

Esame Completo - 11 Luglio 2017

Cognome	
Nome	
Matricola	

Tempo complessivo a disposizione per lo svolgimento: 2 ore 15 minuti

Si usi lo spazio bianco dopo ogni esercizio per la risoluzione

E1	E2	E3	Quesiti	Lab

1 - Esercizio (8 punti)

La rete di un ISP è riportata in figura. L'ISP possiede lo spazio di indirizzamento: 123.123.190.0/23. Definire un piano di indirizzamento in grado di supportare il numero di *host* indicato nella figura.

- a) Indicare le sottoreti IP graficamente nella figura, mettendo in evidenza i confini tra le reti IP ed assegnando una lettera identificativa a ciascuna rete. Assegnare le lettere in ordine alfabetico iniziando dalla rete più grande e procedendo per dimensione decrescente (# indirizzi rete A ≥ # indirizzi rete B ≥). Per ciascuna sottorete definire l'indirizzo di rete, la *netmask* (in formato decimale puntato), e l'indirizzo di broadcast diretto, usando la tabella 1. Assegnare gli indirizzi alle sottoreti a partire da quelli più bassi del blocco 123.123.190.0/23.
- b) Scrivere nella tabella 2 la tabella di instradamento del router R2 nel modo più compatto possibile dopo aver assegnato opportunamente degli indirizzi ai router a cui R2 è connesso.

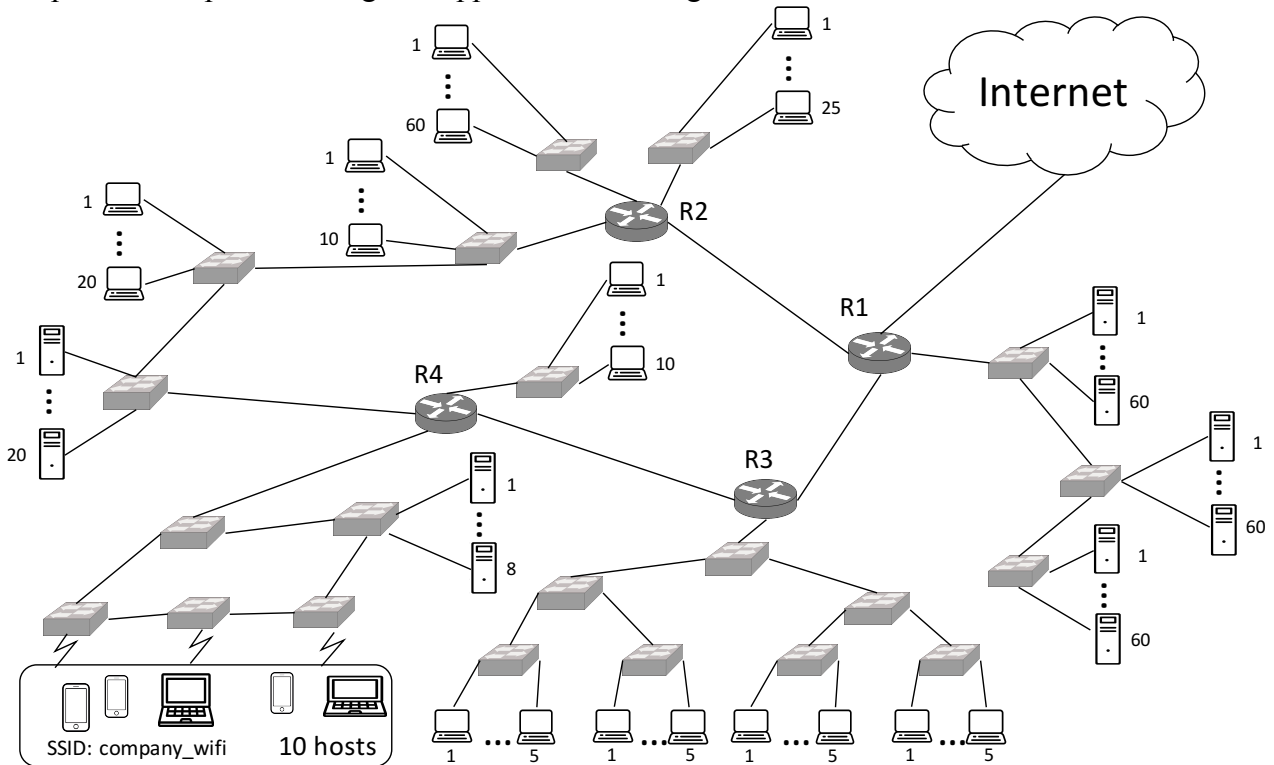
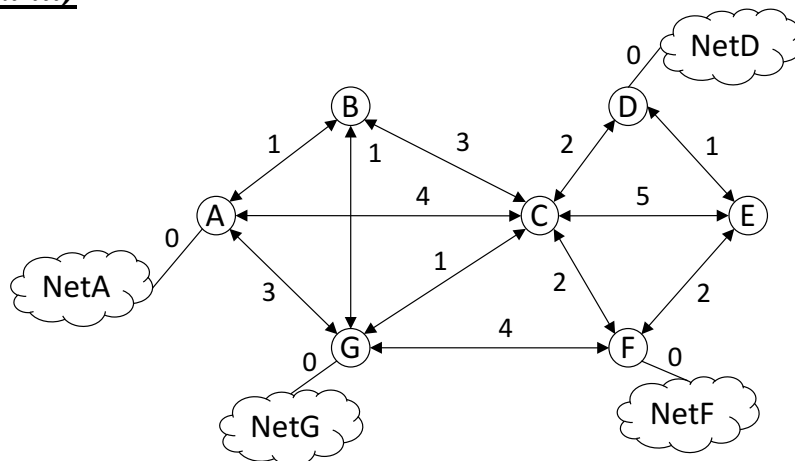


Tabella 1

Rete	Indirizzo di rete	Netmask	Ind. broadcast diretto
A			
B			

Tabella 2

Esercizio 2 (6 punti)



Nella rete in figura è rappresentato il grafo di una rete in cui sono presenti dei router (A, B, C, D, E, F, G) e 4 reti (NetA, NetD, NetF, NetG). I costi di attraversamento sono indicati accanto ad ogni link, i link sono bidirezionali e simmetrici. Si chiede di:

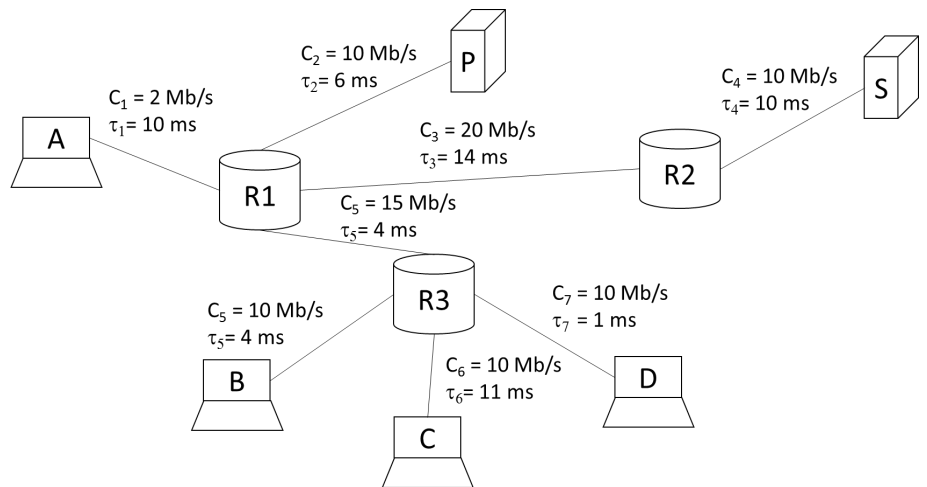
- a) Calcolare mediante l'algoritmo di Bellman-Ford l'albero dei cammini minimi con sorgente A e destinazioni tutti gli altri router (si omettano le reti nel grafo). Indicare:
 - nella Tabella A, il valore dell'etichetta ad ogni step in cui il nodo viene analizzato; nel caso lo step successivo non modifichi l'etichetta dello step precedente occorre riscrivere l'etichetta dello step precedente.
 - nella figura sopra, l'albero trovato
- b) Sulla base dell'albero dei cammini calcolato al punto precedente, indicare i Distance Vector (DV) relativi alle reti NetA, NetD, NetF e NetG, inviati dal router G nella modalità Split Horizon senza Poisonous Reverse. Per ogni DV inviato indicare chiaramente: il destinatario del DV, le reti raggiungibili comunicate ed i rispettivi costi.

Tabella A

Nodo A	Nodo B	Nodo C	Nodo D	Nodo E	Nodo F	Nodo G

Esercizio 3 (4 punti)

Si assuma B, P e S siano rispettivamente client, proxy e server HTTP. B richiede a P un contenuto che non è presente nella cache e che quindi P deve richiedere a S. Il contenuto consiste in un documento HTML base di 40 KByte e 9 immagini di 0.6 MByte ciascuna. Per tutto il tempo sono presenti e attivi i seguenti flussi TCP: A-S, C-S, D-S.



Assumiamo che:

- i messaggi di apertura connessione TCP e richieste HTTP sono di lunghezza trascurabile,
- si trascurano i tempi di accodamento nei nodi,
- la trasmissione dei file avviene a un ritmo medio di trasmissione pari al valore di condivisione equa delle risorse.

Si calcoli il tempo di trasferimento necessario, inteso come intervallo tra l'istante in cui B richiede il primo file a P e l'istante in cui B riceve interamente l'ultimo file da P, nelle due modalità:

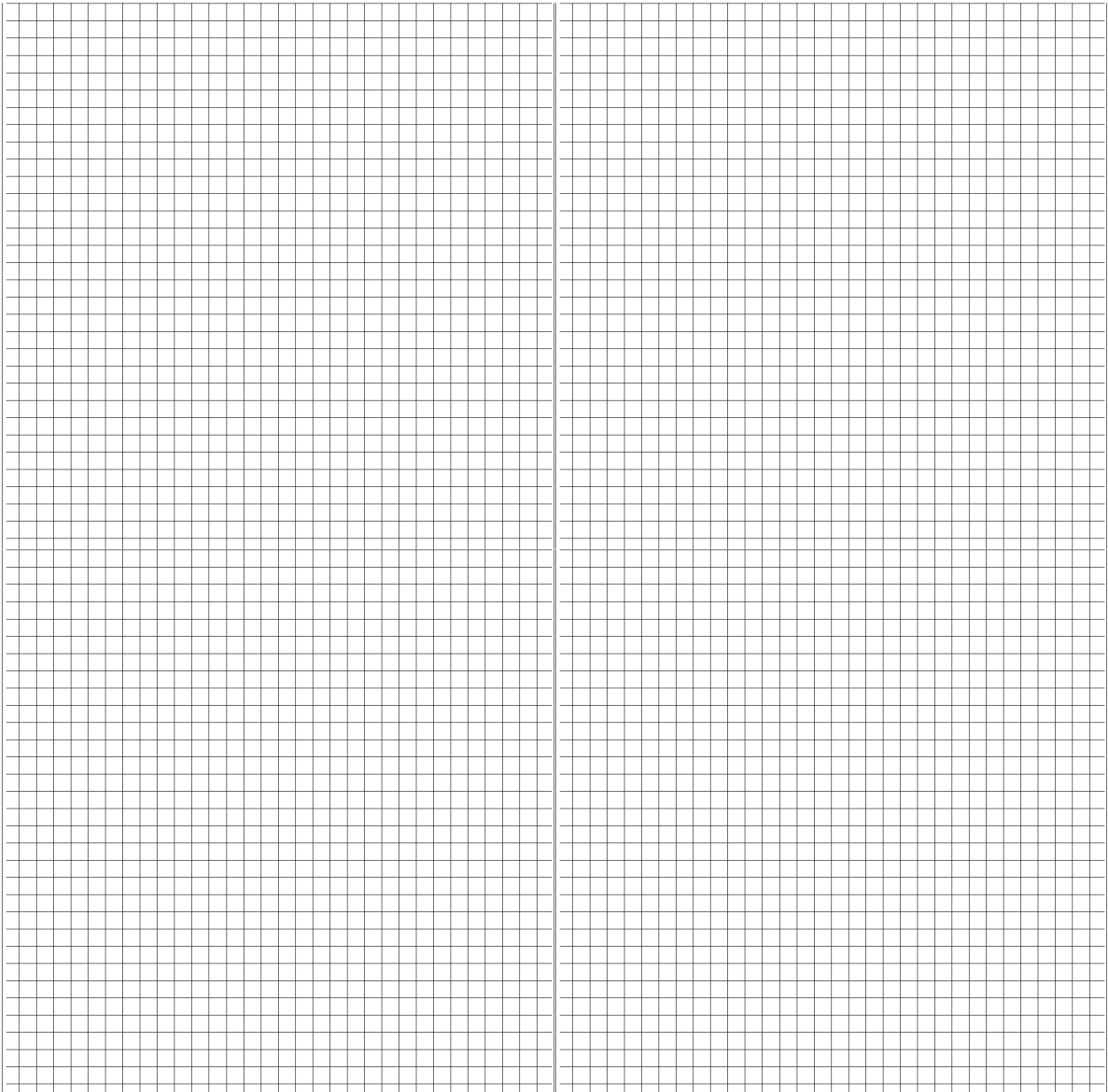
- connessione HTTP persistente (una singola connessione) e
- connessione HTTP non persistente (con trasmissione in parallelo delle immagini).

Si assuma che le connessioni HTTP tra S e P e tra P ed A avvengano con la stessa modalità.

		Flusso B-P	Flusso P-S	Unità mis.
Caso a	RTT			
	R_{HTML}			
	T_{HTML}			
	R_{OBJ}			
	T_{OBJ}			
Caso b	RTT			
	R_{HTML}			
	T_{HTML}			
	R_{OBJ}			
	T_{OBJ}			

$T_{Tot_a} =$

$T_{Tot_b} =$



4-Domande (9 punti)

DI - Un *router* è caratterizzato dalla seguente configurazione delle interfacce locali e della seguente tabella di *routing*. Per ciascuno dei pacchetti indicati di seguito (caratterizzati da interfaccia di provenienza, indirizzo di destinazione, dimensione e valore dei *flag* Do-not-Fragment) dire come si comporta il router specificando se procede con inoltro diretto, indiretto o se scarta il pacchetto (**tipo inoltro**). Indicare **chiaramente l'interfaccia di inoltro, la riga della tabella di *routing* "scelta" per l'inoltro indiretto ed eventualmente il motivo per cui il pacchetto viene scartato.**

eth0: 123.123.144.254, 255.255.240.0 MTU=500 [byte]

eth1: 212.111.128.254, 255.255.252.0 MTU=800 [byte]

<i>Riga #</i>	<i>Destinazione</i>	<i>Netmask</i>	<i>Next Hop</i>
1	131.175.32.0	255.255.224.0	212.111.128.222
2	131.175.64.0	255.255.192.0	123.123.150.254
3	0.0.0.0	0.0.0.0	212.111.130.254

212.111.131.23 da *eth1*, L=400 [byte], DF=1

Tipo inoltro: Interfaccia inoltro: Riga tabella (se necessario):
Eventuale motivo di scarto:

131.175.96.44 da *eth1*, L=600, DF=1

Tipo inoltro: Interfaccia inoltro: Riga tabella (se necessario):
Eventuale motivo di scarto:

131.175.192.34 da *eth1*, L=600 [byte], DF=0

Tipo inoltro: Interfaccia inoltro: Riga tabella (se necessario):
Eventuale motivo di scarto:

D2 - Un sistema di accesso multiplo a divisione di tempo (TDMA) è caratterizzato da slot di durata $T_s = 1$ [ms], con un rapporto $T_{\text{utile}}(\text{dati}) / T_{\text{guardia}} = 4$. Il sistema serve 20 stazioni e ciascuna ha una velocità di tributario pari a $v = 1$ [kb/s]. Indicare

- la durata temporale della trama, T_{TRAMA} ;
- il numero di bit di ciascuna stazione trasmessi in ogni slot, k ;
- il rate di trasmissione del segnale multiplato, C .

D3 - A proposito del parametro Maximum Segment Size (MSS) nel protocollo TCP [rispondere negli spazi previsti]:

- a) A che cosa serve?

b) Durante quale momento della connessione TCP viene configurato?

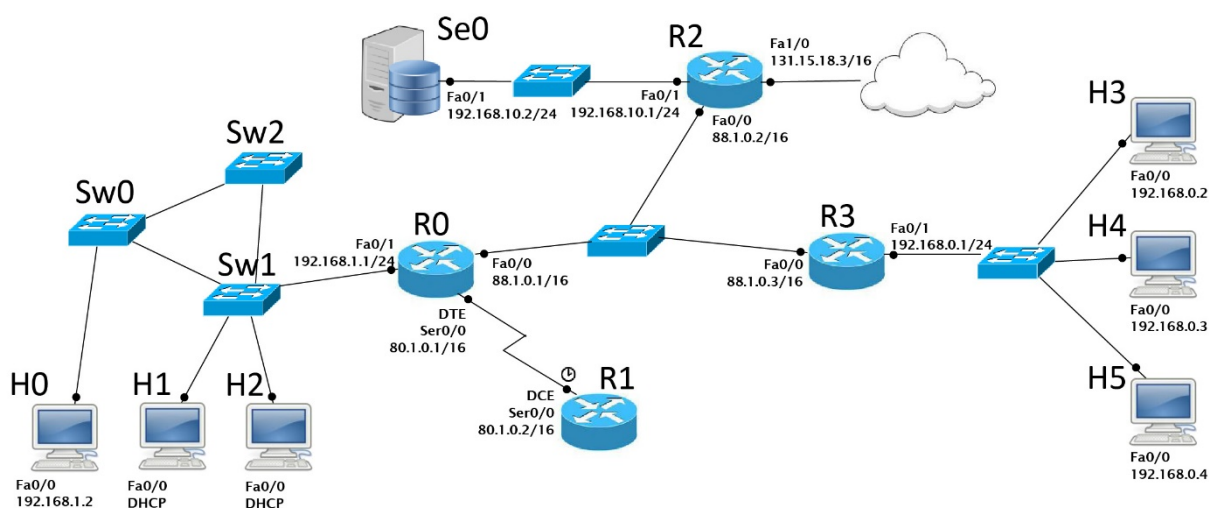
c) In quale campo del header del segmento TCP viene trasmesso?

d) In base a quale caratteristica della rete viene normalmente scelto il suo valore?

5 – Laboratorio (6 punti)

Packet Tracer

Si consideri la rete in figura



Attenzione:

- Indirizzi IP e gateway sono già stati configurati per i 6 host.
- Le interfacce dei router R0, R2 e R3 sono già state configurate ed attivate come in figura.
- Le reti /24 sono reti private
- Indicare sempre prima del comando il prompt visualizzato dal sistema, prestando attenzione alla modalità di partenza in ciascuna richiesta

Q1) Configurare ed attivare l'interfaccia seriale Ser0/0 del router **R1** assumendo un collegamento a 20 Mbit/s.

```
R1>
```

Q2) Configurare il routing statico sul router **R3** in modo che possa raggiungere tutte le reti pubbliche e internet, minimizzando il numero di regole necessarie.

```
R3 (config)#
```

Socket programming

Si vuole scrivere un'applicazione client/server UDP per conteggiare il numero di vocali presenti in una stringa.

Il client chiede all'utente di inserire una stringa, il server risponde indicando il numero di vocali presenti nella stringa (sia maiuscole che minuscole). **Hint:** `y.count(x)` conta quante volte appare l'elemento `x` nella lista `y`.

UDP client

```
from socket import *

serverName = 'localhost'
serverPort = 9999

-----

message = raw_input('Inserisci una frase:')
clientSocket.sendto(message, _____ )

try:
    reply, serverAddress = _____
    print reply
except error, v:
    print "Il server non ha risposto entro il timeout..."
finally:
    clientSocket.close()
```

UDP server

```
from socket import *

serverPort = 9999
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))

print 'Server pronto a ricevere!'

vocali = ['A', 'E', 'I', 'O', 'U']

while 1:
    messaggio, clientAddress = serverSocket.recvfrom(2048)

    num = 0
    for voc in vocali:
        num = num + messaggio.count(voc)
    risposta = "Il messaggio contiene " + str(num) + " vocali."
    serverSocket.sendto(risposta, clientAddress)
```

Q1) Completare lo script "UDP client" date le seguenti specifiche:

- Utilizzare indirizzi IPv4
- Time-out in ricezione: 5 secondi.
- Lunghezza buffer di ricezione: 2048 byte.

Q2) Con quale messaggio risponde il server se il client invia la stringa "aEniU" ?

Q3) Se si considera la seguente versione del server UDP (**in grassetto ci sono evidenziate le modifiche rispetto a UDP server1**), con quale messaggio risponde il server se il client invia la stringa "aEniU" ?

UDP server2

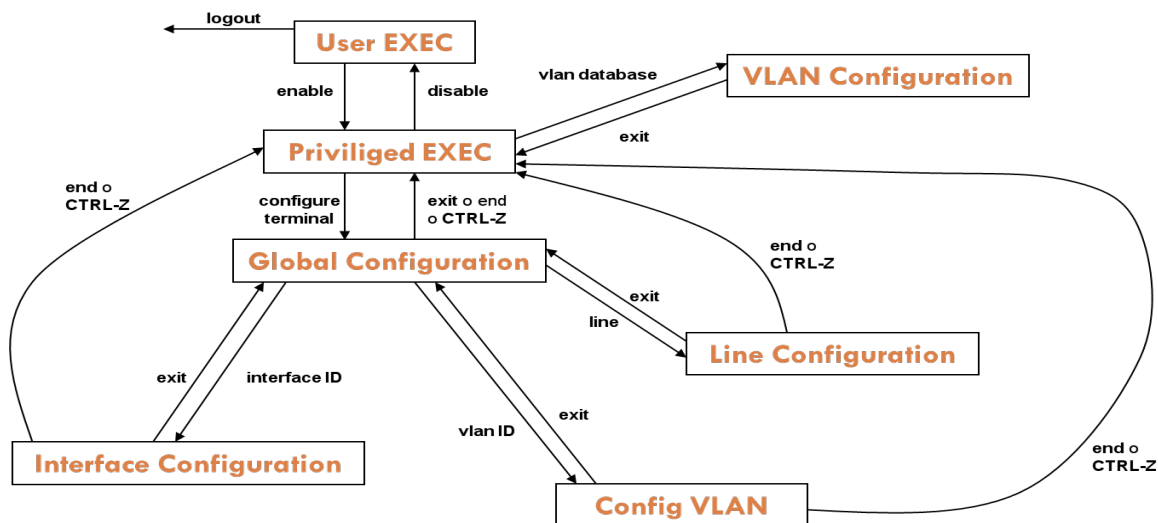
```
from socket import *

serverPort = 9999
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))

print 'Server pronto a ricevere!'

vocali = ['A', 'E', 'I', 'O', 'U']

while 1:
    messaggio, clientAddress = serverSocket.recvfrom(2048)
    messaggioUp = messaggio.upper()
    num = 0
    for voc in vocali:
        num = num + messaggioUp.count(voc)
    risposta = "Il messaggio contiene " + str(num) + " vocali."
    serverSocket.sendto(risposta, clientAddress)
```

Comandi

<pre>Router> Router> show cdp clock controllers frame-relay history interfaces ip version</pre>	<p>Modalità User EXEC</p> <ul style="list-style-type: none"> -CDP information -Display the system clock -Interface controllers status -Frame-Relay information -Display the session command history -Interface status and configuration -IP information -System hardware and software
<pre>Router> enable Router# Router# show access-lists arp cdp clock controllers frame-relay history interfaces ip running-config startup-config version</pre>	<p>Modalità Privileged EXEC</p> <ul style="list-style-type: none"> -List access lists -Arp table -CDP information -Display the system clock -Interface controllers status -Frame-Relay information -Display the session command history -Interface status and configuration -IP information -Current operating configuration -Contents of startup configuration -System hardware and software status
<pre>Router# copy running-config startup-config</pre>	<ul style="list-style-type: none"> -Salvare la configurazione corrente
<pre>Router# configure terminal Router(config)# Router(config)# hostname HOSTNAME Router(config)# banner motd Router(config)# enable secret PASSWORD Router(config)# no enable secret</pre>	<p>Modalità Global Configuration</p> <ul style="list-style-type: none"> -Cambiare nome al router -Impostare messaggio del giorno -Impostare password -Disabilitare password
<pre>Router(config)# interface TYPE SLOT/PORT Router(config-if)# no shutdown Router(config-if)# shutdown Router(config-if)# ip address IP_ADDRESS NETMASK Router(config-if)# clock rate CLOCK_RATE</pre>	<p>Configurare interfaccia</p> <ul style="list-style-type: none"> -Attivare interfaccia -Disattivare interfaccia -Assegnare IP -Clock seriale
<pre>Router(config)# line vty 0 4 Router(config-line)# password PASSWORD Router(config-line)# login Router(config-line)# ^Z</pre>	<p>-Accesso via rete (remoto).</p> <ul style="list-style-type: none"> -Impostare la password per l'accesso via rete
<pre>Router(config)# line console 0</pre>	<p>Accesso via porta console</p>
<pre>Router(config)# ip dhcp pool NAME POOL</pre>	<p>DHCP</p> <ul style="list-style-type: none"> -Nome pool indirizzi

Fondamenti di Internet e Reti

Proff. G. Maier, I. Filippini

<pre>Router(dhcp-config)# default-router ROUTER_IP_ADDRESS Router(dhcp-config)# network NETWORK_IP_ADDRESS NETMASK Router(dhcp-config)# ip dhcp excluded-address EXCLUDED_IP_ADDRESS</pre>	<ul style="list-style-type: none"> -Assegnare il default gateway al pool -Definire la rete a cui appartengono gli indirizzi -Escludere un indirizzo dal pool
<pre>Router(config)# ip route DEST_PREFIX DEST_NETMASK NEXTHOP/INTERFACE Router(config)# no ip route DEST_PREFIX DEST_NETMASK NEXTHOP/INTERFACE</pre>	<ul style="list-style-type: none"> -Aggiungere una rotta statica -Rimuovere una rotta statica
<pre>Router(config)# router rip Router(config)# no router rip Router(config-router)# version N Router(config-router)# network A.B.C.D Router(config-router)# passive-interface TYPE SLOT/PORT Router# debug ip rip Router# no debug ip rip Router# show ip route Router# show ip route rip Router# show ip protocols Router# show ip rip database</pre>	<ul style="list-style-type: none"> -Abilitare RIP -Disabilitare RIP -Scegliere la versione -Definire le reti che usano RIP -Configurare un'interfaccia in modalità passiva. -Abilitare/disabilitare il debug per il protocollo RIP - Ottenere la tabella di routing -Visualizzare le entry nella tabella di routing ottenute con RIP - Ottenere l'elenco dei protocolli di routing attivi e il loro stato - Visualizzare le informazione raccolte dal routing RIP
<pre>Router(config)# router ospf ID-PROCESS Router(config)# no router ospf ID-PROCESS Router(config-router)# network A.B.C.D NET_WILDCARD area N Router(config-router)# auto-cost reference- bandwidth BANDWIDTH_VALUE Router(config)# interface TYPE SLOT/PORT Router(config-if)# ip ospf cost COST VALUE</pre>	<ul style="list-style-type: none"> -Abilitare OSPF -Disabilitare OSPF -Definire le reti che usano OSPF -Modificare il valore di banda di riferimento -Modificare la metrica costo
<pre>Router(config)# router eigrp N Router(config)# no router eigrp N Router(config-router)# network A.B.C.D Router(config-router)# metric weights TOS K1 K2 K3 K4 K5</pre>	<ul style="list-style-type: none"> -Abilitare EIGRP -Disabilitare OSPF -Definire le reti che usano EIGRP -Modificare i pesi delle metriche
<pre>Router(config)# interface TYPE PORT/SLOT Router(config-if)# ip nat inside Router(config-if)# ip nat outside Router(config)# access-list LIST_NUM permit NET_ADDR NET_WILDCARD Router(config)# ip nat inside source list LIST_NUM interface OUTSIDE_INTERFACE_NAME overload</pre>	<p>Configurazione NAT</p> <ul style="list-style-type: none"> -definizione ruolo porte - Creare una lista di indirizzi a cui sarà permesso il NAT - Associare il NAT alla lista indicata prima
<pre>Router(config)# interface TYPE PORT/SLOT Router(config-if)# ip nat inside Router(config-if)# ip nat outside Router(config)# ip nat inside source static tcp IP_INSIDE PORT_INSIDE IP_OUTSIDE PORT_OUTSIDE</pre>	<p>Configurazione Port Forwarding</p> <ul style="list-style-type: none"> -definizione ruolo porte - Associare staticamente l'indirizzo e la porta esterna a quelli interni
<pre>Switch> enable Switch# show spanning-tree Switch> enable Switch# config Switch(config)# spanning-tree vlan 1 priority 0</pre>	<p>SPANNING TREE</p> <ul style="list-style-type: none"> -Controllare lo stato del protocollo STP -Impostazione di uno switch come Root Bridge

Codice esercizi laboratorio

UDP client

```
from socket import *
serverName = 'localhost'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
message = raw_input('Input lowercase sentence:')
clientSocket.sendto(message, (serverName, serverPort))
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
print modifiedMessage
clientSocket.close()
```

UDP server

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))
print "The server is ready to receive"
while 1:
    message, clientAddress = serverSocket.recvfrom(2048)
    print "Datagram from: ", clientAddress
    modifiedMessage = message.upper()
    serverSocket.sendto(modifiedMessage, clientAddress)
```

UDP error management

```
from socket import *
serverName = 'localhost'
serverPort = 12001
clientSocket = socket(AF_INET, SOCK_DGRAM)
clientSocket.settimeout(5)
message = raw_input('Input lowercase sentence:')
try:
    clientSocket.sendto(message, (serverName, serverPort))
    modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
    # in case of error blocks forever
    print modifiedMessage
except error, v:
    print "Failure"
    print v
finally:
    clientSocket.close()
```

TCP client

```
from socket import *
serverName = 'localhost'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = raw_input('Input lowercase sentence:')
clientSocket.send(sentence)
modifiedSentence = clientSocket.recv(1024)
print 'From Server:', modifiedSentence
clientSocket.close()
```

TCP server

```
from socket import *
serverPort = 12000
```

```
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind('', serverPort)
serverSocket.listen(1)
print 'The server is ready to receive'
while True:
    connectionSocket, clientAddress = serverSocket.accept()
    print "Connection form: ", clientAddress
    sentence = connectionSocket.recv(1024)
    capitalizedSentence = sentence.upper()
    connectionSocket.send(capitalizedSentence)
    connectionSocket.close()
```

TCP client persistent

```
from socket import *
serverName = 'localhost'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
while True:
    sentence = raw_input('Input lowercase sentence ( . to stop):')
    clientSocket.send(sentence)
    if sentence == '.':
        break
    modifiedSentence = clientSocket.recv(1024)
    print 'From Server:', modifiedSentence
clientSocket.close()
```

TCP server persistent

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind('', serverPort)
serverSocket.listen(1)
while True:
    print 'The server is ready to receive'
    connectionSocket, clientAddress = serverSocket.accept()
    print "Connection form: ", clientAddress
    while True:
        sentence = connectionSocket.recv(1024)
        if sentence == '.':
            break
        capitalizedSentence = sentence.upper()
        connectionSocket.send(capitalizedSentence)
    connectionSocket.close()
```

TCP auto client

```
from socket import *
import time
serverName = 'localhost'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
for a in range(100):
    clientSocket.send('A')
time.sleep(1)
clientSocket.send('.')
#clientSocket.recv(1024)
clientSocket.close()
```

TCP auto server

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind('', serverPort)
serverSocket.listen(1)
while True:
    print 'The server is ready to receive'
    connectionSocket, clientAddress = serverSocket.accept()
    print "Connection form: ", clientAddress
    while True:
        sentence = connectionSocket.recv(1024)
        if sentence == '.':
            break
        print len(sentence)
#         connectionSocket.send(capitalizedSentence)
    connectionSocket.close()
```

TCP server thread

```
from socket import *
import thread
def handler(connectionSocket):
    while True:
        sentence = connectionSocket.recv(1024)
        if sentence == '.':
            break
        capitalizedSentence = sentence.upper()
        connectionSocket.send(capitalizedSentence)
    connectionSocket.close()
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.setsockopt(SOL_SOCKET, SO_REUSEADDR, 1)
serverSocket.bind('', serverPort)
serverSocket.listen(1)
while True:
    print 'The server is ready to receive'
    newSocket, addr = serverSocket.accept()
    thread.start_new_thread(handler, (newSocket,))
```