge as the master of that interface

```
root@b1:~$ ip link set dev eth0 master mainbridge
```

the name of the
interface to be
connected

the name of the
bridge to connect
the interface



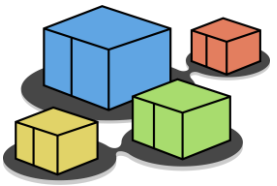
bringing up the bridge

- the bridge is born with its state DOWN
 - this means that it is switched off
- the following command allows to switch up/down a bridge (or even a network interface)
 - it can be also used for ethX interfaces

```
root@b1:~$ ip link set up dev mainbridge
```

the desired state

the name of the bridge to be brought on



bridge ageing time

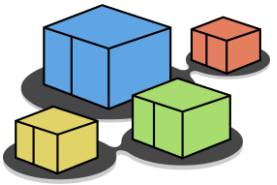
- the ageing time is the number of seconds that a MAC address will be kept in the FDB (Filtering Data Base)
- by default, it is set to 300 (5 minutes)
- to change the ageing time:

```
root@b1:~$ brctl setageing mainbridge 600
```

utility to control
bridges

name of the bridge

new ageing time,
set in seconds



b1.startup

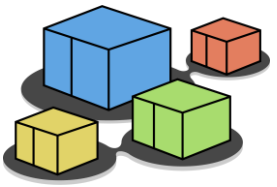
b1.startup

```
ip link add name mainbridge type bridge

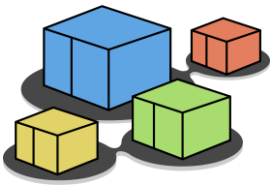
ip link set dev eth0 master mainbridge
ip link set dev eth1 master mainbridge
ip link set dev eth2 master mainbridge
ip link set dev eth3 master mainbridge

ip link set up dev mainbridge

brctl setageing mainbridge 600
```



let's start the lab

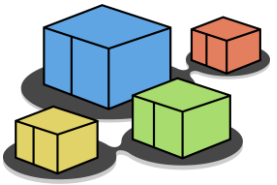


show the FDB of a bridge

- to show the FDB of a bridge, the `brctl showmacs` command can be used

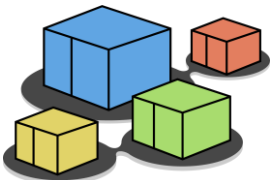
```
root@b1:~$ brctl showmacs mainbridge
```

port no	mac addr	is local?	ageing timer
1	00:00:00:00:00:b1	yes	0.00
1	00:00:00:00:00:b1	yes	0.00
2	00:00:00:00:00:b2	yes	0.00
2	00:00:00:00:00:b2	yes	0.00
3	00:00:00:00:00:b3	yes	0.00
3	00:00:00:00:00:b3	yes	0.00
4	00:00:00:00:00:b4	yes	0.00
4	00:00:00:00:00:b4	yes	0.00



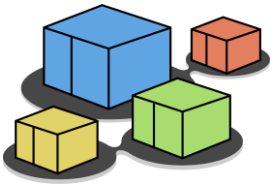
auto-learning local interfaces

- the Linux bridge automatically learns the MAC addresses of all its local interfaces
 - he already knows them because they are hosted on the same device
- the ageing timer is shown as 0.00 because those entries never expire



the port column

- in the `brctl showmacs <bridge_name>` command, the first column of the output is called port
- the port indicates the number of the virtual port of the bridge
- a linux bridge has a hard limit (hardcoded in the Kernel) of 1024 ports
- the ports are assigned starting from 1 in the order they are connected to interfaces
 - the order of the commands that enslave the interfaces



learn dynamic FDB entries – part 1

- if a packet is sent from a PC, the bridge will automatically learn its MAC address
- try to send a packet from pc1 to pc2 using scapy

```
root@pc1:~$ scapy
>>> p=Ether(dst='00:00:00:00:00:02', src='00:00:00:00:00:01')
>>> sendp(p, iface='eth0')
Sent 1 packets.
```



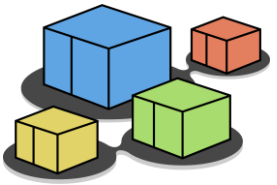

learn dynamic FDB entries – part 2

- now, take a look at the FDB of mainbridge again

```
root@b1:~$ brctl showmacs mainbridge
port no mac addr          is local? ageing timer
1      00:00:00:00:00:01    no         18.54
1      00:00:00:00:00:b1     yes         0.00
1      00:00:00:00:00:b1     yes         0.00
2      00:00:00:00:00:b2     yes         0.00
2      00:00:00:00:00:b2     yes         0.00
3      00:00:00:00:00:b3     yes         0.00
3      00:00:00:00:00:b3     yes         0.00
4      00:00:00:00:00:b4     yes         0.00
4      00:00:00:00:00:b4     yes         0.00
```

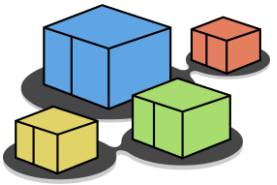
the MAC address
of pc1 has been
learnt

this timer
always
increases,
showing how
long the entry
has been in the
FDB



bridge ageing time

- the "ageing timer" of the previous slide always increases, showing how long the entry has been in the FDB
- eventually, the entry expires and simply disappears



exercises

- why, after the above experiment, in the FDB there is no entry for pc2?
- try to sniff from pc3 while sending the packet from pc1 to pc2, what happens?
- send a "reply" packet from pc2, how the FDB changes?
- try again to sniff from pc3 while sending the packet from pc1 to pc2, is the result the same?
- try to send packets with the wrong source address