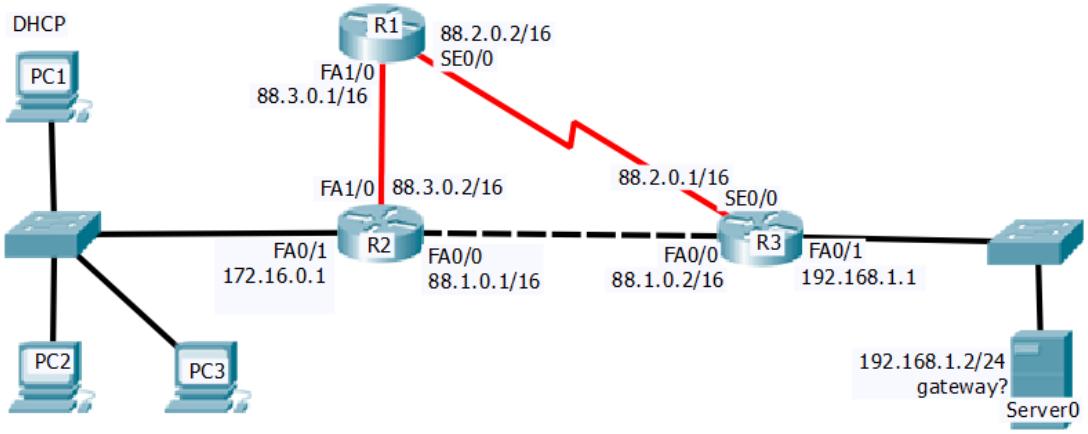
Laboratorio - 19 Febbraio 2019

Cognome	
Nome	
Matricola	

Packet Tracer

Si consideri la rete in figura



Q1) Configurare il protocollo **RIP versione 2** sul router **R2**, abilitare solo l'annuncio delle reti ***pubbliche*** ad esso connesse, e configurare come ***passiva*** l'interfaccia verso la rete privata ad esso connessa. *(2 punti)*
(NB: Scrivere in modo esplicito la modalità del router in cui deve essere eseguito ogni comando)

```
R2>enable
R2# configure terminal
R2(config)#router RIP
R2(config-router)#version 2
R2(config-router)#network 88.1.0.0
R2(config-router)#network 88.3.0.0
R2(config-router)#passive-interface fa0/1
```

Q2) Configurare NAT su **R2** per permettere l'accesso a Internet attraverso FA1/0 dalla rete privata. *(2 punti)*

```
R2>interface fa0/0
R2(config-if)# ip nat outside
R2(config-if)# exit
R2(config)# interface fa0/1
R2(config-if)# ip nat inside
R2(config-if)# exit
R2(config)# interface fa1/0
R2(config-if)# ip nat outside
R2(config-if)# exit
R2(config)# access-list 1  permit   172.16.0.0  0.0.255.255
R2(config)# ip nat inside source list 1 interface FastEthernet1/0 overload
```

Socket Programming

Si vuole scrivere un'applicazione client/server per emulare l'applicazione PING utilizzando socket UDP per calcolare il RTT e la percentuale di perdita di pacchetti nel collegamento fra client e server. Si ipotizzi che il server, una volta ricevuta la ECHO Request proveniente dal client, estragga un numero casuale (rand) e decida di rispondere al PING con una ECHO REPLY SOLO nel caso in cui il numero estratto sia maggiore di 9.

Q1) Completare lo script "UDP server". *(1 punto)*
Q2) Quanti PING sono scambiati prima di chiudere la connessione? **10** *(1 punto)*

UDP client

```
import socket
import time
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_addr = ('localhost', 12000)
sock.settimeout(1)
try:
    for i in range(1, 11):
        start = time.time()
        message = 'Ping #' + str(i) + " " + time.ctime(start)
        try:
            sent = sock.sendto(message, server_addr)
            print("Sent " + message)
            data, server = sock.recvfrom(4096)
            print("Received " + data)
            end = time.time();
            elapsed = end - start
            print("RTT: " + str(elapsed) + " seconds\n")
        except socket.timeout:
            print("#" + str(i) + " Requested Time out\n")
finally:
    print("closing socket")
    sock.close()
```

UDP server

```
import random
from socket import *
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', 12000))
while True:
    message, address = serverSocket.recvfrom(1024)
    message = message.upper()
    rand = random.randint(0, 20)
    if rand > 9
        serverSocket.sendto(message, address)
```