Computers Networks

Laboratory exercises

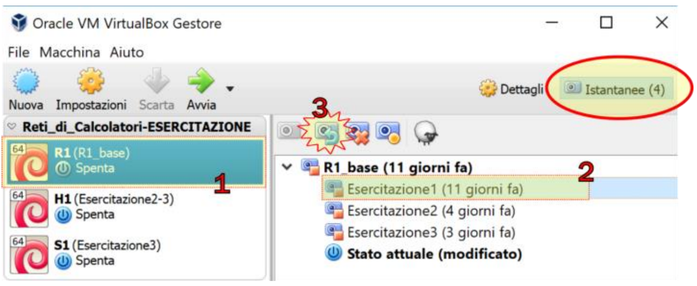
The aim of following exercises is to get awareness about an elementary network architecture, setting up basic features and services constitutive the main skeleton of modern computers networks: routing table, DHCP, DNS, web and mail servers. Contextually a network analysis is proposed due to observe more in details the behaviour of the network protocols involved.
The exercises are "independent" each others, but their sequencing is recommended, initial configuration of an exercise is the final of the previous one.

Basic information

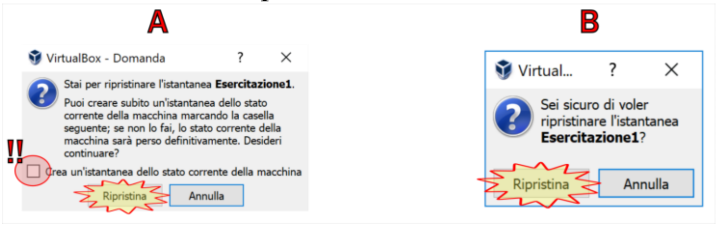
Exercises are based on a virtual environment set-up with VirtualBox©.
Each exercise uses its own hardware and software configuration saved in appropriate snapshots of the guest[1] systems, this configuration **must not be modified** for the correct progress of tasks.

To start the training, it is necessary activate for all systems the snapshot relative to the desired exercise.

1. Select the VM (e.g. *R1*) and the section *istantanee/snapshot*.
2. Select the snapshot of the desired exercise (e.g. *Esercitazione1*)
3. Click on *Ripristina istantanea/snapshot*.


Oracle VM VirtualBox Gestore
File Macchina Aiuto
Nuova Impostazioni Scarta Avvia
Dettagli
Istantanee (4)
3
Reti_di_Calcolatori-ESERCITAZIONE
R1 (R1_base) Spenta
1
H1 (Esercitazione2-3) Spenta
S1 (Esercitazione3) Spenta
R1_base (11 giorni fa)
2
Esercitazione1 (11 giorni fa)
Esercitazione2 (4 giorni fa)
Esercitazione3 (3 giorni fa)
Stato attuale (modificato)

4. Depending on the current state of the guest system selected, two different dialog boxes could be popped-up. In case "A" **clear** the option "*Crea istantanea dello stato corrente della macchina*", then click on "*Ripristina*"

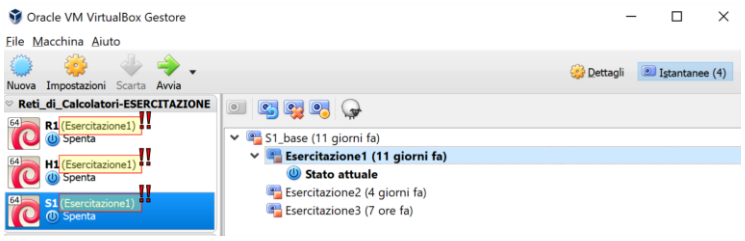


Stai per ripristinare l'istantanea **Esercitazione1**.
Puoi creare subito un'istantanea dello stato corrente della macchina marcando la casella seguente; se non lo fai, lo stato corrente della macchina sarà perso definitivamente. Desideri continuare?
☐ Crea un'istantanea dello stato corrente della macchina

Sei sicuro di voler ripristinare l'istantanea **Esercitazione1**?

[1]In the scope of virtualization , physical system is denoted as 'host' while virtualized system is denoted as 'guest'; the term 'host' however will be used to indicate a generic system either physical or virtualized.

5. Repeat previous steps for the others VMs, all VMs must be referred to the same configuration number as example in picture.


Oracle VM VirtualBox Gestore
File Macchina Aiuto
Nuova Impostazioni Scarta Avvia
Dettagli
Istantanee (4)
Reti_di_Calcolatori-ESERCITAZIONE
R1 (Esercitazione1)
Spenta
H1 (Esercitazione1)
Spenta
S1 (Esercitazione1)
Spenta
S1_base (11 giorni fa)
Esercitazione1 (11 giorni fa)
Stato attuale
Esercitazione2 (4 giorni fa)
Esercitazione3 (7 ore fa)

6. Boot-up all the VMs. It is possible to click with right button on group name *"Reti di Calcolatori-ESERCITAZIONE"* and select *"Avvia → Avvio normale"*

It is possible do the training with own laptop, just install VirtualBox and download from Ladispe web site section "*Corsi -> Reti di Calcolatori*" the archive of preconfigured VMs. During the import in VirtualBox, if required, pay attention to **NON initialize the MAC address** of the network interfaces.

General description

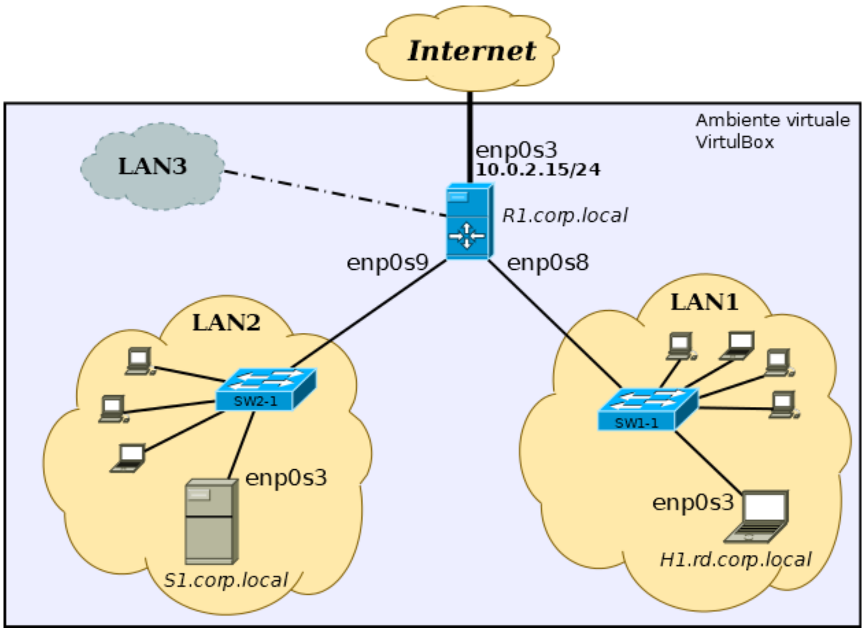
Three guest systems **R1, H1, S1**, connected each others with a virtual network are used for the exercises. The systems are *Linux Debian 9.x* and it is possible to login with the following accounts: ***root, user1, user2*** with password ***"networks"*** for all. For simplicity it is possible doing the training by accessing systems with *root* credential, but best practices do not recommend this kind of approach, both *user1* and *user2* can invoke commands with administrative privileges using *"sudo"*, for example:

```
$ sudo wireshark &
```

Note: the character "*$*" o "*#*" at the beginning of the command line specifies the privileges of the current logged-in user: "*$*" corresponds to a user with reduced privileges, while "*#*" specifies the prompt for user root.

The **C:\Temp** directory of the physical host is mounted automatically by guest systems at mountpoint **/media/sf_temp** allowing data exchange between systems.

The picture shows the network topology used in the exercises.



The router R1 allow the connection to the “real” network through NAT service provided by VirtualBox.
For every network interface of the guest systems the following table shows the name as defined by VirtualBox, the name as defined by the O.S. ,the MAC address and the network where the interface is connected to.

	Interfaccia VB/S.O.	**MAC**	**LAN**
R1	Scheda1/enp0s3	aa:aa:aa:aa:00:00	VBox NAT
	Scheda2/enp0s8	aa:aa:aa:aa:00:11	LAN1
	Scheda3/enp0s9	aa:aa:aa:aa:00:22	LAN2
H1	Scheda1/enp0s3	cc:cc:cc:cc:00:00	LAN1
S1	Scheda1/enp0s3	ee:ee:ee:ee:00:00	LAN2

In the guest system the list of the network interfaces is got executing:

```
user1@r1:~$ ip link show
```

In Debian 9.x network interfaces follow a denomination schema based on their typology and their connection to the system (ethernet, wireless, embedded on motherboard, on PCIex slot…), in this configuration interface name has pattern: ***enpXsY***[2] where X,Y are numbers based on the system configuration and can be inferred from the output of command **lspci**.

[2]For further details see: https://www.freedesktop.org/wiki/Software/systemd/PredictableNetworkInterfaceNames/

EXERCISE 0: Traffic Analysis

By looking at the configuration information of your Windows host (use the `ipconfig` command), find out: the IP address of the host, its netmask and the IP address of the default gateway.
Capture and filter (through proper filters) the traffic resulting from your station due to execution of the command:

```
ping [address]
```

where `[address]` is the IP address of another PC in the lab.
Make sure, prior to execution, that the ARP cache is empty. To check the saved addresses on the local machine run:

```
arp -a
```

To delete all the prior addresses, you need the execute the command:

```
arp2 -d
```

(*The usual command is* `arp -d` *with the console run as administrator. Due to access privileges issues in flushing ARP cache of laboratory's machines, use the* `arp2` *command from* `c:\windows\`, *instead of the traditional* `arp` *command)*

Note: use an IP address (e.g., 10.2.1.3), not a name (www.polito.it).

Questions

1. Looking at the network configuration of your station, which is the network part of the IP address? Which is the host part?
2. Observe the captured packets and identify the set of packets not related to the ping execution. Use a proper Wireshark filter to display only the packets interesting for this exercise.
3. Observe captured packets and describe the purpose of the main IP header fields (version, source and destination addresses, TTL, ToS). How is the protocol of the packet carried in the payload field identified? Is there any fragmented packet? Why?
4. Which are the two packets related to the execution of the ping command? What is the purpose of these packets? Explain to which host the source and destination MAC addresses of those packets belong to.
5. Is the source MAC address in the second IP packet the same as the destination MAC address of the first IP packet? Why?
6. What is the Destination IP address in the first packet?
7. What is the purpose of the Target IP Address field of the first IP packet?

(Second Part)

Using the same configurations given in the first part, repeat the ping command by targeting an IP address external to the laboratory network.
Capture, using appropriate filters, the traffic generated by your station following the command:

```
ping 8.8.8.8
```

Make sure, before entering the command, to clear the station's ARP cache. (`arp2` command as described above).

Questions

1. What are the IP source and IP destination of the captured ICMP packets?
2. What are the MAC source and MAC destination of the captured ICMP packets?
3. To whom does the destination MAC address of the generated ICMP packets match?
4. Indicate which are the main differences between the packets generated in the first part.
5. Indicate the MAC address of 8.8.8.8

(Third Part)

Start the *R1* virtual machine and repeat the ping command by targeting an IP address external to the laboratory network.
Capture, using appropriate filters, the traffic generated by your station following the command:

```
ping -c 4 8.8.8.8
```

To clear the station's ARP cache, run `arp -d,` executable as root on Linux systems.

Questions

1. What are the IP source and IP destination of the captured ICMP packets?
2. What are the MAC source and MAC destination of the captured ICMP packets?
4. Which are the main differences between the packets generated in the second part?

NOTE: If the student deems relevant, he/she can report in the following lines the content of the packets captures.

Capture (First Part)

```
ping [ip address of lab machine]
```

N.	L2	L3	Payload Description
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			

Capture (Second Part)

```
ping 8.8.8.8
```

N.	L2	L3	Payload Description
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			

Capture (Third Part)

```
Ping -c 4 8.8.8.8
```

N.	L2	L3	Payload Description
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			

EXERCISE 1

This exercise contemplates:

- study of the address space for a simple Ethernet network given certain constraints
- configuration at IP level of the systems
- brief analysis of the network traffic

Boot up the three nodes

Part one

With the following range of IP addresses 192.168.0.0/16, implement a network according to the provided topology and with following specification:

- hosts are connected by switch.
- LAN1 has to contain about 100 hosts.
- LAN2 has to contain about 50 hosts.
- It will be possible add a third LAN (LAN3) for 50 more hosts.
- Use the address space that achieve the minimal "waste" of IP.
- Assign to LANs default gateways the last available IP of their own subnet.
- Assign to hosts H1 and S1 the first available IP of their own subnet.
- R1 is the DHCP server for LAN1 and LAN2.

Asses the CIDR address space for the three subnets and assign IP address to systems

For the whole three subnets, 200 IP address are needed, therefore a network in /24 of the provided range in /16 is enough. For example, we can choose the first net /24 available i.e. 192.168.0.0/24 and we divide it in three subnets. For addressing the 100 hosts in LAN1 7 bit of the host section are needed, thereby we use a /25 net. The remaining range has to be divided in two networks of 50 host each, therefore we need 6 bits in the host part of IP address, that is two /26 net.

	Net	**Range**	**Netmask**
LAN1	192.168.0.0/25	192.168.0.0 192.168.0.127	255.255.255.128
LAN2	192.168.0.128/26	192.168.0.128 192.168.0.191	255.255.255.192
LAN3	192.168.0.192/26	192.168.0.192 192.168.0.255	255.255.255.192

IP address(CIDR)		
	NIC	**IP**
R1	enp0s3	10.0.2.15/24 (*VBox dhcp, do NOT modify*)
R1	enp0s8	192.168.0.126/25
R1	enp0s9	192.168.0.190/26
H1	enp0s3	192.168.0.1/25
S1	enp0s3	192.168.0.129/26

Hosts network configuration[3]

The router R1 that provides DHCP service has its own interfaces with a static configuration, while other hosts H1 and S1 have interfaces with DHCP configuration.
Network settings are defined in the file `/etc/network/interfaces`.
Edit[4] the file on all systems, every network interface must have its own section in the file. IP address assignment (in static or dynamic mode) is achieved by defining following settings:

STATIC IP	**DHCP**
auto ***enp0s3*** iface ***enp0s3*** inet static address ***10.10.0.62*** netmask ***255.255.255.0***	auto ***enp0s3*** iface ***enp0s3*** inet dhcp

Note: in bold-italic the data that have to be proper arranged for the specific configuration (for this and succeeding examples)

For more detailed information see the *man page* of the file (`man interfaces`)
To apply the new configuration run the following chain of commands (specify the right netwok interface to be updated, in this example *"enp0s8"*):

```
# ifdown -v enp0s8 && ip addr flush dev enp0s8 && ifup -v enp0s8
```

Alternatively **before**[5] edit the file stop the network services with one of the following commands:

```
# service networking stop
# systemctl stop networking.service
```

After editing start again network services with one of the following commands:

```
# service networking restart
# systemctl restart networking.service
```

To activate/deactivte a single network interface (e.g. *"enp0s3"*) run:

```
# ifup/ifdown enp0s3
```

Verify if the configuration is orrect with one of the following commands:

```
# ip addr show   (suggested)
# ifconfig
```

To allow clients achieving its own network configurations, it is necessary set-up the DHCP server.

DHCP server configuration

The DHCP service runs on R1 and the service daemon is ***dhcpd***. Edit the configuratio file `/etc/dhcp/dhcpd.conf` and update it with the right parmeters: specify the IP address ranges and others options in accord with the address space and the IP assignment previous defined, grant to H1 and S1 a "*reservation*" to permit at the DHCP server always to gives the same IP address to these clients, pay attention to the MAC addresses needing for the reservation.

Note: IP range of addresses provided by DHCP **must not** include neither the IP of reservations nor eventual static IP configured on hosts.

Restart DHCP service with one of the following commands:

---

[3]The networks interfaces can be configured in different ways other than the ones here illustrated.
[4]Administrative privileges are needed to edit the file. The following editors are available: vim, nano, leafpad
[5]Otherwise the old IP may not be deleted getting an incorrect network configuration and only by the command "`ip addr show`" it is possible to see the assignment of two IP address to the interface.

```
# service isc-dhcp-server restart
# systemctl restart isc-dhcp-server.service
```

If there is no output the command successful, file syntax is correct and dhcp service is running

```
# cat /var/log/syslog
# tail [-f] /var/log/syslog
```

netstat command shows information about current network connections. Verify that dhcp service is running on R1: identify addresses, protocols and ports used[6] by dhcp daemon.

```
[root@r1 ~]# netstat -anup |grep dhcpd
udp        0            0 0.0.0.0:12165        0.0.0.0:*            1938/dhcpd
udp        0            0 0.0.0.0:67           0.0.0.0:*            1938/dhcpd
udp6       0            0 :::53949             :::*                 1938/dhcpd
```

Check configuration

To be sure that dhcp clients have received the correct network configuration restart (on clients) the *networking* service and verify with command ifconfig or alternatively this command:

```
# ip addr show
```

If more than one IP is present for an interface, it is necessary remove the IP not used, e.g. removing of IP *192.168.0.180* from interface *enp0s9*

```
# ip addr del 192.168.0.180/26 dev enp0s9
```

Network analysis

Before proceeding be sure to empty the arp cache, in order to visualize all the traffic to be generated. On all nodes, run from root:

```
# arp -a
```

delete eventual items present with:

```
# arp -d N.N.N.N
```

where *N.N.N.N* represents the IP address associated at arp item to be removed.
Launch Wireshark on R1 in order to catch traffic on network interfaces of LAN1 and LAN2, do a ping against H1 and S1 specifying their IP address

```
$ ping 192.168.x.x
```

1. Which couple of packets are generated invoking the ping command? Specify the owner of MAC address both source and destination for captured packets

In this order the couples of ethernet frame are generated

1.	ARP request:	MAC src=R1	MAC dst=H1|S1
2.	ARP reply:	MAC src=H1|S1	MAC dst=R1
and			
3.	ECHO request:	MAC src=R1	MAC dst=H1|S1
4.	ECHO reply:	MAC src=H1|S1	MAC dst=R1

First pair allows at hosts to exchange each other's information about their own MAC address.
Second couple is the ICMP communication and it is repeated for arbitrary times.

[6]La presenza di due porte UDP casuali oltre alla 67 (definita dalle specifiche di protocollo) è dovuta alle opzioni specificate in fase di compilazione del programma e sono relative a funzionalità DDNS (vedere man dhcpd)

2. For which reason are there others packets in addition to icmp protocol?

Assuming that in source node the arp cache is empty, a broadcast arp request is generated, before generating the echo request, to get the MAC address of the destination host.

At the end of ping also the destination host generates an arp request, but this time is unicast, against the MAC of source host of the ping, due to refresh its own arp cache.

3. Are MAC source of second packet and MAC destination of third packet the same? Why?

With the second frame the destination host transmits its MAC to the source host of the echo request. The source host can now compose the frame containing the IP+ICMP packet to be sent on the network.

4. Invoke a ping between H1 and S1. Which can be the cause of the missing connection?

Only the pair R1-H1 and R1-S1 can communicate because on the systems is not yet configured a routing table.

5. Analyze the captured traffic and specify the meaning of the main fields of an IP datagram (version, source and destination address, TTL, ToS). Which header field identify the payload protocol?Are there fragmented datagram? Why?

See the reference book of the course or RFC 791 and its updates https://tools.ietf.org/html/rfc791

Payload is identified from field PROTOCOL.

There are no fragmented datagram because the dimension of the header IP and the payload (ICMP protocol in this example) is less than the Ethernet MTU (1500 bytes)

6. Capture the traffic generated by a DHCP request, stopping and restarting network services on H1 or S1. It is recommended to run the capture on the opposite machine where the services are restarted.

Which IP (source and destination) are used to start the communication? Which transport protocol is being used? Specify the source and destination ports of the first datagram.
How many frames are needed to complete the operation of address assignment?

```
# service networking stop
# service networking start
```

The client senda a broadcast frame with source IP 0.0.0.0 and destination IP 255.255.255.255. The transport protocol is UDP with source port 68 and destination port 67.

To complete the operation 4 frames are needed.

Part two

Routing tables configuration

Display the routing tables of the three nodes with:

```
$ ip route show
```

or (only for root user):

```
# route -n
```

1. Wich hosts do need to configure their routing table? Why?

Only H1 and S1 need to configure the routing table with the adding the default gateway.

In fact, R1 knows all the networks which is connected to and its default gateway is provided from VirtualBox NAT. In our configuration, the DHCP server on R1 provides only the IP address to clients, consequently the gateway and the routing table have to be manually configured.

To add an entry in the routing table invoke[7]:

```
# ip route add IP_NETWORK/XX via IP_GW dev enpXsY
```

- *IP_NETWORK/XX* : the network to reach, in CIDR format
- *IP_GW* : the IP address of the gateway through who the network can be reached
- *enpXsY* : the network interface through the gateway can be reached

[7]Alternatively it is possible use the command 'route', see man page for its syntax and details

Configure the table of guests H1 and S1 to allow routing between the two networks

```
H1:     # ip route add 192.168.0.128/26 via 192.168.0.126 dev enp0s3
S1:     # ip route add 192.168.0.0/25 via 192.168.0.190 dev enp0s3
```

Make **only one** ping request (with option "***-c 1***") between H1 and S1 and capture the traffic that flows through R1 selecting the network interfaces *enp0s8*, *enp0s9*.

2. Why despite the option "*-c 1*" two frames with *echo request* and two with *echo reply* are present? List the reasons of this behaviour and the differences between frame of the same type.

The first frame *echo request* captured on *enp0s8* is the direct delivery make by H1 to its next hop, the TTL is 64, the frame ethernet has source MAC of H1 and destination MAC of R1, moreover the frame has its own checksum value.

The second frame is captured on *enp0s9*: R1 reads its own routing table and makes the direct delivery of frame to the destination host S1, the TTL is decremented by 1 and it is 63, the frame ethernet has source MAC of R1 and destination MAC of S1, moreover the frame has its own checksum value.

Similar considerations can be done for the answer frames of *echo reply*.

With root privileges run on H1 the following command and capture the traffic flowing on interfaces *enp0s8* and *enp0s9* of R1

```
# traceroute -I -q 1 192.168.0.129
```

3. Analyze the capture and give a curt description of how the command works

traceroute tries to outline the path of a datagram from source to destination, listing the crossing router. To achieve this behaviour the program makes a series of ping request starting with a TTL=1 for the first datagram and incrementing the TTL one by one for the following sent packets. With TTL=1 the first packet it will be discard at first hop (the default gateway), so the second packet, with TTL=2 will be discard at second hop and so on till the destination. Every router, where the datagram has TTL=0, if it is opportunely configured, answers to the source with a packet icmp type 11 (Time To Live exceed), from whom the source knows the route of the packet originated.

Verify that the traffic originated by systems H1 and H2 is not routed to other destinations than LAN1 and LAN2, doing a ping against external hosts (e.g. 8.8.8.8).
Update the routing table of nodes H1 and S1 to allow the routing to other destinations (tip: generic destination is called "*default*")

```
H1:     # ip route add default via 192.168.0.126 dev enp0s3
S1:     # ip route add default via 192.168.0.190 dev enp0s3
```

Display the new routing table and verify external connectivity.

4. Is it possible simplify this new routing table?

H1 and S1 have only one network interfaces, hence just one default route can be specified for all destinations and routing to LAN1/LAN2 is not necessary

5. Repeat the *traceroute* command against destination 8.8.8.8 and capture the generated traffic.
   - Do all routers crossed respond with icmp packet of type 11 (TTL exceed)?

No, not all routers are configured for reply with ICMP packet TTL exceed.

Update DHCP server configuration

To provide at clients the correct routing table we need to update the DHCP server configuration file specifying for all the served subnets its own *default gateway*: add the following directive, with the right value, in sections *subnets*:

```
option routers 192.168.x.x;
```

Part three: a real network

Using the physical host of the laboratory, make a capture of the traffic on the ethernet interface, due to observe the behaviour of a real network.
What does change from the simulate exercise situation?

In a real network, lots of services are running on a lot of hosts (e.g. authentication and connection to network shares) and continuous traffic is being generated to make all this working

Without see the network configuration of the laboratory host, but only by a capture with Wireshark, get the default gateway and the DNS of the host.

It is possible to make a ping request to, for example, *www.polito.it* that it is on an external network than the laboratory LAN, in order the host makes a DNS request for the name resolution of destination and consequently it sends an ethernet frame containing echo request to its default gateway.

EXERCISE 2

This exercise has the aim to show the behaviour of a simple DNS system and it contemplates:

- to associate a name space to the network under exam
- to configure for this name space an authoritative name server ables to forward to another external name server query that it can not solve
- brief network traffic analysis

Start up all three nodes

Part one

DNS[8]: brief overview

A DNS server can be configured in different ways:

- **authoritative primary server**: it provides and it keeps zones information that it is authoritative for, therefore it can change the DNS database of zone/zones under its own competence;
- **authoritative secondary server**: it provides information of zone/zones get by another authoritatve server for that/those zone/zones, in practice it is a replica of another authoritative server.
- **cache server**: it solves queries for others network hosts and it keeps a cache of results in order to dispatch subsequent requests, it does not hold zone competence.

A zone represents the name space (domain) managed by one or more servers (one primary and one or more secondary) that are defined authoritative for the zone.
The DNS system stores information in record called *resource record* **RR** consisting, in its most basic form, of three fields: *'Name, Type, Value'.* The content of *Type* field determines the meaning of the others two fields *Name* and *Value.* For example, a type "A" record associates to the alphanumeric string in the *Name* field the IP address in the *Value* field, making the translation possible.

```
      Name                     Type           Value
server01.polito.it.             A          192.168.10.10
```

A *CNAME* record associates to the string of the *Name* field the value in the *Name* field of a *A* type record, that is it provides an alias for a given hostname.

```
      Name                     Type           Value
www.polito.it.                 CNAME       server01.polito.it.
```

In addition to providing translation service from domain name to IP address the DNS system allows the so-called ***reverse resolution***, that is from a given IP address it allows to obtain the related hostname. In this situation IP addresses are handle like "labels" and they are considered to be *name* in a special domain called ***in-addr.arpa.*** A **PTR** type record (pointers to another part of the name space) is associated to each of these "labels" and its value is the real searched hostname. A PTR type record has for its *Value* field the content of the *Name* field of a A type record. The domain names of the *in-addr.arpa* zone represents the IP that needs translation, but it is specified with the octets in reverse order.

[8]Per maggiori dettagli si consultino RFC 1034,1035 e successivi.
Per un elenco di RFC relativo a DNS si consulti https://www.isc.org/community/rfcs/dns/

For example in the reverse search domain the name associated to the IP “*192.168.0.1*” is “*1.0.168.192.in-addr.arpa*” and its related *PTR* record is:

```
           Name                       Type           Value
1.0.168.192.in-addr.arpa.             PTR         name-host.domain.name.
```

and its related *A* type record in the “*domain.name*” domain is:

```
           Name                       Type           Value
name-host.domain.name.                A           192.168.0.1
```

Note: pay attention at the final ‘.’ (dot) of every value in the *Name* field.

The *in.addr.arpa* zone is, to all intents and purposes, a DNS domain like any others and it is handled by the DNS system without any differences, thanks to use of the reverse notation it is possible to trace back the A type record related.
To reverse solve the IP of network 192.168.0.0/24 the “reverse” DNS zone “*0.168.192.in-addr.arpa”* must be configured and populated with PTR type records to translate.

A DNS service implementation

In Debian 9.x the DNS server is BIND9[9] (ver. 9.10.3) by Internet Systems Consortium (ISC), one of the most popular DNS server.

A name space has to be associated to the network configured during the first exercise. The network is identified with the domain **“corp.local”** and S1 is the authoritative server for the zone.
The following table shows IP, names and services assigned to the hosts: define the type of resource records needed to populate the DNS. Identify also the reverse search zones and their PTR record associated.

IP	DNS name	SERVICE
10.0.2.15 192.168.0.126 192.168.0.190	r1.corp.local	
192.168.0.1	h1.rd.corp.local	
192.168.0.129	s1.corp.local www.corp.local intra.corp.local corp.local mail.corp.local pop3.corp.local	 web web mail (send/receive) mail (send/receive) mail (reading)

Notes and tips.

All the IP showed must be associated, with the right kind of resource record, to the values showed in the contiguos cell (*DNS name* field)
Column SERVICE shows the services associated to the *DNS name* of the related row.
It is possible to associate more than one A type record to a DNS name (e.g. in case of multihomed systems).
Every A type record must have its corresponding PRT type record in the reverse search domain *in-addr.arpa*.
S1 is the authoritative server (every zone contains a NS type record that specifies its own authoritative server):
for zone: *corp.local*
for the following reverse search zone: *0.168.192.in-addr.arpa* *2.0.10.in-addr.arpa*
An MX type record *must* have associated to a A type record, that is the VALUE field must contain a string corresponding to the NAME field of a A type record, in any case it must never have the value of the *Name* field of a CNAME[10] type record.

[9]https://www.isc.org/downloads/bind/ for software and manuals or on the Ladispe web site section *Corsi - Reti di Calcolatori*
[10]See RFC 2181 section 10.3 (https://tools.ietf.org/html/rfc2181#section-10)

The field *Name* of a CNAME type record <u>*must not coexist*</u>[11] with other kinds of records, that is its content must be unique: it must not exist a CNAME record having in its *Name* field the same value of the *Name* field of a NS record.

RR for ***corp.local*** zone		
Name	**Type**	**Value**
corp.local.	NS	s1.corp.local.
corp.local.	MX	mail.corp.local.
r1.corp.local.	A	10.0.2.15
	A	192.168.0.126
	A	192.168.0.190
s1.corp.local.	A	192.168.0.129
mail.corp.local.	A	192.168.0.129
pop3.corp.local.	CNAME	s1.corp.local.
www.corp.local.	CNAME	s1.corp.local.
intra.corp.local.	CNAME	s1.corp.local.
h1.rd.corp.local.	A	192.168.0.1

RR for ***0.168.192.in-addr.arpa*** zone		
Name	**Type**	**Value**
0.168.192.in-addr.arpa.	NS	s1.corp.local.
1.0.168.192.in-addr.arpa.	PTR	h1.rd.corp.local.
129.0.168.192.in-addr.arpa.	PTR	s1.corp.local.
129.0.168.192.in-addr.arpa.	PTR	mail.corp.local.
126.0.168.192.in-addr.arpa.	PTR	r1.corp.local.
190.0.168.192.in-addr.arpa.	PTR	r1.corp.local.
RR for ***2.0.10.in-addr.arpa*** zone		
2.0.10.in-addr.arpa.	NS	s1.corp.local.
15.2.0.10.in-addr.arpa.	PTR	r1.corp.local.

[11]See RFC 1912 section 2.4 (https://tools.ietf.org/html/rfc1912#section-2.4)

BIND9 configuration on S1

BIND9 is configured by a set of files normally stored in /etc/bind/ , these file are included by the main configuration file named.conf :
- named.conf.options: it contains the general configuration options f.i. the working directory, the listening interfaces and ports for the service;
- named.conf.local: it contains the configuration of the zones managed by the server;
- named.conf.default-zones: it contains the configuration af some default zones, for example the root-server;
- db.*: these are the zone files where the RRs are specified.

Edit named.conf.options:

- specify as listening IP addresses (*ip1* and *ip2*) the one configured for the network interface and the looback IP in the following directive:

```
listen-on { ip1; ip2; };
```

- enable the ability of the server to forward the query incoming from any host with the directives (a production environment MUST be more restrictive):

```
recursion yes;
allow-recursion { any; };
```

- set as unique forwarding DNS the following host *130.192.225.79* with these directives:

```
forward only;
forwarders { ip; };
```

- disable listening on IPv6 specifying *none* in the directive *listen-on-v6;*
- comment the directive *dnnsec-validation*

Edit named.conf.local:

- define zones for that **S1 is the authoritative server** (both direct and reverse search zones) and specify the file containing the RRs for that zones. Suit the following example for the desired configuration

```
zone "nome.di.zona" {
        // The server has a master copy of the data for the zone
        // and will be able to provide authoritative answers for it
        type master;
        // file with RRs
        file "/etc/bind/db.nome.di.zona";
};
zone "1.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/db.1.168.192.in-addr";
};
```

Create the zone files *db.** specified in the previous file and populate them with the correct RRs. Suit the following example to the configuration to implement, for each of the three zones.

```
$TTL    86400
@       IN      SOA         dns.example.org. root.localhost. (
                                         1           ; Serial
                                         604800   ; Refresh
                                         86400      ; Retry
                                         2419200  ; Expire
                                         86400 )    ; Negative Cache TTL
                       IN      NS      dns.example.org.
                       IN      MX      10 mail.example.org.
dns                    IN      A       192.168.1.1
                       IN      A       192.168.10.1
```

```
                        IN      A       10.0.1.1
mail                    IN      A       192.168.3.3
web01.example.org.      IN      A       192.168.3.2
                        IN      HINFO   Intel_Xeon
www.example.org.        IN      CNAME   web01.example.org.
```

Notes and tips [12].

The first symbol "@" represents the name of the zone as defined in the file *named.conf.local*.

The SOA (Start of Authority) record, whose name matches the zone name (because the first column is "@"), returns general information about the DNS zone. In this example the zone is *example.org*, *dns.example.org* is the main DNS server and *root.localhost* is the e-mail of the administrator (where the "@" is replaced by "."); then there are the serial number of the domain (to update at every change in the zone configuration so the secondary server can notice the change) and several timers that govern the transfer rate of zones and the life of the records.

The hostnames do not end with a "." (dot) are handled as relative name (not FQDN) and the zone name is added for suffix.

Rows starting with a blank space or a "@" are referred to the previous record: in this example three A type records are associated to *dns.example.org*.

The "IN" value in the second column specifies the class of the record (IN=internet).

The value "10", after the MX field, represents a priority value and it is used in that cases where more than one MX records are present.

Furthermore add for every hostname (so 'A' type RR) the following RRs types:

- HINFO: for information about CPU and OS (RFC 1035), f.i.
  ```
  IN      HINFO       vCPU Debian
  ```
- TXT: generic text field (RFC 1035), f.i.
  ```
  IN        TXT       descriptive_text
  ```

Restart the service with

```
# systemctl restart bind9.service
```

Verify the absence of errors

```
# systemctl status bind9.service
```

With *netstat* verify that the service is listening on the previously settled interfaces.

Updating of DHCP in R1

The configuration of DHCP server has to be updated to let clients using this service get the correct DNS configuration. On the DHCP server edit the file /etc/dhcp/dhcp.conf and specify in the global section (outside every *subnet* or *group* declaration) the following directives, use the right values:

```
option domain-name "domain.name.";
option domain-name-servers IP_SERVER_DNS;
```

Add in the global section the following option that permit to indicate a list of domains to "append" at the hostname if it is not specified with FQDN (f.i. in a command *ping*)

```
option domain-search "domain.name.";
```

Restart the service to apply the changes.

The host R1 is already configured to use S1 as DNS[13].

Part two

Tools for DNS

To query a DNS, in Unix-like environments "***dig***" or "***host***" commands are used, while in Windows there is "***nslookup***". This last command is available also for Unix-like environments, but it can have not consistent behaviour between various distributions, therefore its usage is discouraged.

[12]Per maggiori dettagli si consultino RFC 1034,1035 e successivi.

[13]The configuration is done with the file */etc/dhcp/dhclient.conf*

dig (domain information groper): brief overview

```
dig  [@server] [-b address] [-c class] [-f filename] [-k filename] [-m]
     [-p port#] [-q name] [-t type] [-x addr] [-y [hmac:]name:key] [-4] [-6]
     { [name] [type] [class] } [queryopt...]
```

The simplest form uses the following sintax:

```
dig @server name type
```

- **server** is the hostname or the server IP address to query
- **name** is the name of the record to look up (e.g. www.example.com)
- **type** is the type record of the query (e.g. A, CNAME, MX)

If "*@server*" is omitted, dig looks up the file /etc/resolv.conf to find a DNS server.
If "*type"* is omitted dig uses the type "A" by default.
For example to find the IP related to the hostname www.example.com

```
dig @IP.SERVER.DNS www.example.com A
```

By default answers are very verbose and are showed also information about the query and the authoritative server of the zone.
Common query options:

- **-q**: specifies the record name to search for.
- **-t**: specifies the record type. If *-q* is set to a domain name and *-t* is "AXFR" a transfer zone is requested.
- **-x**: specifies the IP address for the *reverse lookup*. In this case the options *-q* e *-t* are not requested. By default dig invokes a query using as hostname the inverse IP notation, adding *.in-addr.arpa* as suffix and setting *PTR* as type.

Repeating the previous parameters, it is possible doing queries on more records.
Query options are general options influencing the query requests and the answers visualization. Every option is preceded by a "+" and if necessary by "*no*" to disable the effect. If they are specified just after the *dig* command they have global validity for every query.

- **+[no]tcp:** [not]uses the TCP protocol when the query type is AXFR.
- **+[no]recurse:** [not]sets the recursive query (by default the query is recursive).
- **+[*no*]trace:** [not]sets the query as iterative. If it is enabled dig makes an iterative query starting from the root server (TLD) of the searched hostname and showing the answer from each server that was used to resolve the query (by deafault is disabled).
- **+[*no*]short:** [not]sets the verbose output that is the default value.

Others useful options that affect the output of the command can be the following:

- +[no]cmd
- +[no]comments
- +[no]question
- +[no]all
- +[no]answer

Examples

```
dig @8.8.8.8 www.google.it A   (it returns the IP of www.google.it)
dig +ncmd +nocomments +noquestion @8.8.8.8 www.google.it A   (like previous)
dig @8.8.4.4 google.it NS   (it returns authoritative DNS servers for the google.it domain)
dig @8.8.4.4 -x 8.8.8.8   (it does the revers search for the IP 8.8.8.8 quering DNS 8.8.4.4)
dig @8.8.8.8 google.it -t AXFR   (it requests a zone transfer)
```

Note: generally DNS servers allow zone transfer only to trusted host.

rnd: name server control utility

It controls the server operation, for example:

- cache deletion

```
rndc flush
```

- reload configuration file and zones after change

```
rndc reload
```

For all the showed commands see the relative man page for whole explanation.

DNS query

On H1 start Wireshark and do the following queries, select S1 as DNS server.

1. Verify that *h1.rd.corp.local* is a “A” record.

```
dig h1.rd.corp.local A
```

2. Display all records relative to IP *192.168.0.129* e *10.0.2.15*

```
dig -x 192.168.0.129 -x 10.0.2.15
```

3. Verify that *www.corp.local* is an alias and establish its relative A record

```
dig -q www.corp.local -t CNAME
dig -q www.corp.local -t A
```

4. Display all records relative to the host *r1.corp.local*.

```
dig -q r1.corp.local -t ANY
```

5. Display all records associated to *corp.local*.

```
dig -q corp.local -t ANY
```

6. Display all records of the zone *corp.local*

```
dig -q corp.local -t AXFR
```

   ◦ Which are the differences in the client and server communication over previous queries? Try explaining the motivation.

In the previous queries on single record it is used the UDP protocol, while for zone transfer (that usually take place between DNS servers for the zone configuration update) the TCP protocol is used to have a reliable data transfer.

7. Make a <u>non recursive</u> query for the *A* record related to *ipv6.polito.it.* Which is the answer? Why?

```
dig -q ipv6.polito.it -t A +norecurse
```

Due to the fact that S1 is not able to resolve *ipv6.polito.it* (it is not authoritative for the zone *polito.it*, furthermore it hasn’t the information in its cache) and <u>the query is not recursive</u>, BIND does not forward the request to its configured forwarder server, but it answers with a list of root servers. This root servers are the only one, for the configuration of S1, that are able to deal the request.

8. Repeat the query with the recursion. What is the result? Verify the behaviour of the DNS server using Wireshark on it to see the network traffic.

Si ottengono le informazioni richieste. Il server inoltra la richiesta al proprio forwarder.

The server returns the information requested. The server forwards the request to its forwarder server

9. Repeat the step 7. Is it possible to get the correct answer now? Why?

The answer now is correct as at step 8 because the DNS has the information in its cache.

Part three (Optional)

To solve the anomaly for the host H1 add a new DNS zone called *rd.corp.local* where register the host H1 with its real name "*H1*" and not "*H1.RD*". It is also desired that *rd.corp.local* represents a host with IP address *192.168.0.139*, reachable also as *www.rd.corp.local*.
Establish the necessary records for the new zone and reconfigure the system to allow both the direct search and the reverse search.

Tips.

- In the zone *corp.local* delete the RRs related to *h1.rd.corp.local*
- Update the file *named.conf.local* adding the new zone
- Create a new zone file *db.rd.corp.local* with the necessary RRs (SOA, NS, A, CNAME, HINFO, TXT)
- Update the zone file *0.168.192.in-addr.arpa* with the new PTR record

Modifica RR per zona ***corp.local***

Name	Type	Value
~~h1.rd.corp.local~~	~~A~~	~~192.168.0.1~~

RR per nuova zona ***rd.corp.local***

Name	Type	Value
rd.corp.local.	NS	s1.corp.local
	A	192.168.0.130
h1.rd.corp.local.	A	192.168.0.1
www.rd.corp.local.	CNAME	rd.corp.local

Nuovo RR per zona ***0.168.192.in-addr.arpa***

Name	Type	Value
130.0.168.192.in-addr.arpa.	PTR	rd.corp.local

Reload the BIND configuration

```
# rndc reload
```

1. Make a query of "A" type and "NS" type for *rd.corp.local* to verity the right translation.

```
dig -q rd.corp.local -t A
dig -q rd.corp.local -t NS
```

2. Make a query for all records associated to *rd.corp.local.*

```
dig -q rd.corp.local -t ANY
```

3. Make a transfer for the zone *rd.corp.local* to display all records in that zone.

```
dig -t AXFR rd.corp.local
```

4. Repeat step 2, specifying both as a DNS server than as value of the query *ipv6.polito.it*.

```
dig @ dns.ipv6.polito.it -q ipv6.polito.it -t ANY
```

   - What can be said about the searched record (number of authoritative servers, mail servers of the domain, administrator's e-mail address...)?

ipv6.polito.it represents both a domain name with 4 authoritative servers (SOA and NS records) than a hostname (A record). Moreover it represents the mail server for the domain (MX record). The administrator's mail is *risso@polito.it*

5. Ask to server *130.192.225.79* a zone transfer for zone *ipv6.polito.it.*

```
dig @dns.ipv6.polito.it -q ipv6.polito.it -t AXFR
```

- What's the difference between ANY and AXFR types query against a zone name?

The ANY type query displays the records associated at the requested name (both zone name than hostname) and uses UDP protocol. The AXFR type query requests a zone transfer allowing to display all the records of the zone and uses TCP protocol, if the name is not referred to a zone an error is showed.

- Depending from which network you connect (Ladispe or out of Politecnico) you can have different output: which can be the reason of this behaviour?

In general zone transfers are very onerous cause the amount of data that has to be transferred and they are allowed only to trusted servers and networks. In this case the DNS server allows transfer only to Ladispe networks and other servers in the Politecnico networks, so any request of zone transfer from untrusted hosts is denied.

EXERCISE 3

This exercise contemplates:

- the configuration on the same server web of two sites with different hostname using the field "*Host*" of the *HTTP/1.1* header (named based Virtual Hosting)
- sending and reading of mail using *telnet* with protocols *SMTP* and *POP3*
- breve analisi del traffico

Boot all nodes: R1 provides DHCP and routing, S1 provides web and mail server, H1 is a generic client, it is used to make the various required tasks.

Part one

The web server to configure is *Apache 2.4.25* on host S1.
The two sites match the hostnames configured in the exercise 2 that are *www.corp.local* and *intra.corp.local*, their files are already stored in two directories in /var/www/
The server web has to be configured to redirect the requests to the right web site.

Apache uses the directive *VirtualHost*[14] that allows, using the "*Host*" header of HTTP/1.1, to specify the site where forward the request.
In Debian the list of the web sites configured on the system and available to be on-line is in /etc/apache2/sites-available. Create in this folder two copies of the basic file 000-default.conf and call them, for example, www.corp.conf and intra.corp.conf . These files will be the configuration files for the sites *www.corp.local* and *intra.corp.local.*

Note: the file names must end with "*.conf*"

Edit the file created in previous step and specify the right value for the directives:

```
ServerName[15] www.example.com
DocumentRoot /var/www/html
```

For the site *intra.corp.local* disable *persistent HTTP* adding the directive:

```
KeepAlive Off
```

Now it is necessary "enable" the configured sites in order to be visible on-line, for both sites invoke:

```
a2ensite configuration_file_for _the_website
```

Finally reload the web server configuration with:

```
systemctl reload apache2
```

1. From H1 (or R1) use the browser to connect at the homepage of the two sites and capture the traffic.

List the most salient differences between the two captures.

In the answer packet sent by server, the content of the field *Host,* in the HTML header, changes.

The browser always requests a persistent connection using the header:

```
Connection: keep-alive
```

In the answer of *intra.corp.local* the non persistent HTTP is set by using the header:

```
Connection: close
```

In the connection to the site *intra.corp.local* a new TCP session is opened for every object that has an external reference and consequently a higher network traffic is generated. On the contrary in the connection to the site *www.corp.local* a single TCP session can be used to make more than one GET.

[14]Apache2 reference documentation https://httpd.apache.org/docs/2.4/vhosts/
[15]Apache2 reference documentation https://httpd.apache.org/docs/2.4/vhosts/examples.html

2. Download the homepage of both sites using *telnet* and see if the two captures are different than in the previous step. Specify the protocol commands needed.

```
root@h1:~# telnet www.corp.local 80
Trying 192.168.0.129...
Connected to s1.corp.local.
Escape character is '^]'.
GET / HTTP/1.1
host:www.corp.local
```

Notes:
http methods are *case sensitive* – https://tools.ietf.org/html/rfc2616#section-5.1;
after header command "*host:www.corp.local" press Enter* **twice**

Both captures are essentially the same, only one TCP session is opened and then the homepage is required with a GET. In both cases there are no multiple connection, but the browser analyzes the content of the page and then proceeds making new GET for any external objects present in the page.

Part two

Seeing the DSN configuration done in the exercise 2 the host S1 is indicated as the mail server for the domain *corp.local*: *Postfix*[16] is used as SMTP server , while *Dovecot*[17] is used as POP3/IMAP. In this configuration the mail system (on S1) uses the same users of the Linux operating system and deny sending messages to external users.

Execute Wireshark on R1 to see the generated traffic. From H1 connect with *telnet* to the mail server, using SMTP protocol send an email from *user1* to *user2* and then read, again with *telnet*, the messages using POP3 protocol.

1. Specify the listening port used by the SMTP server and the minimal set of commands to be used for sending a message.

Sending with SMTP

```
root@h1:~# telnet mail.corp.local 25
Trying 192.168.0.129...
Connected to mail.corp.local.
Escape character is '^]'.
220 mail.corp.local - Reti di Calcolatori - ESMTP Postfix
HELO h1.corp.local
250 mail.corp.local
MAIL FROM:user1@corp.local
250 2.1.0 Ok
RCPT TO:user2@corp.local
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
SUBJECT:Reti di Calcolatori
ESERCITAZIONE 3
Protocolli di posta: SMTP, POP3
.
250 2.0.0 Ok: queued as 190B4255ED
quit
221 2.0.0 Bye
Connection closed by foreign host.
root@h1:~#
```

Note: SMTP commands and options are case-insensitive - https://tools.ietf.org/html/rfc5321#section-2.4

[16]http://www.postfix.org/
[17]https://www.dovecot.org/

2. Specify the listening port used by the POP3 server and the minimal set of commands to be used for reading a message.

Reading with POP3.

```
root@h1:~# telnet pop3.corp.local 110
Trying 192.168.0.129...
Connected to s1.corp.local.
Escape character is '^]'.
+OK Dovecot ready.
USER user2
+OK
PASS networks
+OK Logged in.
stat
+OK 1 362
list
+OK 1 messages:
1 362
.
retr 1
+OK 362 octets
Return-Path: <user1@corp.local>
X-Original-To: user2@corp.local
Delivered-To: user2@corp.local
Received: from h1.corp.local (h1.rd.corp.local [192.168.0.1])
      by mail.corp.local (Postfix) with SMTP id B802A255ED
      for <user2@corp.local>; Mon, 17 Jul 2017 10:33:46 +0200 (CEST)
SUBJECT:Reti di Calcolatori
ESERCITAZIONE 3

Protocolli di posta: SMTP, POP3
.
dele 1
+OK Marked to be deleted.
quit
+OK Logging out, messages deleted.
Connection closed by foreign host.
root@h1:~#
```

Note: POP3 commands and options are case-insensitive - https://tools.ietf.org/html/rfc1939#section-3

3. See the traffic, what is the main problem about security of the examined system?

It is clearly seeable that the user2's password is sent in clear text on the network!

4. Try to send an e-mail message
   - to an external user not belonging to the *corp.local* domain;
   - to a not existent user of *corp.local* domain

Which answer codes are sent by server?

Mail to an external user:

454 4.7.1 <test@google.it>: Relay access denied

Mail to a non existent user:

550 5.1.1 <user3@corp.local>: Recipient address rejected: User unknown in local recipient table