Università degli Studi Roma Tre
Dipartimento di Informatica e Automazione
Computer Networks Research Group

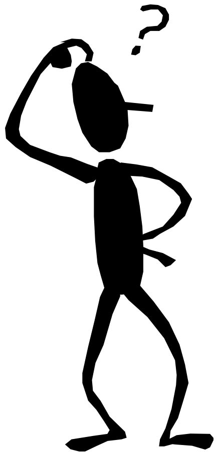
netkit lab

walkthrough

Version	1.3
Author(s)	Massimo Rimondini
E-mail	contact@netkit.org
Web	http://www.netkit.org/
Description	a step-by-step example showing how to set up a complete netkit lab with a few technologies

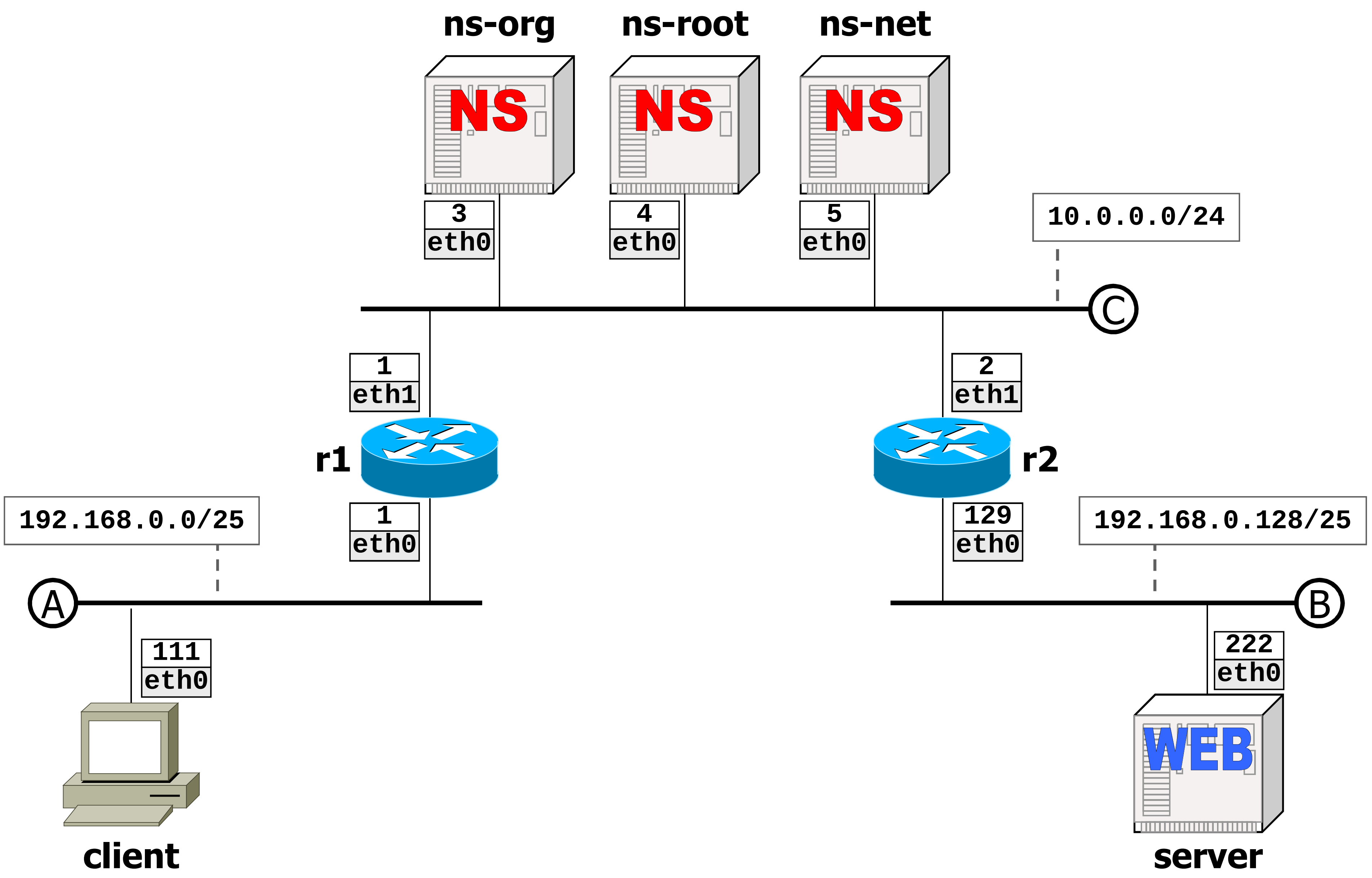
copyright notice

walk through what?



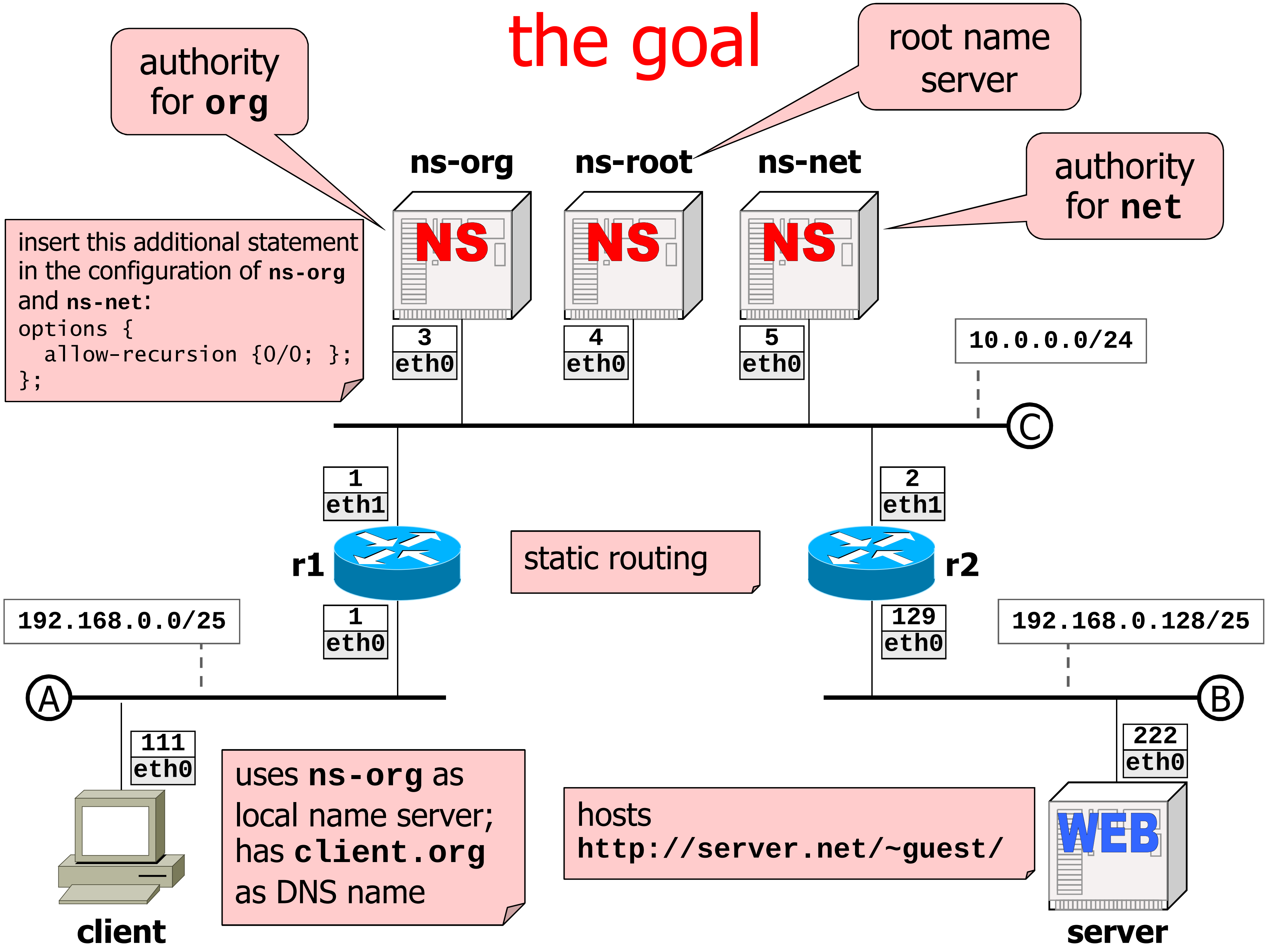
- the goal of this lab is to:
  - put together some technologies presented in other labs
  - step-by-step show how to set up a netkit lab from scratch
- prerequisites
  - it is advisable to take a look at the following netkit labs beforehand:
    - two hosts
    - static routing
    - web server
    - dns

the goal



ns-org
ns-root
ns-net
NS
NS
NS
3
eth0
4
eth0
5
eth0
10.0.0.0/24
C
1
eth1
2
eth1
r1
r2
1
eth0
129
eth0
192.168.0.0/25
192.168.0.128/25
A
B
111
eth0
222
eth0
WEB
client
server

the goal



insert this additional statement in the configuration of **ns-org** and **ns-net**:

```
options {
  allow-recursion {0/0; };
};
```

uses **ns-org** as local name server; has **client.org** as DNS name

hosts **http://server.net/~guest/**

vademecum

1. create an empty directory where to put the lab
2. set up physical topology
3. set up routing
4. set up additional technologies
   - in this lab:
     - web server
     - dns

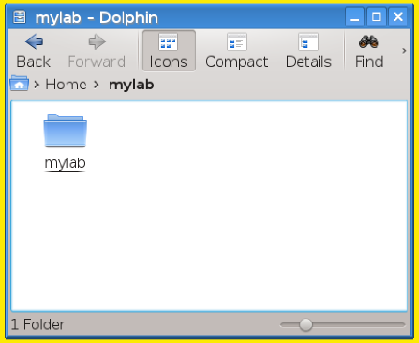
vademecum

1. create an empty directory where to put the lab
2. set up physical topology
3. set up routing
4. set up additional technologies
   - in this lab:
     - web server
     - dns

vademecum

1. create an empty directory where to put the lab

```
host machine
user@localhost:~$ mkdir mylab
user@localhost:~$ cd mylab
user@localhost:~/mylab$ █
```



note: here we use terminal commands for any operations
if you feel more comfortable with a graphical file manager/editor, feel free to use it

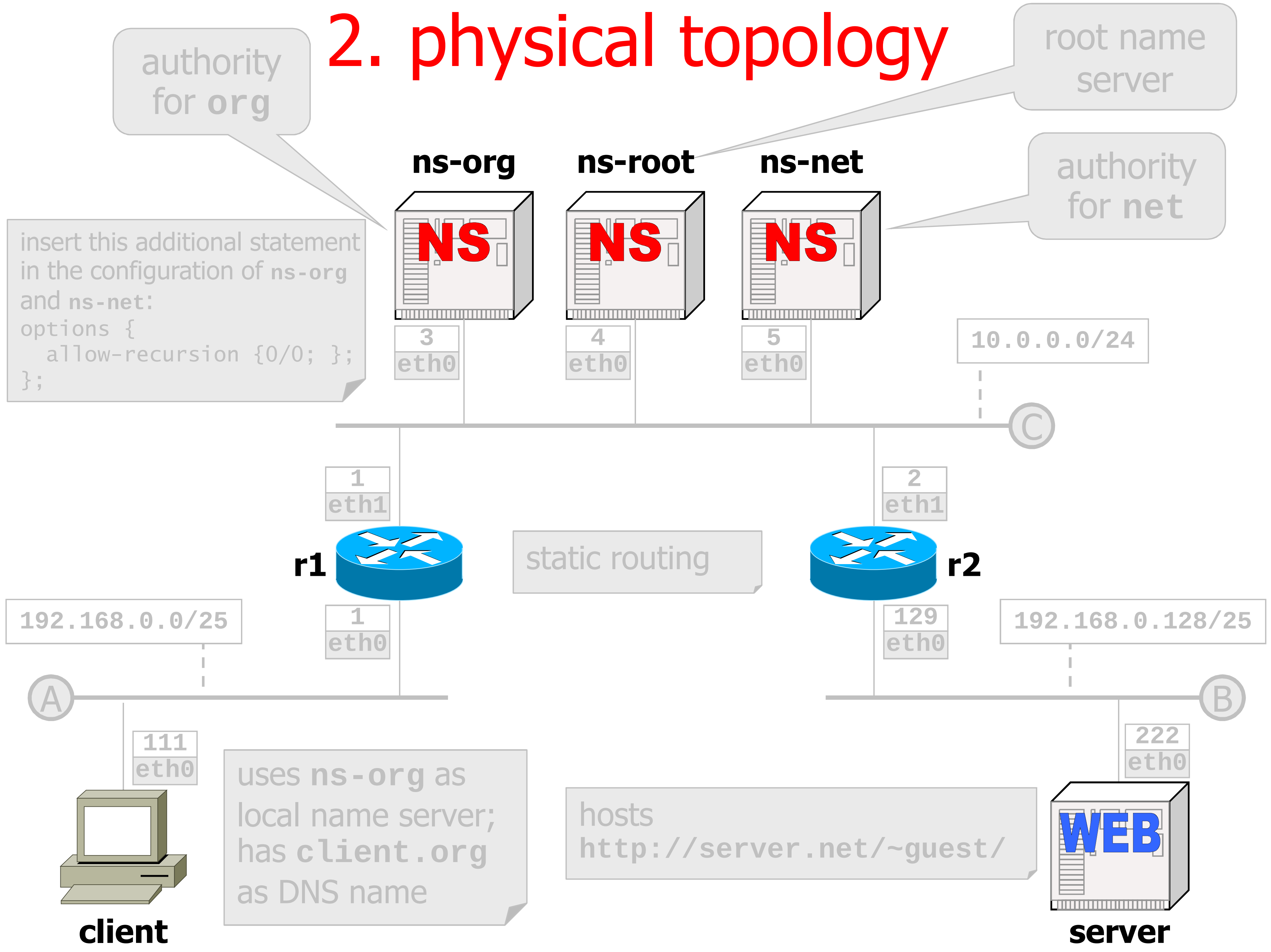
vademecum

1. create an empty directory where to put the lab
2. set up physical topology
3. set up routing
4. set up additional technologies
   - in this lab:
     - web server
     - dns

2. physical topology

- first of all, we tell netkit which virtual machines (=network nodes) the network consists of

2. physical topology



insert this additional statement in the configuration of **ns-org** and **ns-net**:

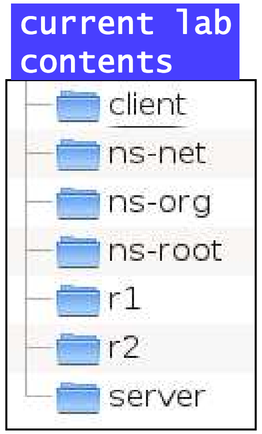
```
options {
  allow-recursion {0/0; };
};
```

uses **ns-org** as local name server; has **client.org** as DNS name

hosts
`http://server.net/~guest/`

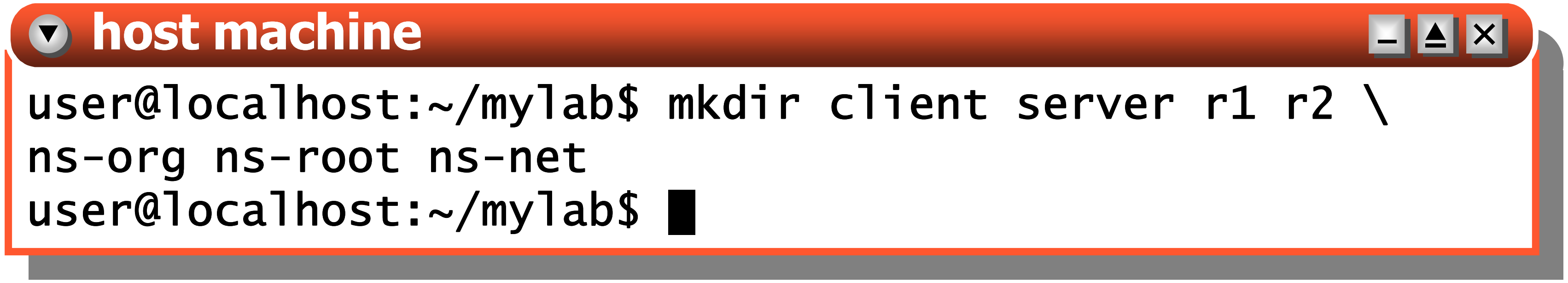
2. physical topology

- in netkit, each virtual machine corresponds to a directory in the lab



current lab contents

```
client
ns-net
ns-org
ns-root
r1
r2
server
```



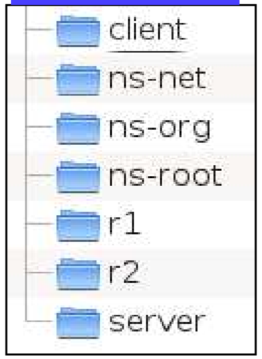
host machine

```
user@localhost:~/mylab$ mkdir client server r1 r2 \
ns-org ns-root ns-net
user@localhost:~/mylab$ █
```

2. physical topology

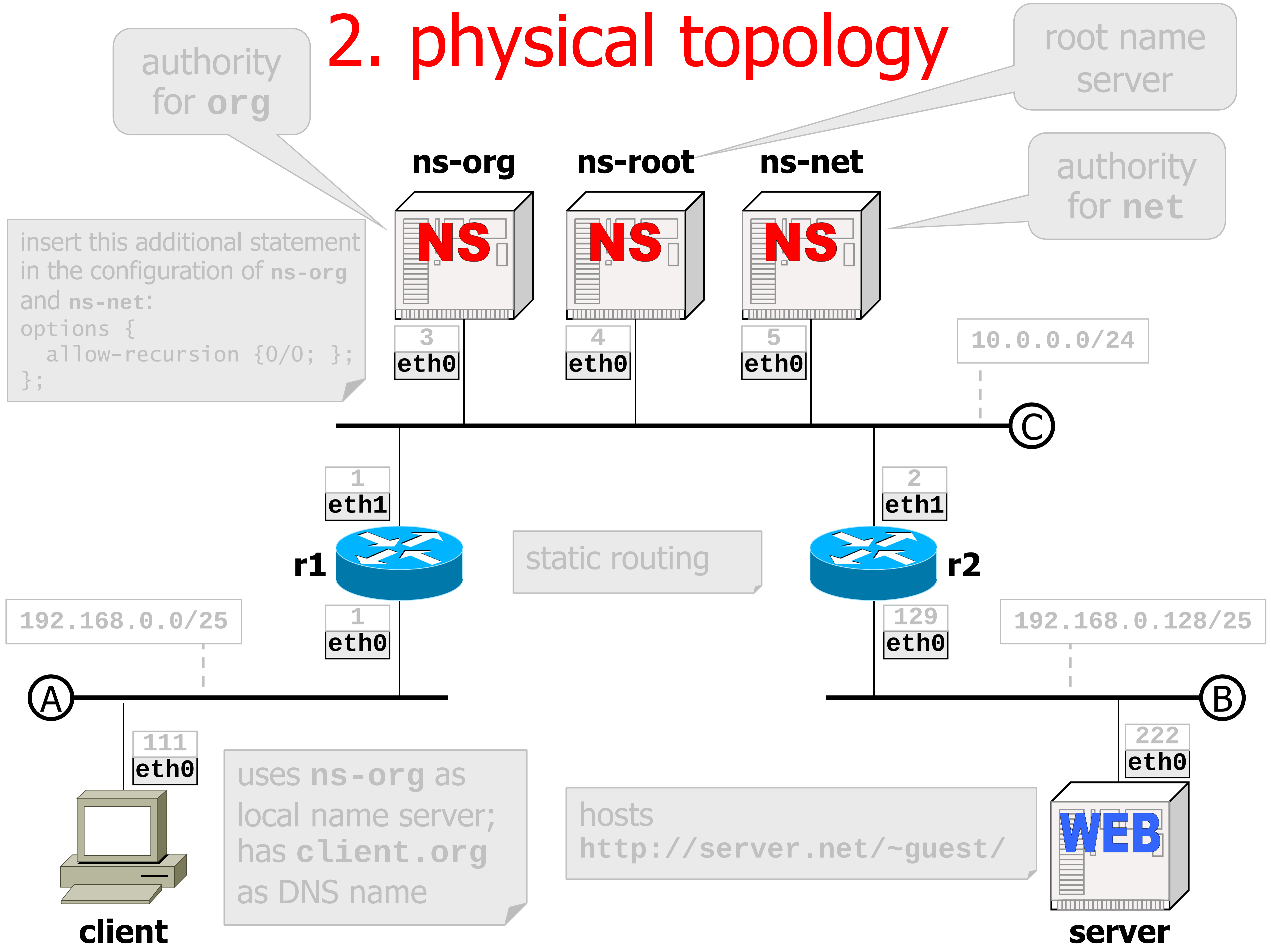
- now, we tell netkit about how virtual machines (=network nodes) are interconnected
- this information goes into file **`lab.conf`**

current lab contents



```
client
ns-net
ns-org
ns-root
r1
r2
server
```

2. physical topology



insert this additional statement in the configuration of ns-org and ns-net:

```
options {
  allow-recursion {0/0; };
};
```

uses ns-org as local name server; has client.org as DNS name

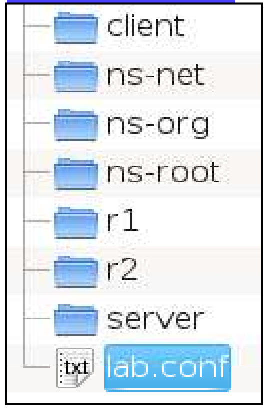
hosts http://server.net/~guest/

2. physical topology

lab.conf

```
client[0]=A

r1[0]=A
r1[1]=C

ns-org[0]=C

ns-root[0]=C

ns-net[0]=C

r2[0]=B
r2[1]=C

server[0]=B
```

current lab contents



```
client
ns-net
ns-org
ns-root
r1
r2
server
lab.conf
```

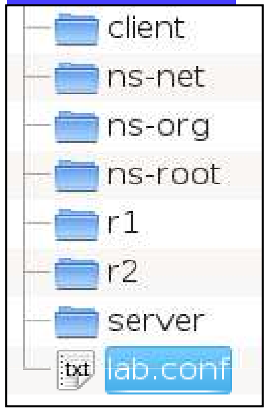
2. physical topology

lab.conf

```
client[0]=A

r1[0]=A
r1[1]=C

ns-org[0]=C

ns-root[0]=C

ns-net[0]=C

r2[0]=B
r2[1]=C

server[0]=B
```

- **`r1`**'s interface **`eth0`** is connected to collision domain **`A`**
- **`r1`**'s interface **`eth1`** is connected to collision domain **`C`**

current lab contents

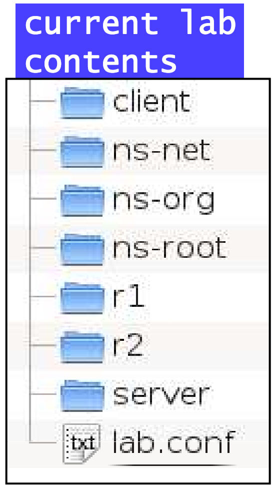


vademecum

1. create an empty directory where to put the lab
2. set up physical topology
3. set up routing
4. set up additional technologies
   - in this lab:
     - web server
     - dns

3. routing

- the configuration of ip routing consists of two parts:
  - assignment of ip addresses to network interfaces
    - achieved using the **`ifconfig`** command
  - configuration of static routing
    - achieved using the **`route`** command
- all these things are commands that virtual machines must run at startup
  - we put them inside **`.startup`** files

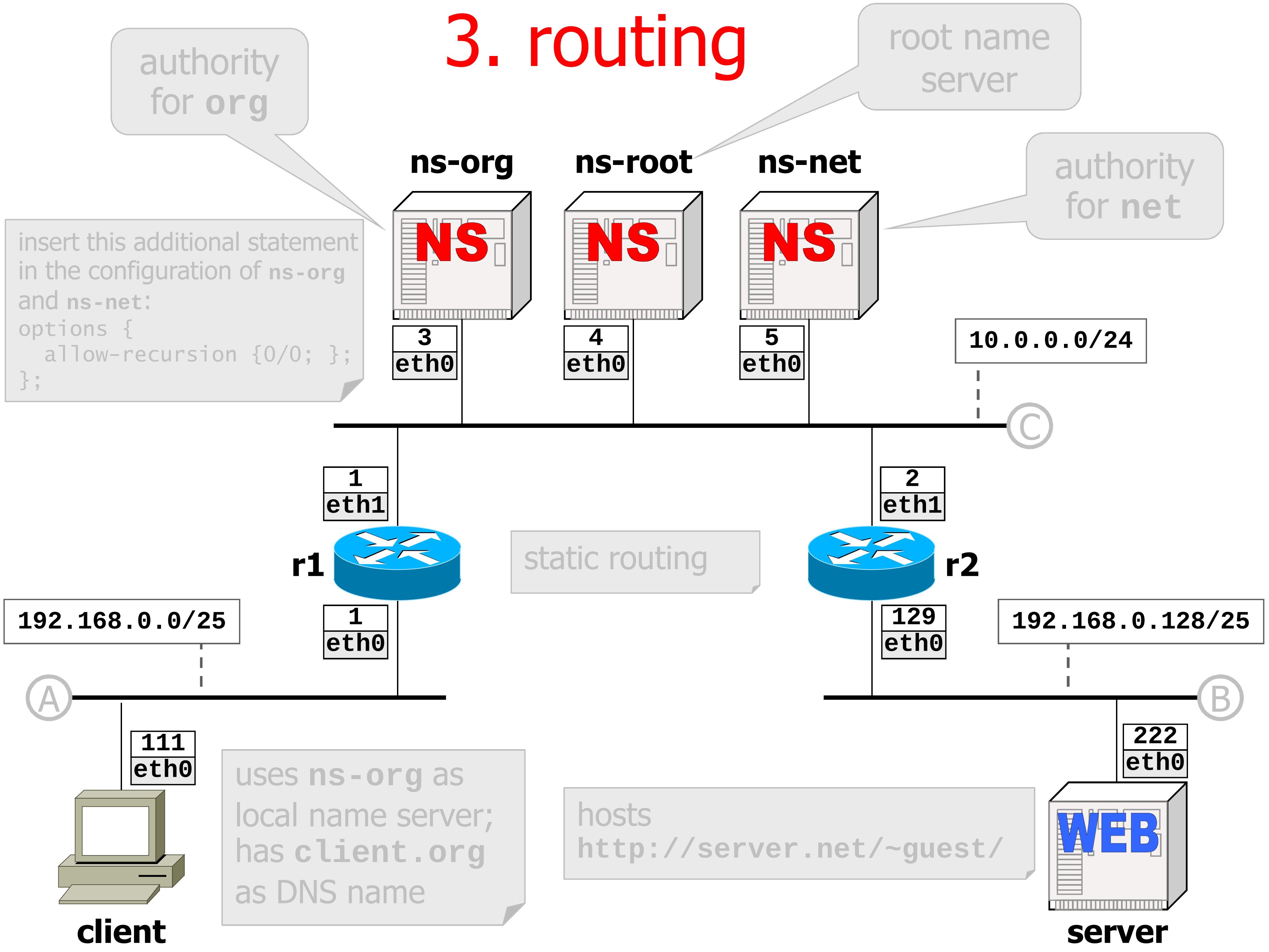


current lab contents

```
client
ns-net
ns-org
ns-root
r1
r2
server
lab.conf
```

tip: these files are very similar to each other, so copy&paste is your friend ;-)

3. routing



insert this additional statement in the configuration of ns-org and ns-net:

```
options {
  allow-recursion {0/0; };
};
```

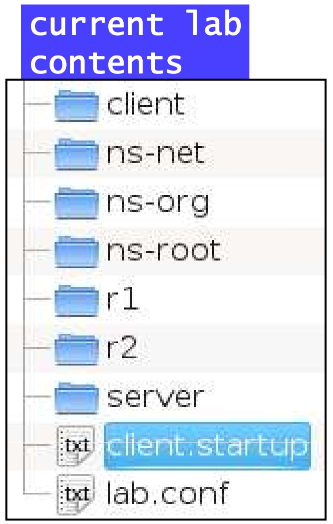
uses ns-org as local name server; has client.org as DNS name

hosts http://server.net/~guest/

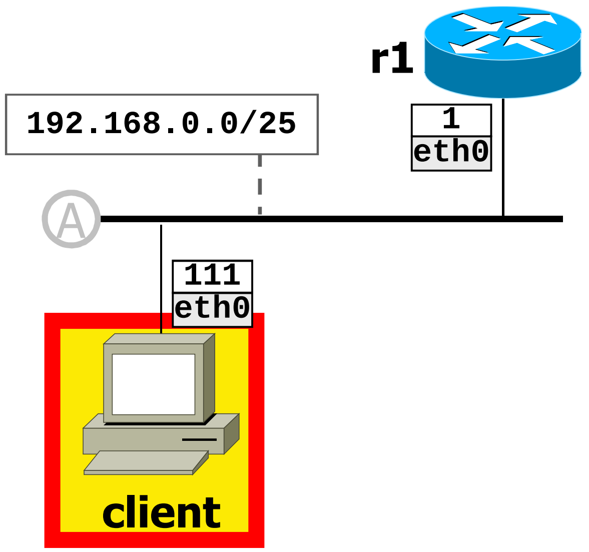
3. routing

client.startup

```
ifconfig eth0 192.168.0.111 netmask 255.255.255.128 up

route add default gw 192.168.0.1 dev eth0
```


current lab contents

- client
- ns-net
- ns-org
- ns-root
- r1
- r2
- server
- client.startup
- lab.conf



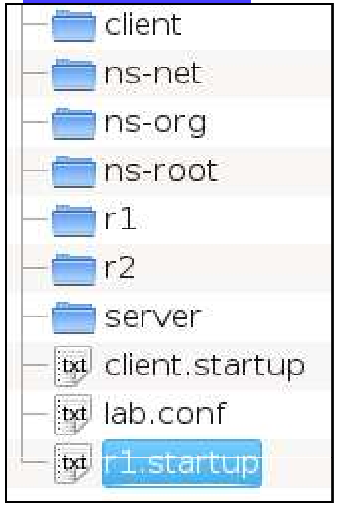
3. routing

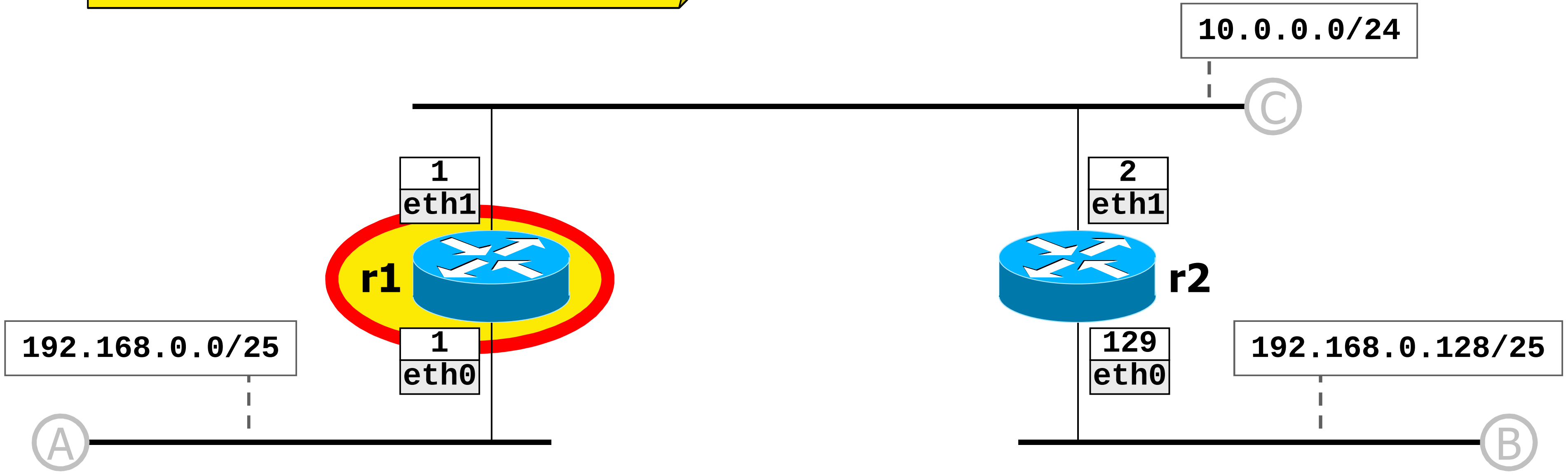
r1.startup

```
ifconfig eth0 192.168.0.1 netmask 255.255.255.128 up
ifconfig eth1 10.0.0.1 netmask 255.255.255.0 up

route add -net 192.168.0.128/25 gw 10.0.0.2 dev eth1
```

each router must learn about non-adjacent networks (only)

current lab contents



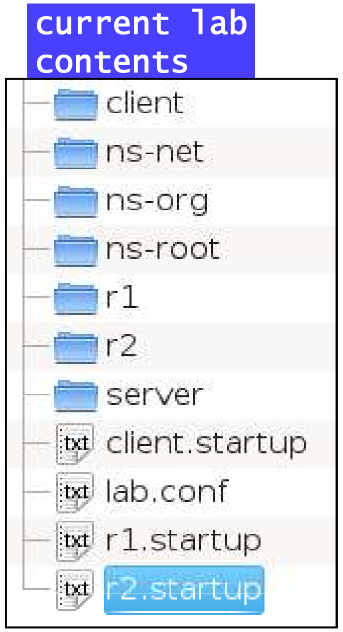


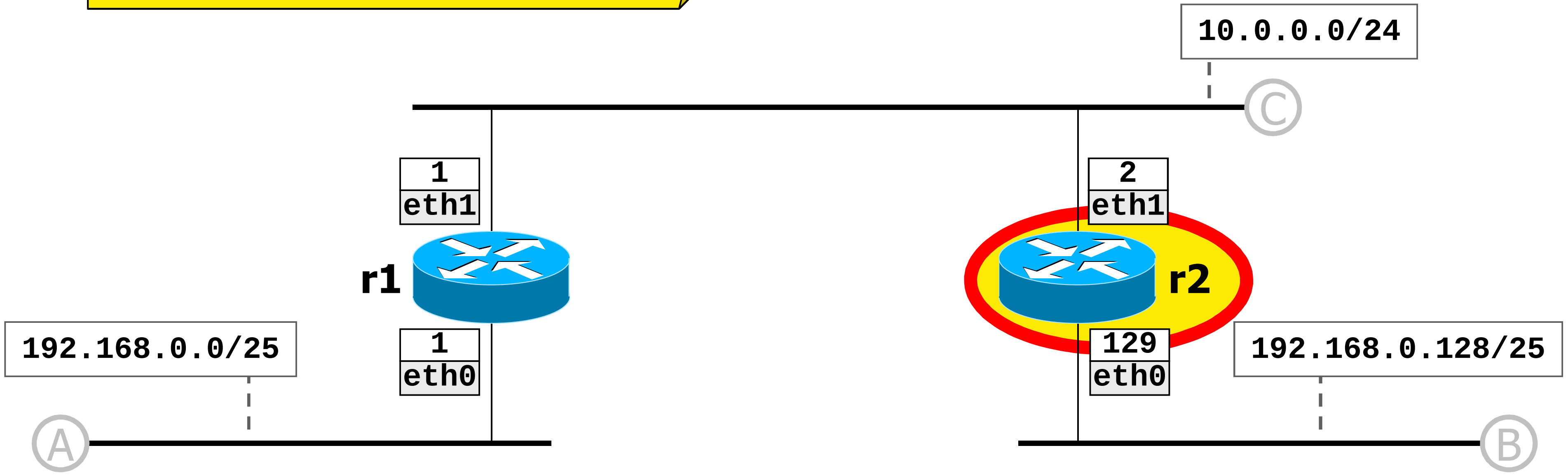
3. routing

r2.startup

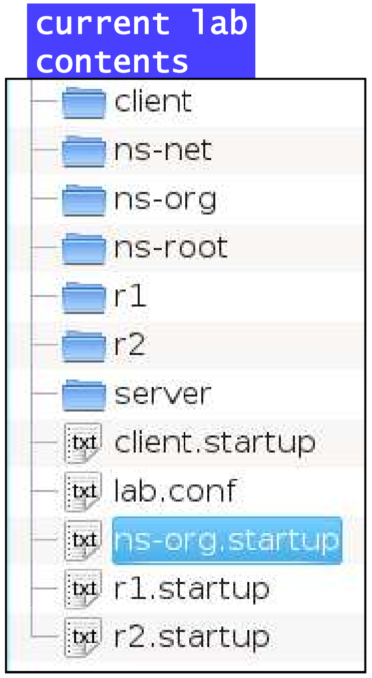
```
ifconfig eth0 192.168.0.129 netmask 255.255.255.128 up
ifconfig eth1 10.0.0.2 netmask 255.255.255.0 up

route add -net 192.168.0.0/25 gw 10.0.0.1 dev eth1
```

each router must learn about non-adjacent networks (only)





3. routing

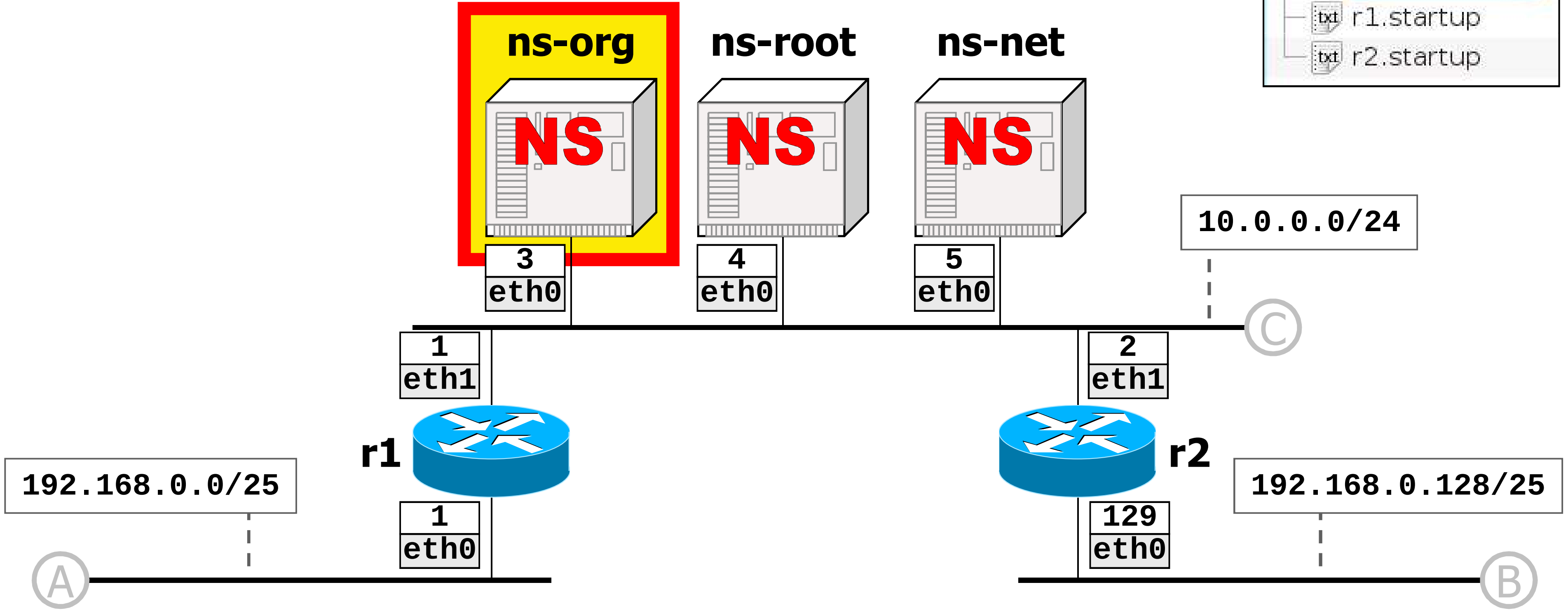


current lab contents

- client
- ns-net
- ns-org
- ns-root
- r1
- r2
- server
- client.startup
- lab.conf
- ns-org.startup
- r1.startup
- r2.startup

ns-org.startup

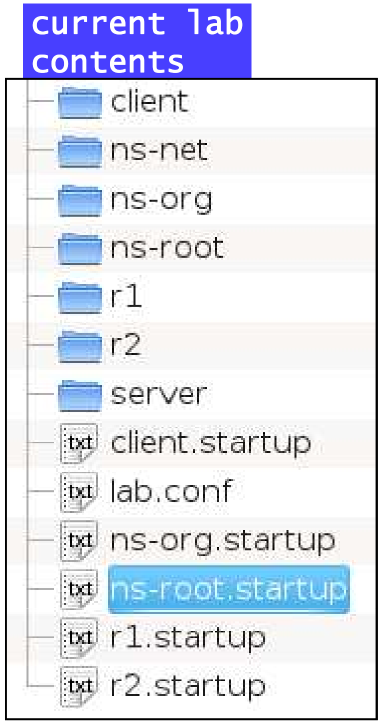
```
ifconfig eth0 10.0.0.3 netmask 255.255.255.0 up

route add -net 192.168.0.0/25 gw 10.0.0.1 dev eth0
route add -net 192.168.0.128/25 gw 10.0.0.2 dev eth0
```

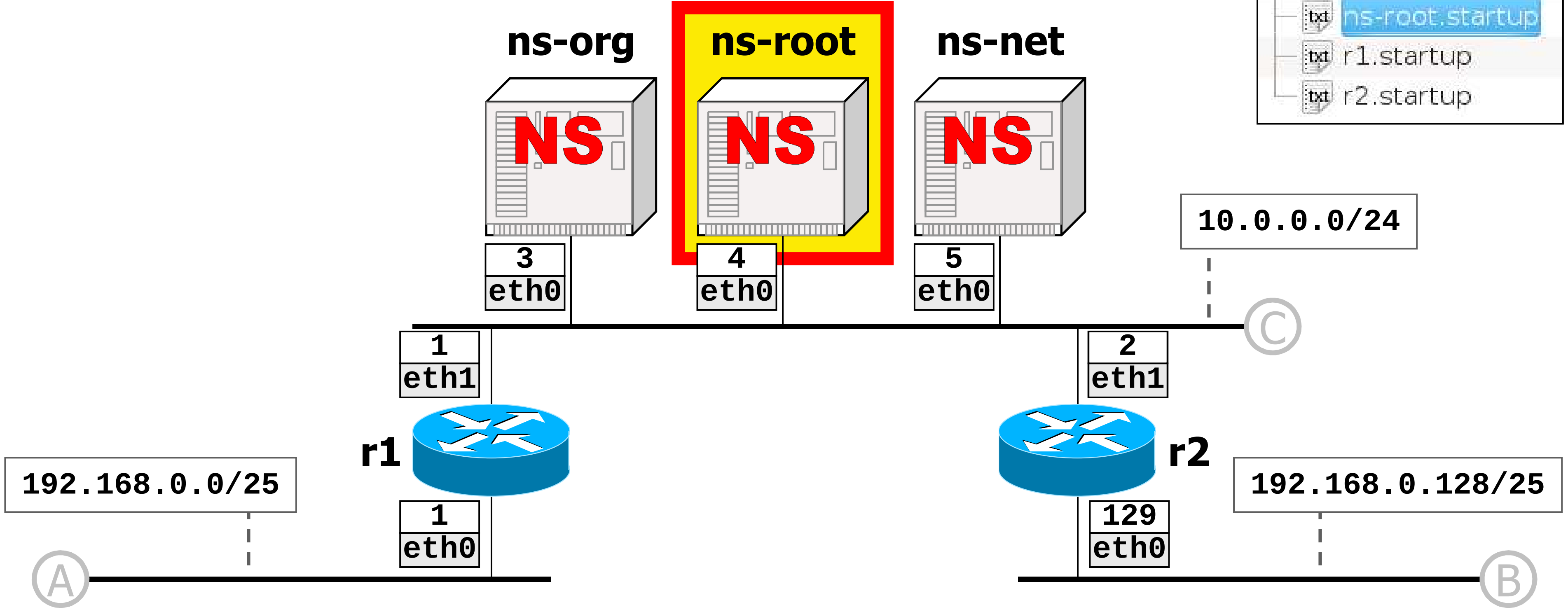


3. routing

ns-root.startup

```
ifconfig eth0 10.0.0.4 netmask 255.255.255.0 up

route add -net 192.168.0.0/25 gw 10.0.0.1 dev eth0
route add -net 192.168.0.128/25 gw 10.0.0.2 dev eth0
```

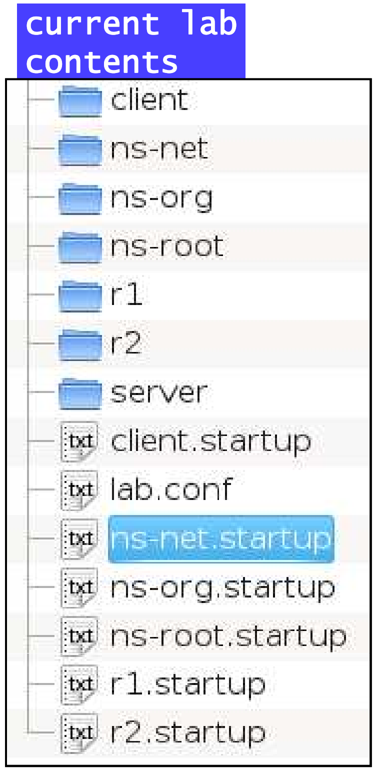




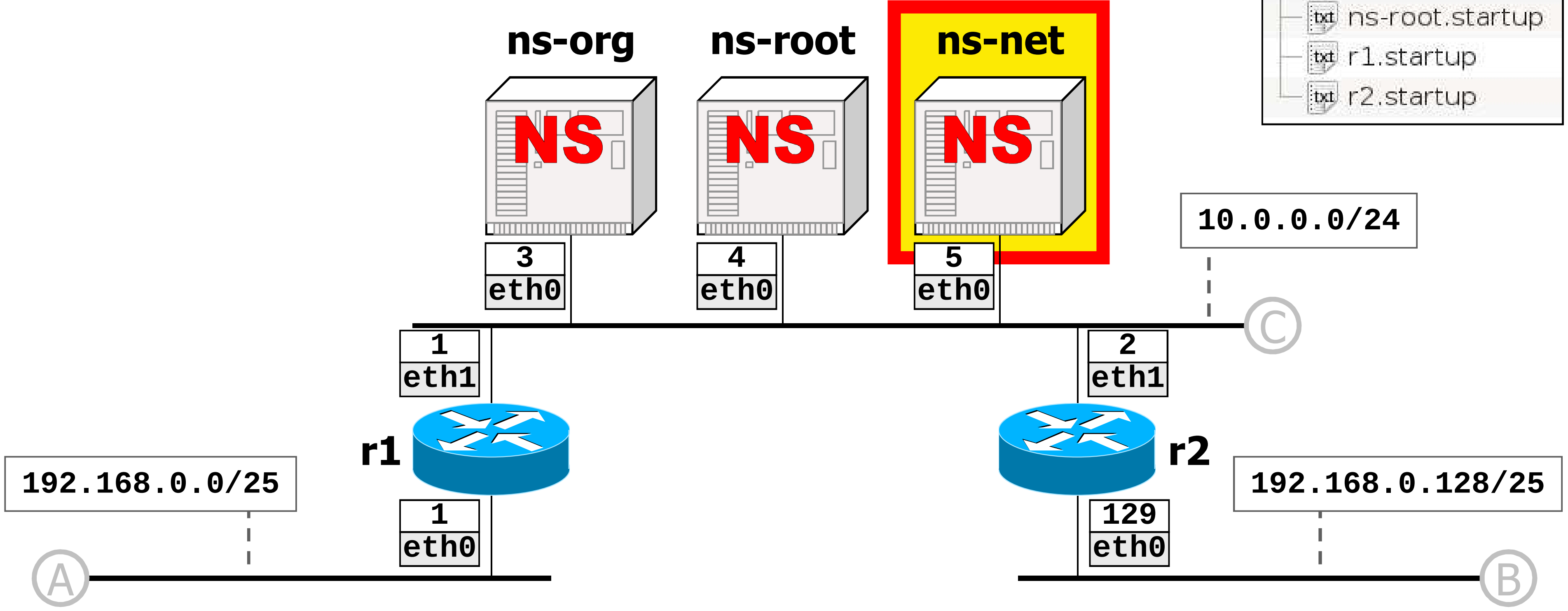
3. routing

ns-net.startup

```
ifconfig eth0 10.0.0.5 netmask 255.255.255.0 up

route add -net 192.168.0.0/25 gw 10.0.0.1 dev eth0
route add -net 192.168.0.128/25 gw 10.0.0.2 dev eth0
```


current lab contents

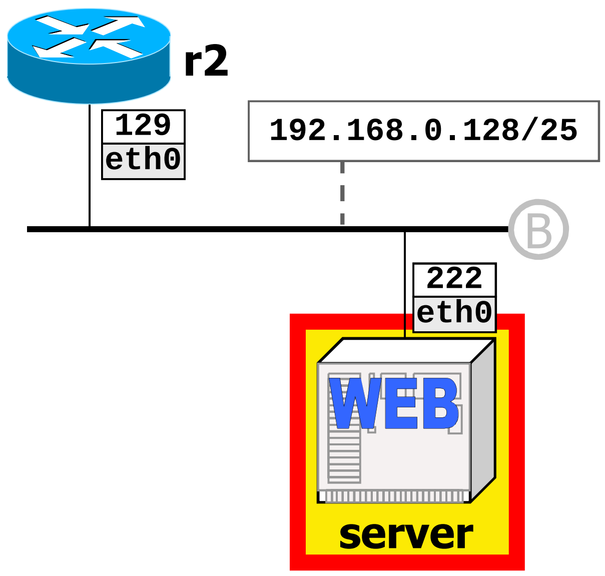
- client
- ns-net
- ns-org
- ns-root
- r1
- r2
- server
- client.startup
- lab.conf
- ns-net.startup
- ns-org.startup
- ns-root.startup
- r1.startup
- r2.startup

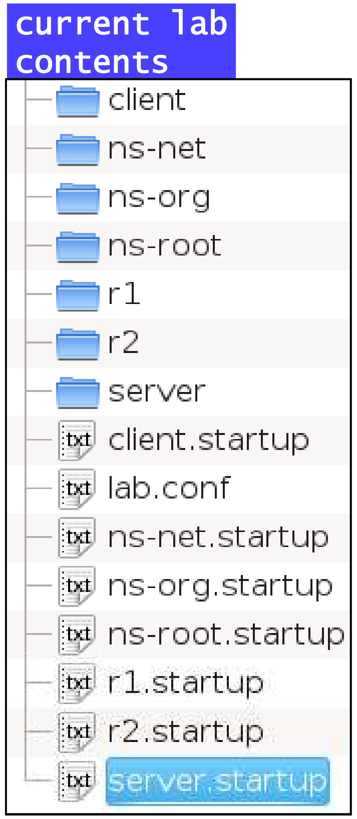


3. routing

server.startup

```
ifconfig eth0 192.168.0.222 netmask 255.255.255.128 up

route add default gw 192.168.0.129 dev eth0
```

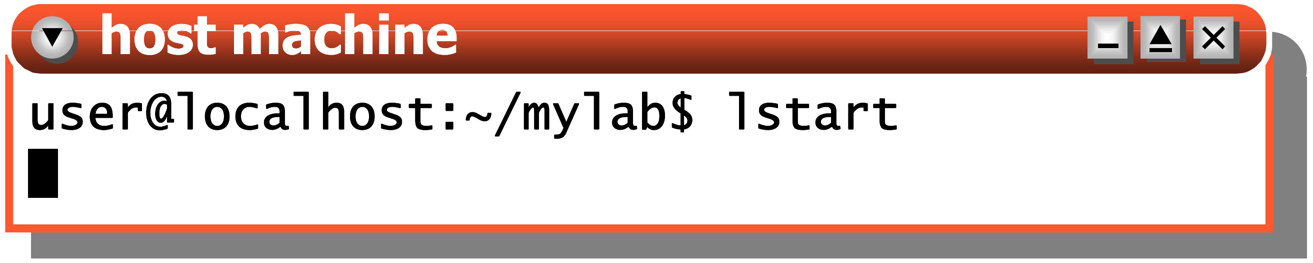



current lab contents

- client
- ns-net
- ns-org
- ns-root
- r1
- r2
- server
- client.startup
- lab.conf
- ns-net.startup
- ns-org.startup
- ns-root.startup
- r1.startup
- r2.startup
- server.startup

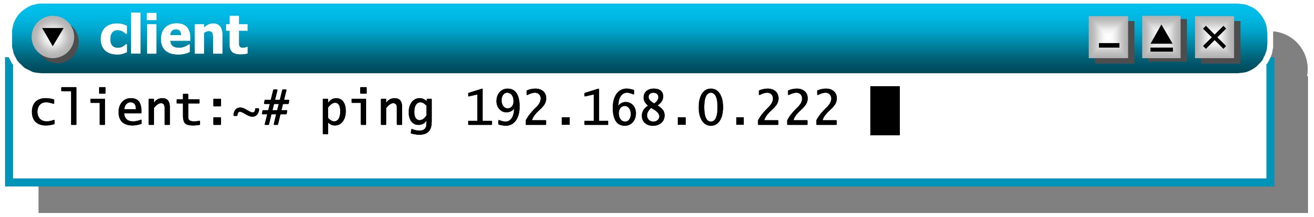
3. routing

- at this point it is strongly advised to start the lab and check that the routing works



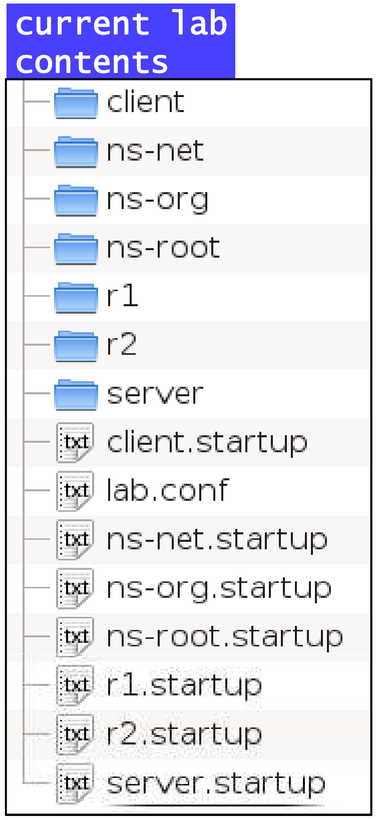
host machine

```
user@localhost:~/mylab$ lstart
█
```



client

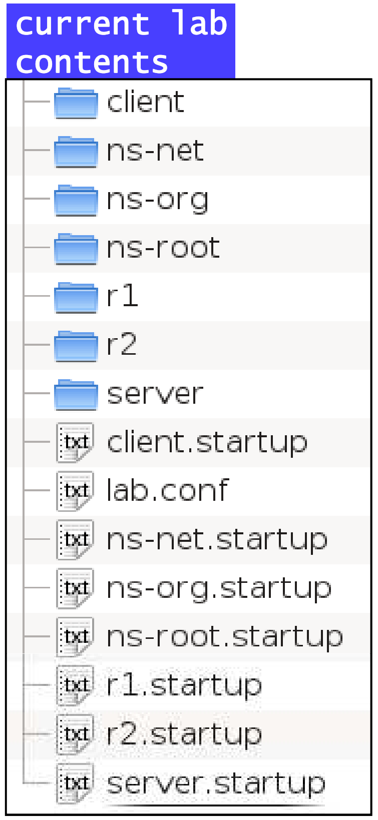
```
client:~# ping 192.168.0.222 █
```



current lab contents

```
├── client
├── ns-net
├── ns-org
├── ns-root
├── r1
├── r2
├── server
├── client.startup
├── lab.conf
├── ns-net.startup
├── ns-org.startup
├── ns-root.startup
├── r1.startup
├── r2.startup
└── server.startup
```

3. routing


current lab contents

```
client
ns-net
ns-org
ns-root
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

- at this point it is strongly advised to start the lab and check that the routing works
  - if it doesn't...

    - ...check physical topology (**`lab.conf`**)
    - ...check boot-time virtual machine messages (errors printed in blue are relevant)
    - ...check routing tables (**`route -n`**)
    - ...

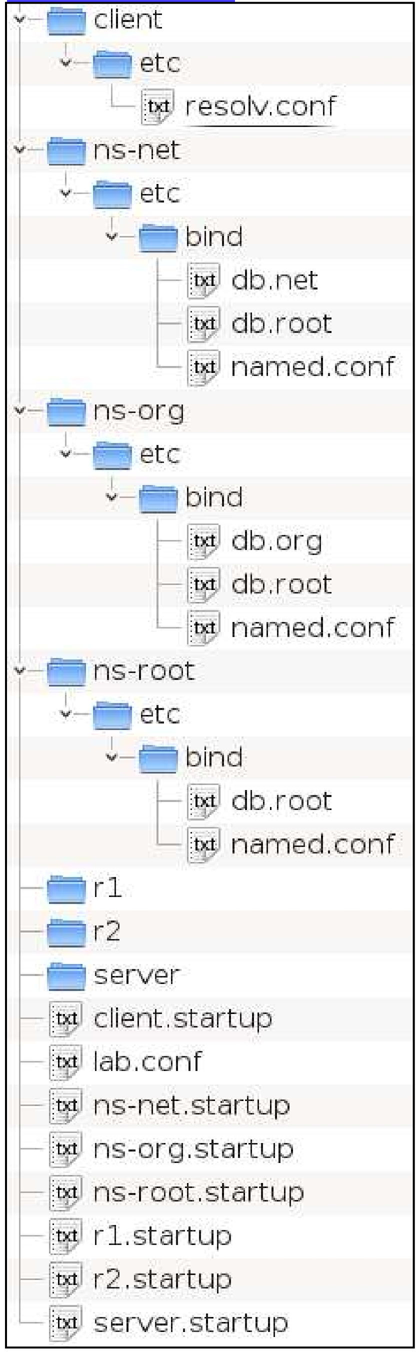
vademecum

1. create an empty directory where to put the lab
2. set up physical topology
3. set up routing
4. set up additional technologies
   - in this lab:
     - web server
     - dns

4. web server

- first of all, we need to instruct **`server`** to start the web server (**`apache`**) at boot time
- since we must set up a user's web site, we also need to enable **`apache`**'s **`userdir`** module
- this is achieved by adding lines to **`server.startup`**

current lab contents



```
client
  etc
    resolv.conf
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

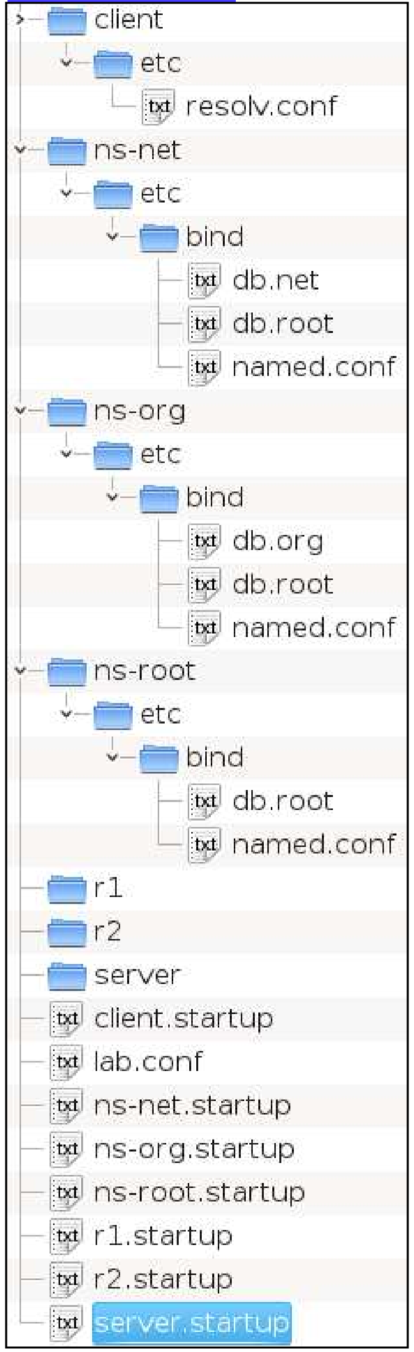
4. web server

- the module must be enabled *before* **apache** is started

server.startup

```
ifconfig eth0 192.168.0.222 netmask 255.255.255.128 up

route add default gw 192.168.0.129 dev eth0

a2enmod userdir
/etc/init.d/apache2 start
```

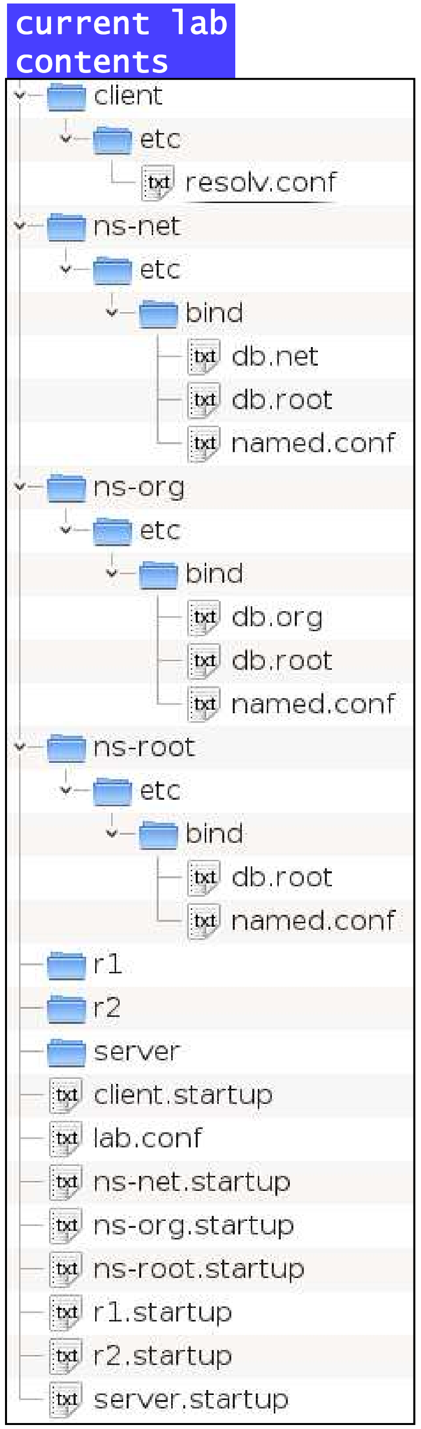
current lab contents



```
client
    etc
        resolv.conf
ns-net
    etc
        bind
            db.net
            db.root
            named.conf
ns-org
    etc
        bind
            db.org
            db.root
            named.conf
ns-root
    etc
        bind
            db.root
            named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

4. web server

- now, we create a simple home page for user **guest** (the only non-root user that is available by default in netkit)
- according to the default configuration of module **userdir**, this page must be placed in **/home/guest/public_html/index.html**
- therefore, we put it in **server/home/guest/public_html/index.html**



current lab contents

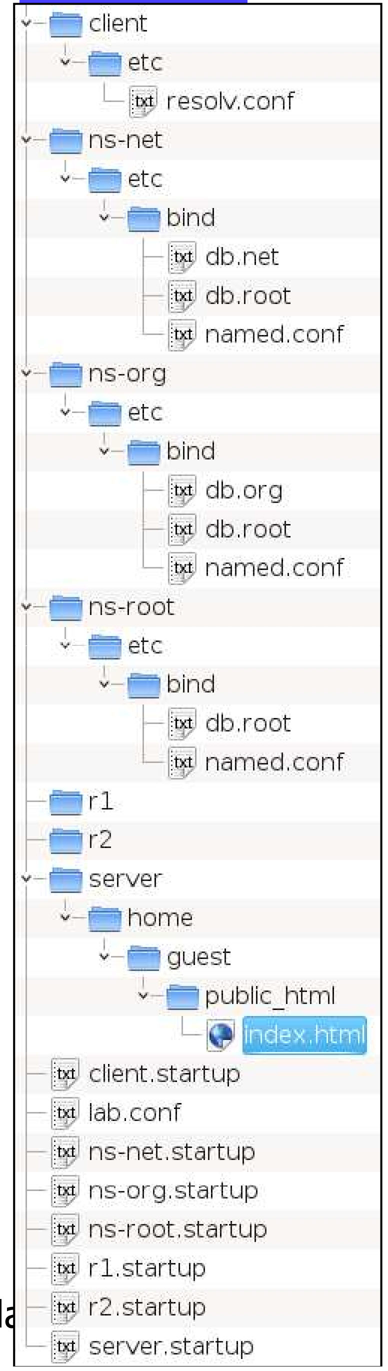
```
client
  etc
    resolv.conf
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

4. web server

hosts
`http://server.net/~guest/`

```
server/home/guest/public_html/index.html

<html>
  <body>
    Hello!
  </body>
</html>
```

current lab contents



```
client
  etc
    resolv.conf
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
  home
    guest
      public_html
        index.html
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

4. web server

- at this point we can start the lab and check that the web server works

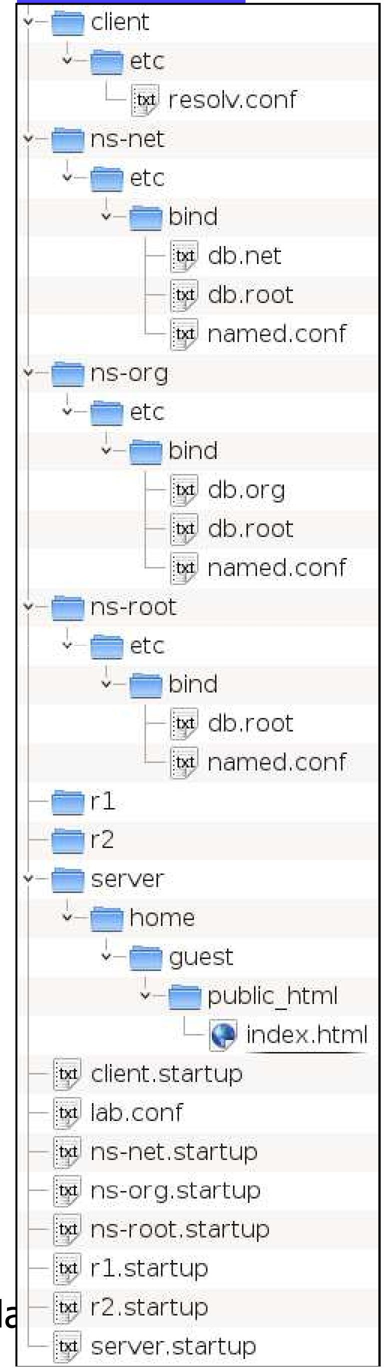
host machine

```
user@localhost:~/mylab$ lstart
```

client

```
client:~# links http://server.net/~guest
```

current lab contents



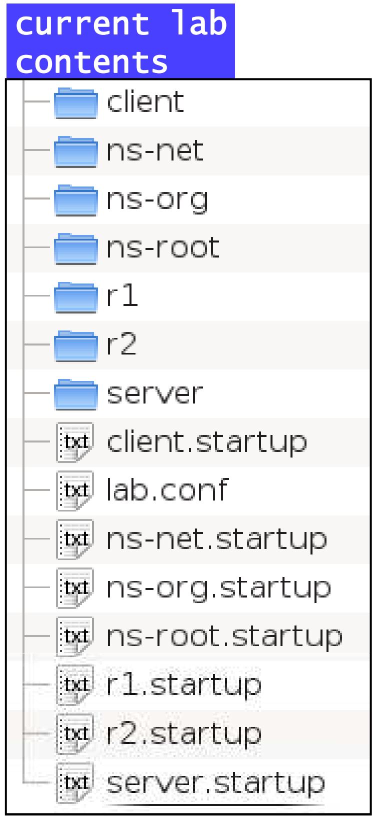
```
client
  etc
    resolv.conf
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
  home
    guest
      public_html
        index.html
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

vademecum

1. create an empty directory where to put the lab
2. set up physical topology
3. set up routing
4. set up additional technologies
   - in this lab:
     - dns
     - web server

4. dns

- first of all, we need to instruct some virtual machines to start a name server software (**`bind`**) at boot time
- we need to add a line to **`.startup`** files

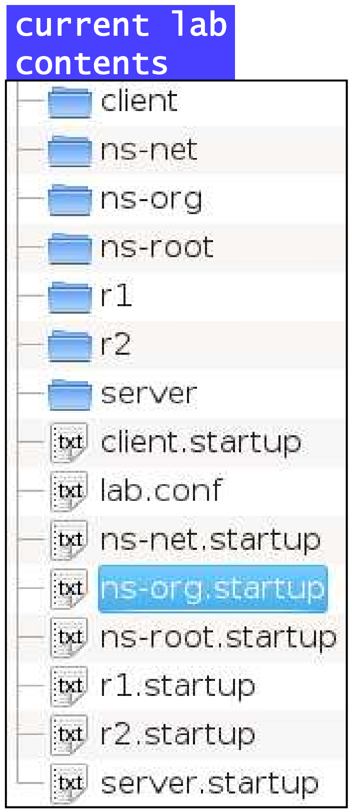

current lab contents

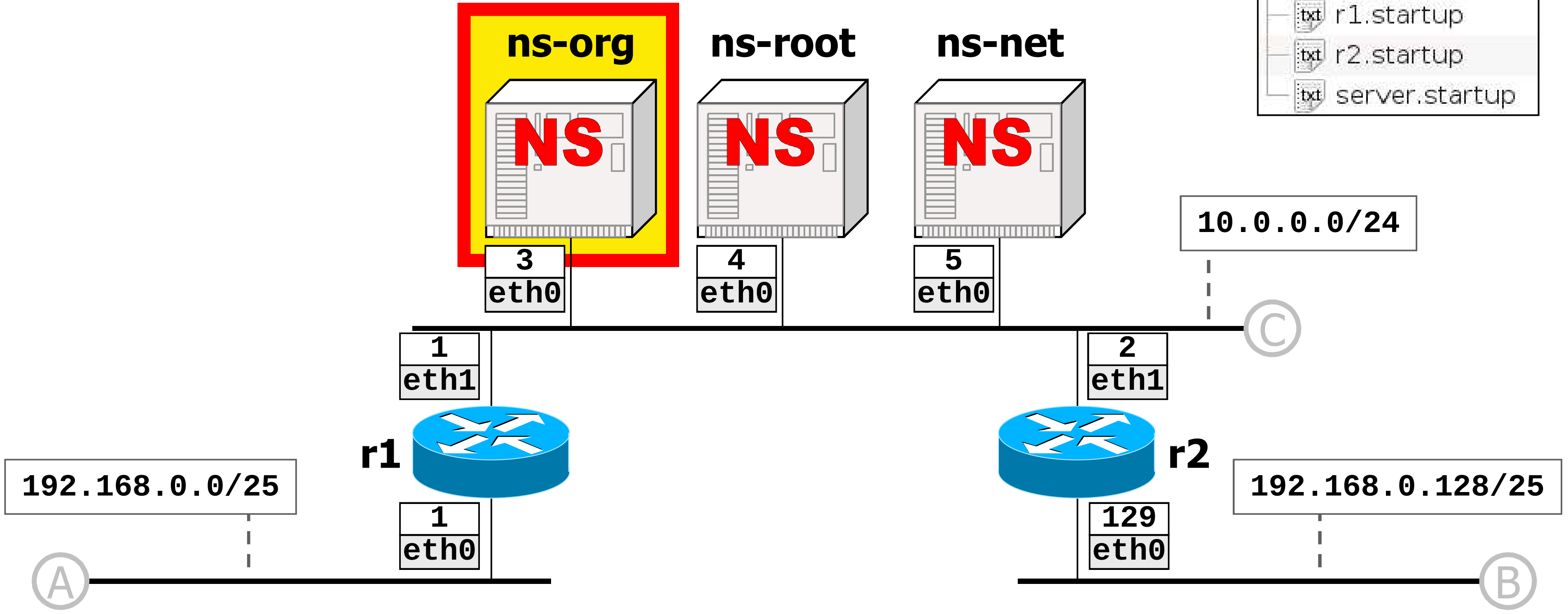
```
client
ns-net
ns-org
ns-root
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

4. dns

ns-org.startup

```
ifconfig eth0 10.0.0.3 netmask 255.255.255.0 up

route add -net 192.168.0.0/25 gw 10.0.0.1 dev eth0
route add -net 192.168.0.128/25 gw 10.0.0.2 dev eth0
/etc/init.d/bind start
```

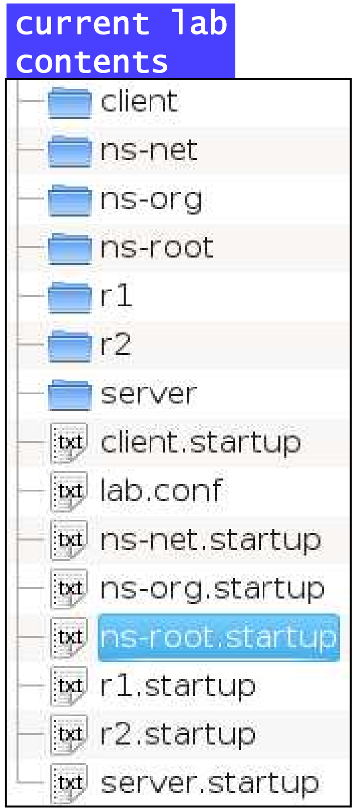




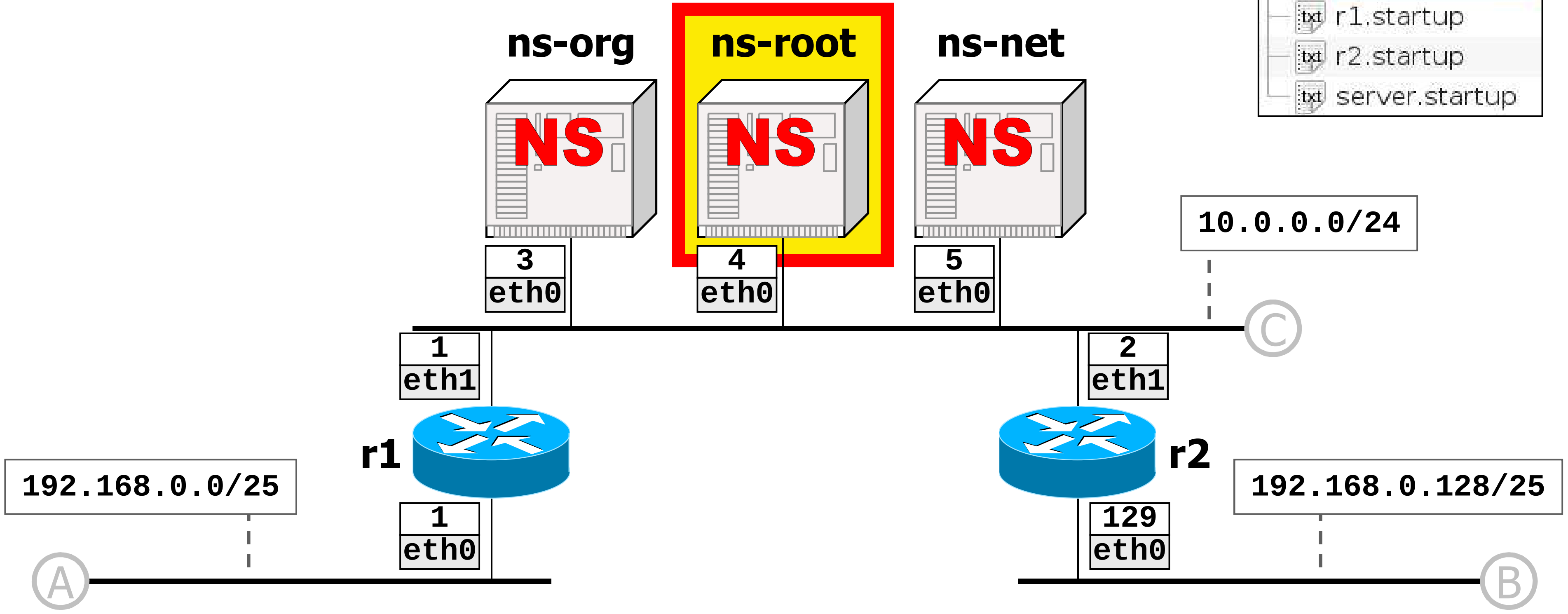
4. dns

ns-root.startup

```
ifconfig eth0 10.0.0.4 netmask 255.255.255.0 up

route add -net 192.168.0.0/25 gw 10.0.0.1 dev eth0
route add -net 192.168.0.128/25 gw 10.0.0.2 dev eth0
/etc/init.d/bind start
```


current lab contents

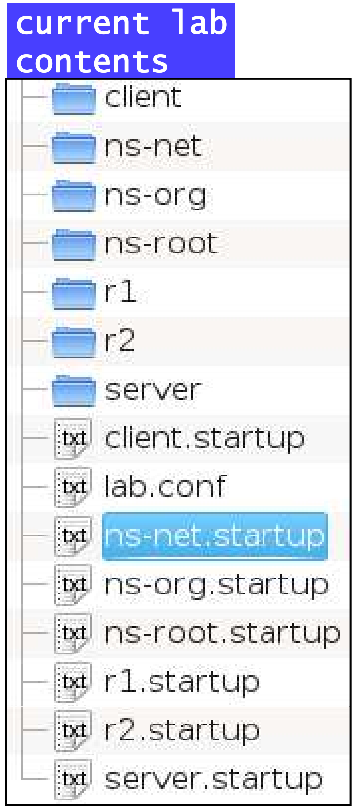
```
client
ns-net
ns-org
ns-root
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```



4. dns

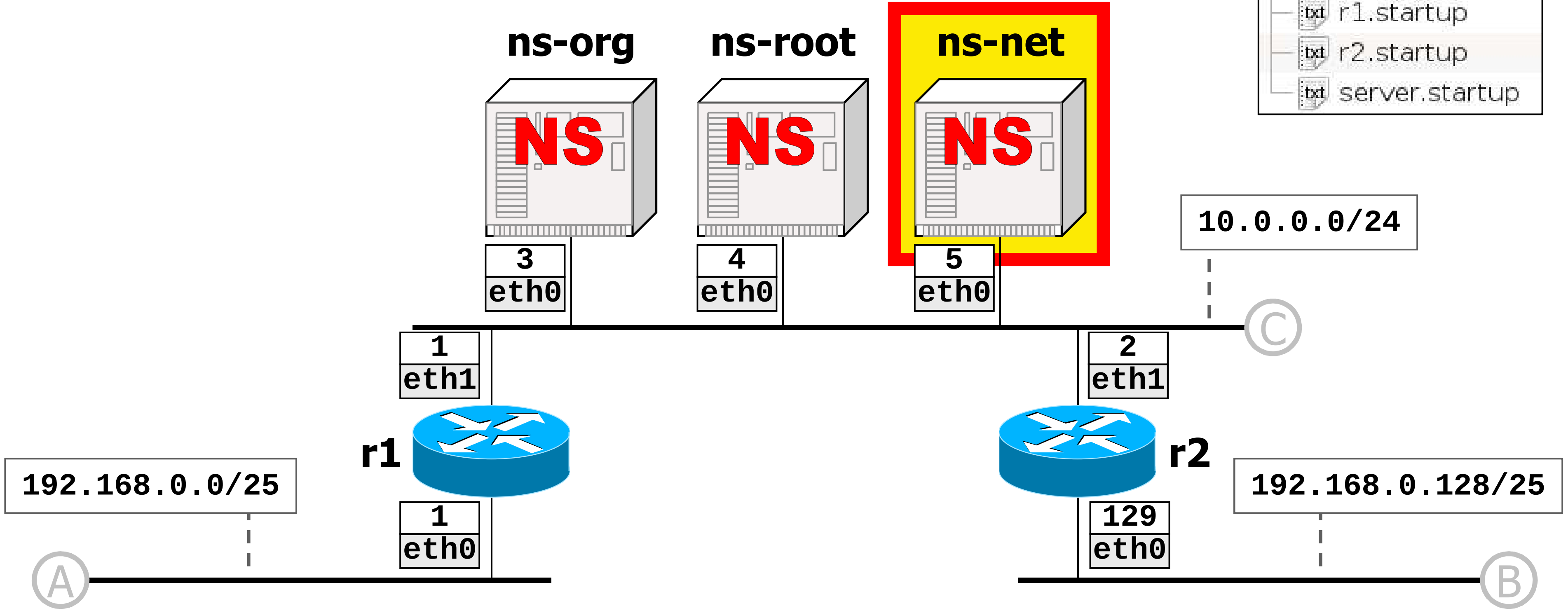
ns-net.startup

```
ifconfig eth0 10.0.0.5 netmask 255.255.255.0 up

route add -net 192.168.0.0/25 gw 10.0.0.1 dev eth0
route add -net 192.168.0.128/25 gw 10.0.0.2 dev eth0
/etc/init.d/bind start
```



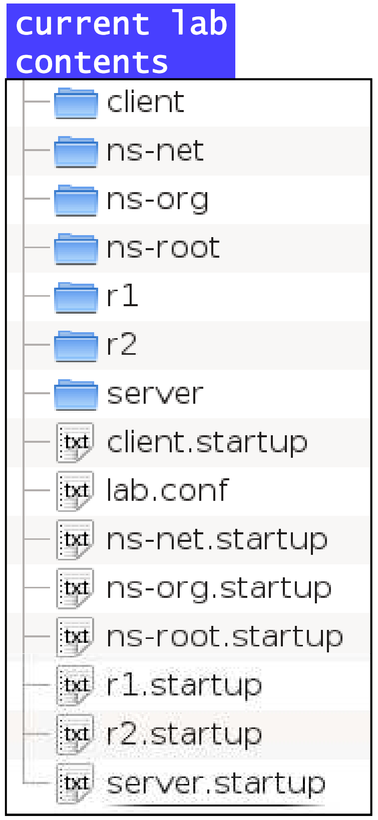
current lab contents

```
client
ns-net
ns-org
ns-root
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```



4. dns

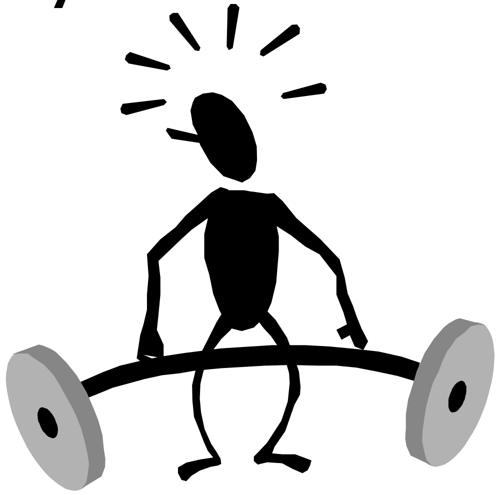
- now, we need to configure the dns service
- dns configuration consists of some files inside **`/etc/bind/`**
  - we create these files inside each virtual machine’s subdirectory
    - **`ns-org/etc/bind`**
    - **`ns-root/etc/bind`**
    - **`ns-net/etc/bind`**

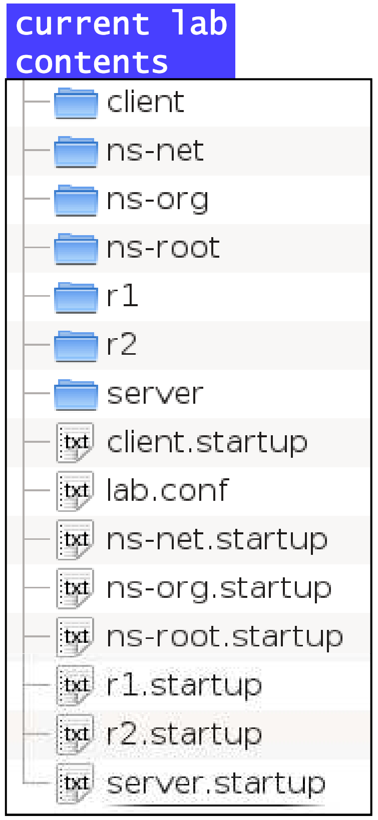

current lab contents

```
client
ns-net
ns-org
ns-root
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

4. dns

- but, hey... dns configuration is rather tricky!





current lab contents

```
client
ns-net
ns-org
ns-root
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

tip: copy files from the dns netkit lab and adjust them as needed

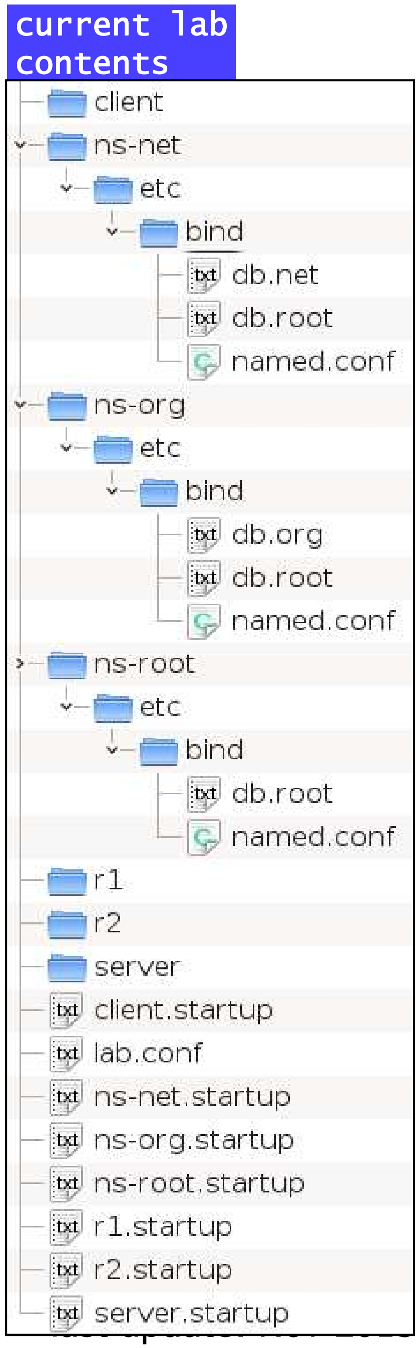
4. dns

- download the dns lab from the netkit web site
- copy files from the dns lab to our lab as follows:

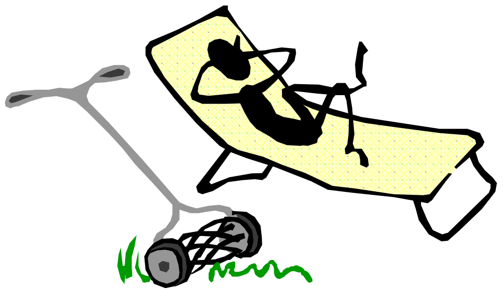
from (dns lab)	to (our lab)
`dnsroot/etc/bind/db.root` `dnsroot/etc/bind/named.conf`	`ns-root/etc/bind`

`dnsorg/etc/bind/db.org`
`dnsorg/etc/bind/db.root`
`dnsorg/etc/bind/named.conf`
	`ns-org/etc/bind`

`dnsnet/etc/bind/db.net`
`dnsnet/etc/bind/db.root`
`dnsnet/etc/bind/named.conf`
	`ns-net/etc/bind`


current lab contents

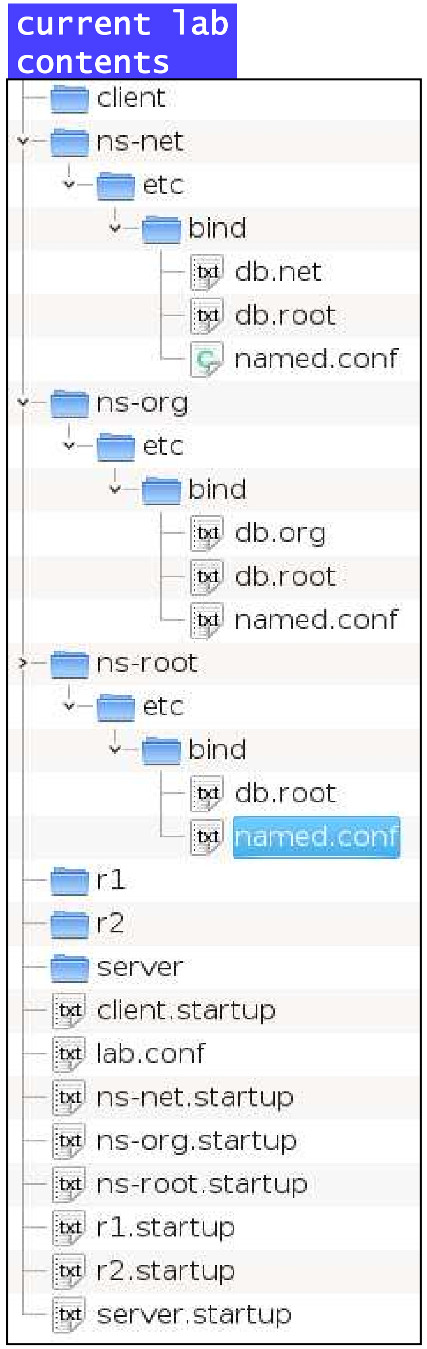
```
client
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```



4. dns

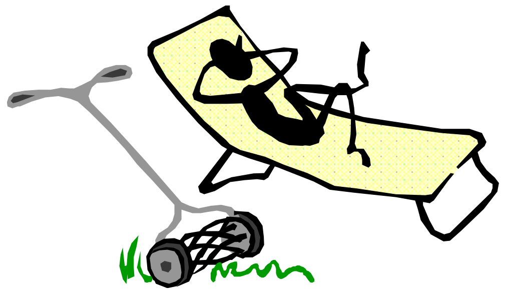
- trim **`named.conf`** contents to the essential

```
ns-root/etc/bind/named.conf

zone "." {
  type master;
  file "/etc/bind/db.root";
};
```



current lab contents

```
client
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```



4. dns

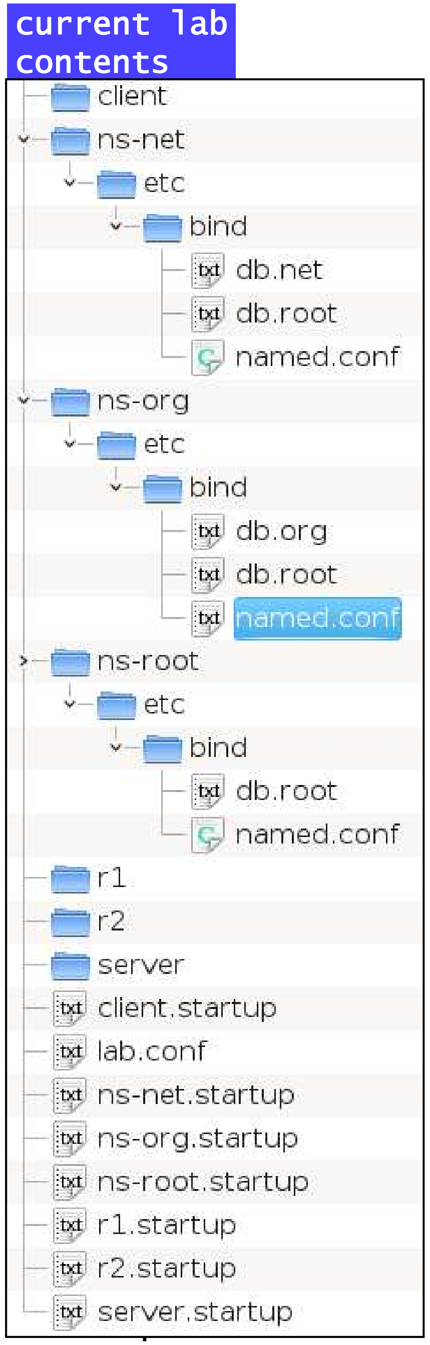
- trim **`named.conf`** contents to the essential and add the requested **`allow-recursion`** statement

ns-org/etc/bind/named.conf

```
options {
  allow-recursion {0/0; };
};

zone "." {
  type hint;
  file "/etc/bind/db.root";
};

zone "org" {
  type master;
  file "/etc/bind/db.org";
};
```

insert this additional statement in the configuration of **`ns-org`** and **`ns-net`**:

```
options {
  allow-recursion {0/0; };
};
```



current lab contents

```
client
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

4. dns

- trim **named.conf** contents to the essential and add the requested **allow-recursion** statement

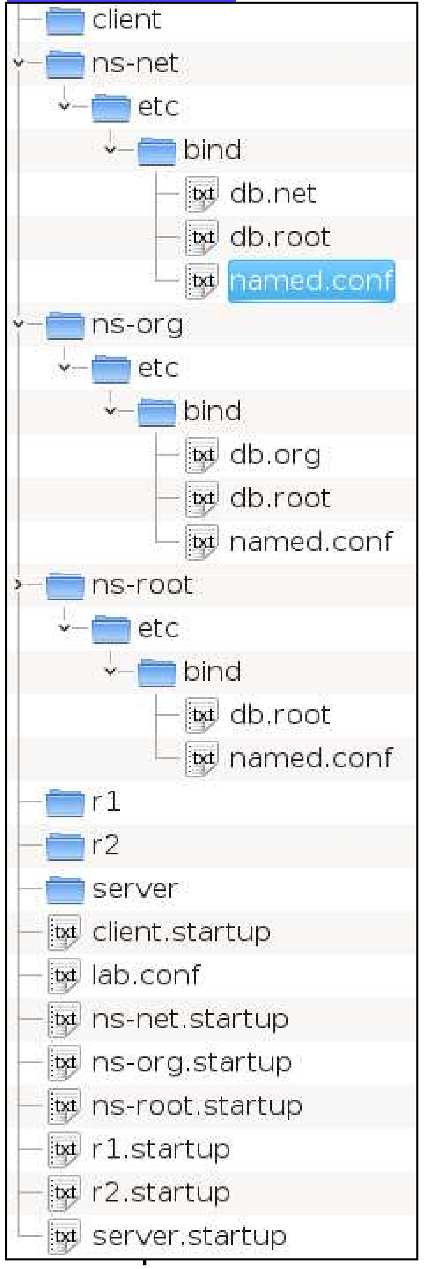
ns-net/etc/bind/named.conf

```
options {
  allow-recursion {0/0; };
};

zone "." {
  type hint;
  file "/etc/bind/db.root";
};

zone "net" {
  type master;
  file "/etc/bind/db.net";
};
```

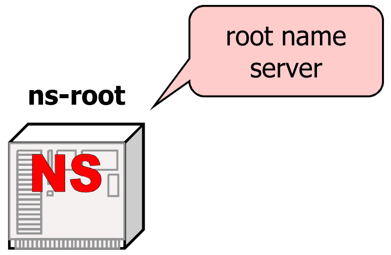
insert this additional statement in the configuration of **ns-org** and **ns-net**:

```
options {
  allow-recursion {0/0; };
};
```

current lab contents



```
client
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```



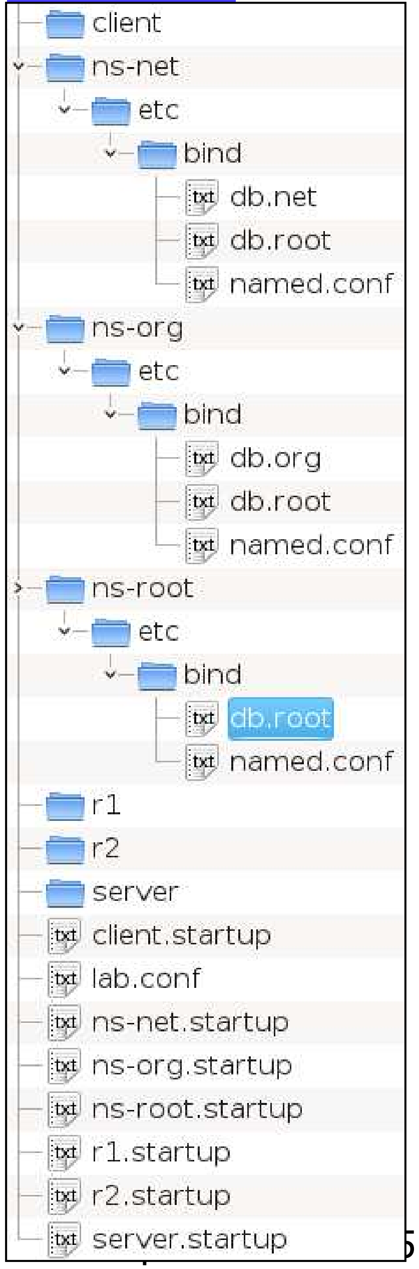
4. dns

- configure authoritative information
  - on **`ns-root`** we just need to update:
    - the address of the root name server
    - the address of the delegated name servers

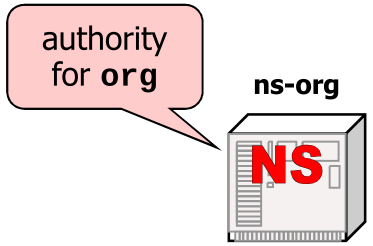
ns-root/etc/bind/db.root

```
$TTL    60000
@                IN SOA   ROOT-SERVER.   root.ROOT-SERVER. (
                              2006031201 ; serial
                              28800 ; refresh
                              14400 ; retry
                              3600000 ; expire
                              0 ; negative cache ttl
                          )

@                IN NS    ROOT-SERVER.
ROOT-SERVER.     IN A     10.0.0.4

org.             IN NS    dnsorg.org.
dnsorg.org.      IN A     10.0.0.3

net.             IN NS    dnsnet.net.
dnsnet.net.      IN A     10.0.0.5
```

current lab contents



```
client
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```



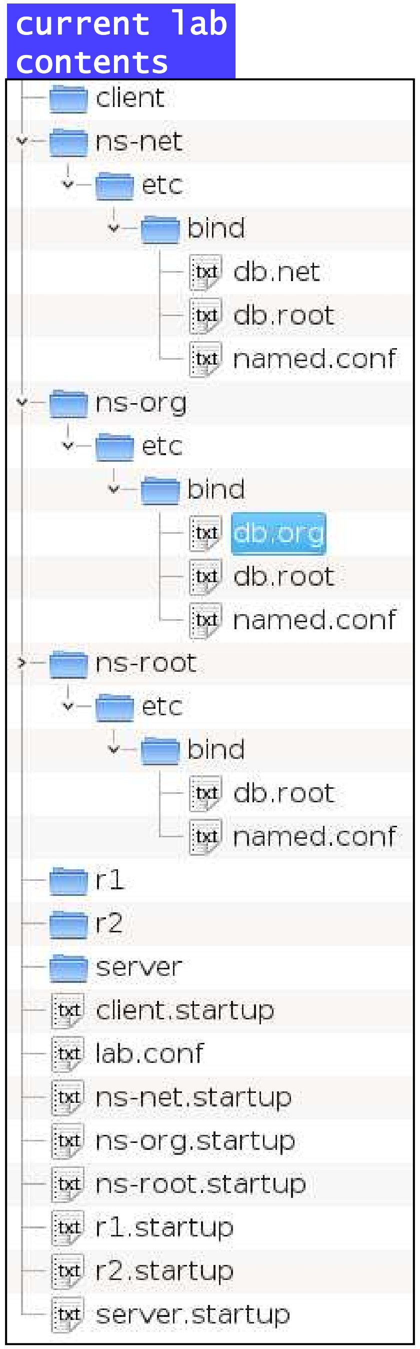
4. dns

- configure authoritative information
  - on **ns-org** we need to:
    - update the address of the authority for **org**
    - add a record for the **client** machine
    - discard all the rest (we have no further delegations in this lab)

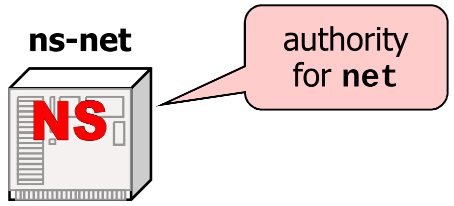
ns-org/etc/bind/db.org

```
$TTL    60000
@           IN SOA  dnsorg.org.  root.dnsorg.org. (
                        2006031201 ; serial
                        28800 ; refresh
                        14400 ; retry
                        3600000 ; expire
                        0 ; negative cache ttl
                    )
@           IN NS   dnsorg.org.
dnsorg      IN  A   10.0.0.3

client      IN  A   192.168.0.111
```

has **client.org** as DNS name


current lab contents

```
client
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

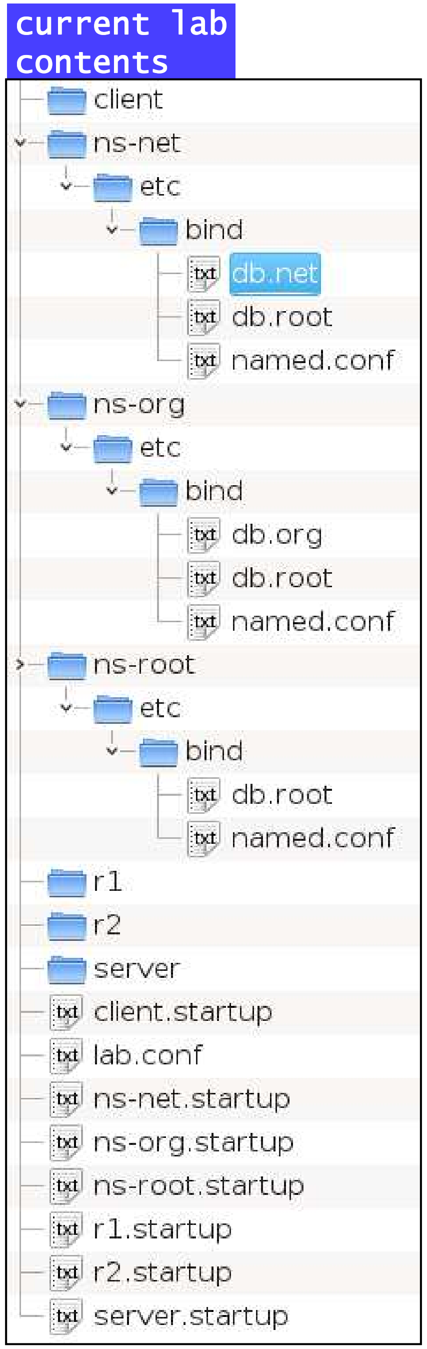


4. dns

- configure authoritative information
  - on **ns-net** we need to:
    - update the address of the authority for **net**
    - add a record for the **server** machine
    - discard all the rest (we have no further delegations in this lab)

ns-net/etc/bind/db.net

```
$TTL    60000
@         IN SOA  dnsnet.net.  root.dnsnet.net. (
                     2006031201 ; serial
                     28800 ; refresh
                     14400 ; retry
                     3600000 ; expire
                     0 ; negative cache ttl
                  )
@         IN NS   dnsnet.net.
dnsnet    IN  A   10.0.0.5

server    IN  A   192.168.0.222
```


current lab contents

```
client
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

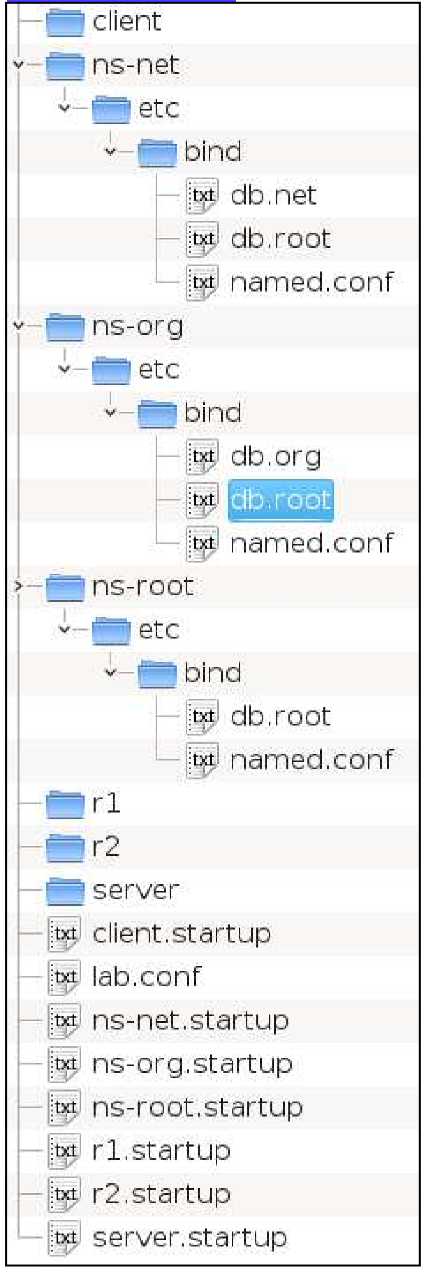
4. dns

- configure hints
  - on all the (non-root) name servers we need to update the address of the root name server

ns-org/etc/bind/db.root

```
.                IN   NS     ROOT-SERVER.
ROOT-SERVER.     IN   A      10.0.0.4
```

current lab contents



```
client
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

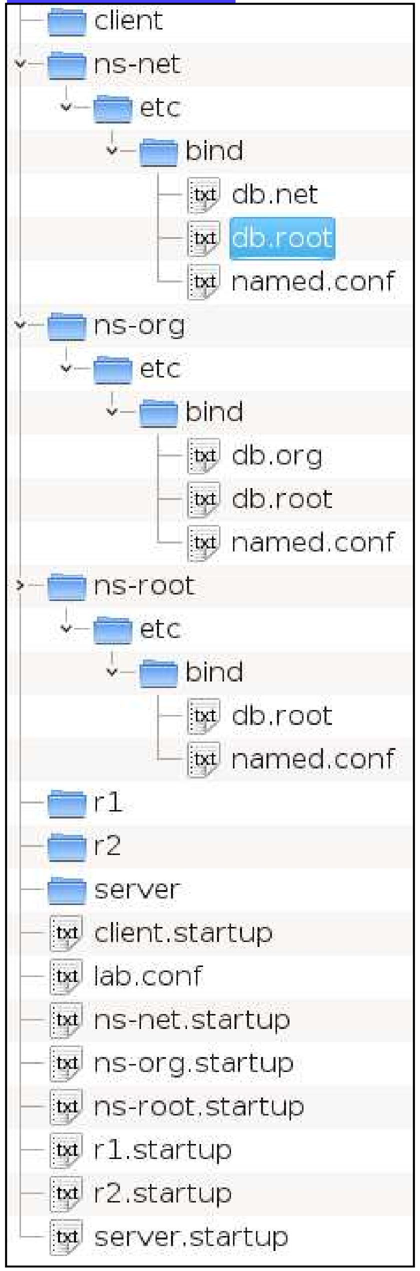
4. dns

- configure hints
  - on all the (non-root) name servers we need to update the address of the root name server

ns-net/etc/bind/db.root

```
.                IN   NS     ROOT-SERVER.
ROOT-SERVER.     IN   A      10.0.0.4
```

current lab contents



```
client
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

uses **ns-org** as local name server;

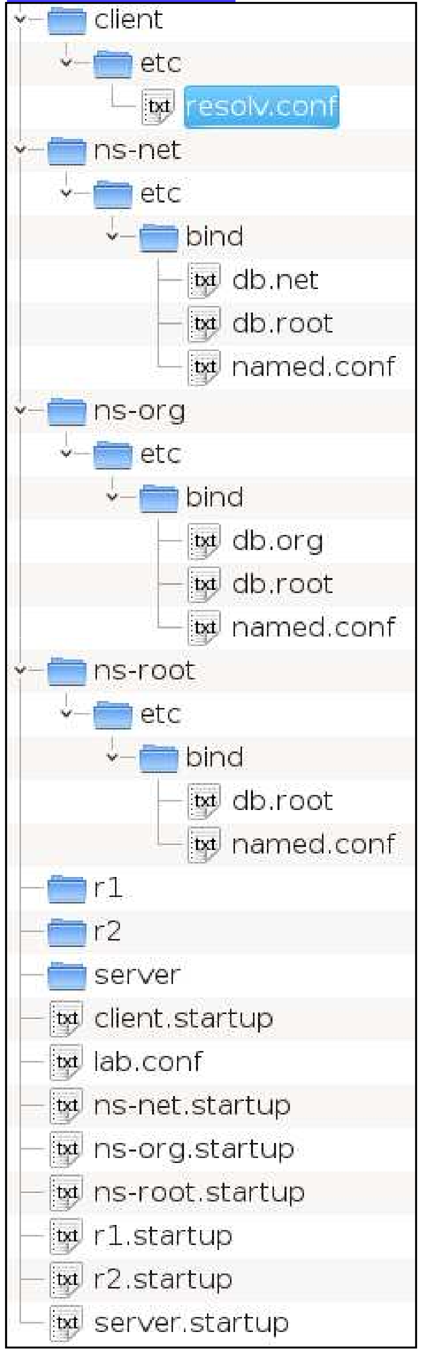
4. dns

- last, but not least
  - configure a resolver for **client**!
  - in Linux, this goes to **/etc/resolv.conf**, therefore we put it in **client/etc/resolv.conf**

client/etc/resolv.conf

```
nameserver 10.0.0.3
```

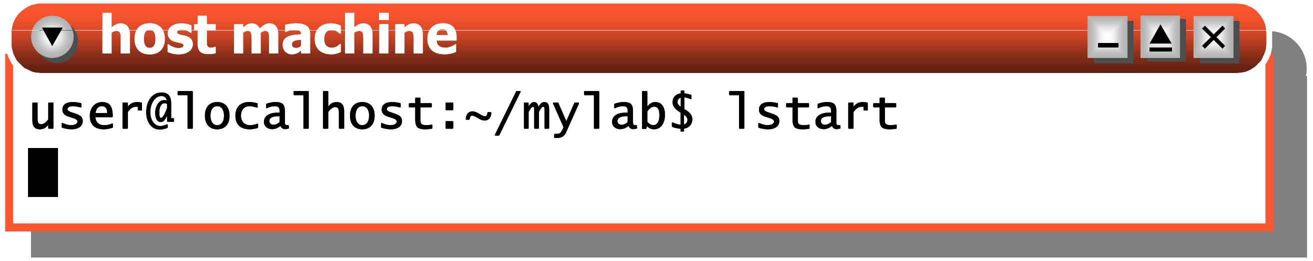
current lab contents



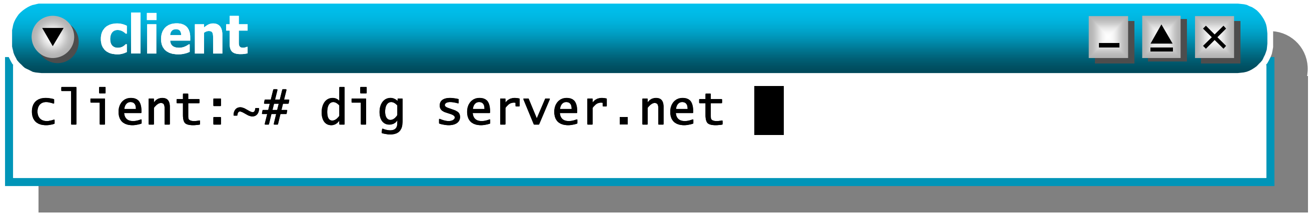
```
client
  etc
    resolv.conf
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

4. dns

- at this point it is a good idea to start the lab and check that the dns works

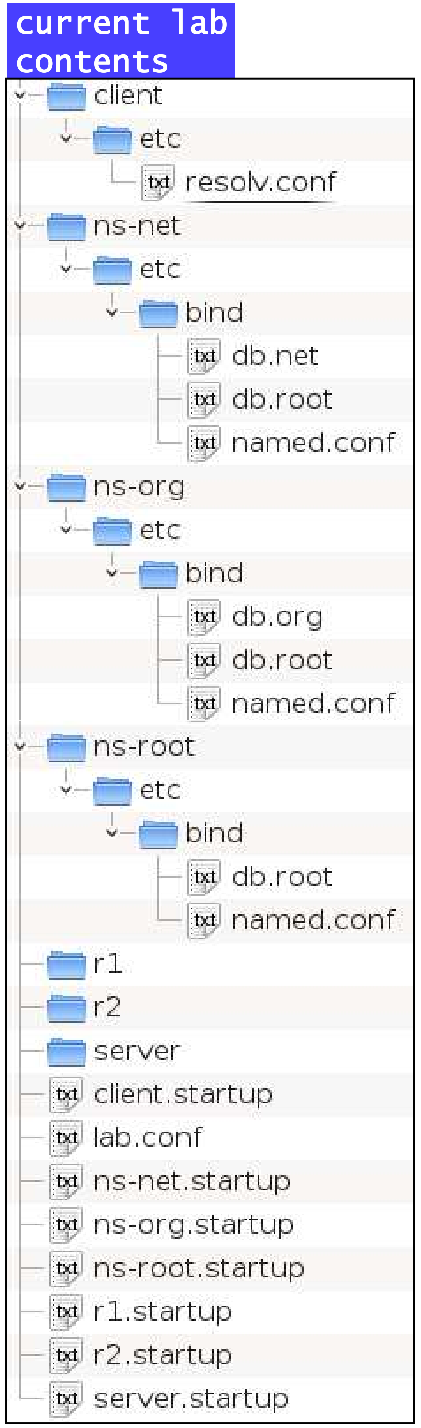


```
user@localhost:~/mylab$ lstart
```



```
client:~# dig server.net
```

current lab contents

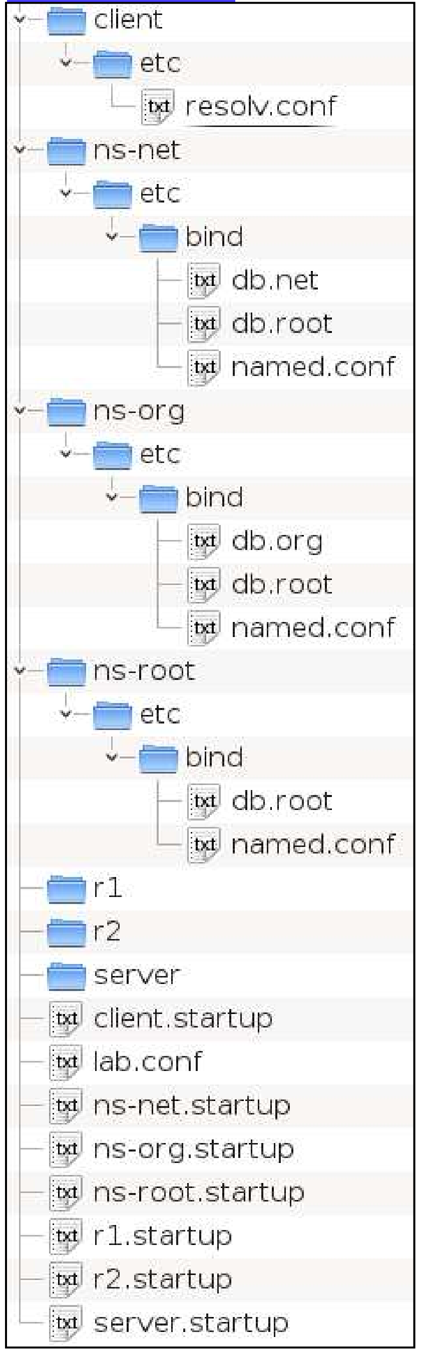


```
client
  etc
    resolv.conf
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

4. dns

- at this point it is a good idea to start the lab and check that the dns works
  - if it doesn't...
    - ...check boot-time virtual machine messages (errors printed in blue are relevant) to see if **`bind`** has failed starting
    - ...check **`/var/log/syslog`** (that's where **`bind`** logs its errors)
    - ... query for intermediate information (e.g., from **`client`** perform an iterative query to get the address of the root name server)
    - ...

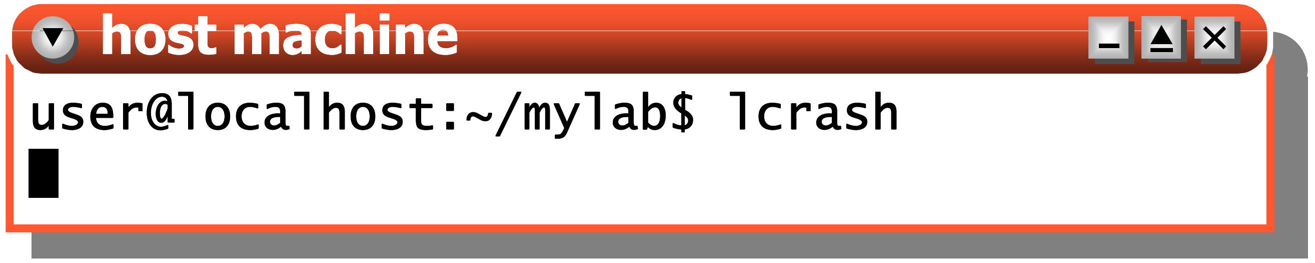
current lab contents



```
client
  etc
    resolv.conf
ns-net
  etc
    bind
      db.net
      db.root
      named.conf
ns-org
  etc
    bind
      db.org
      db.root
      named.conf
ns-root
  etc
    bind
      db.root
      named.conf
r1
r2
server
client.startup
lab.conf
ns-net.startup
ns-org.startup
ns-root.startup
r1.startup
r2.startup
server.startup
```

done

- that’s it!



```
user@localhost:~/mylab$ lcrash
█
```

- after stopping it, the lab can be packed in a tar.gz file for redistribution