Nume și prenume: _____ **Grupa:** _____

Punctaj: _____

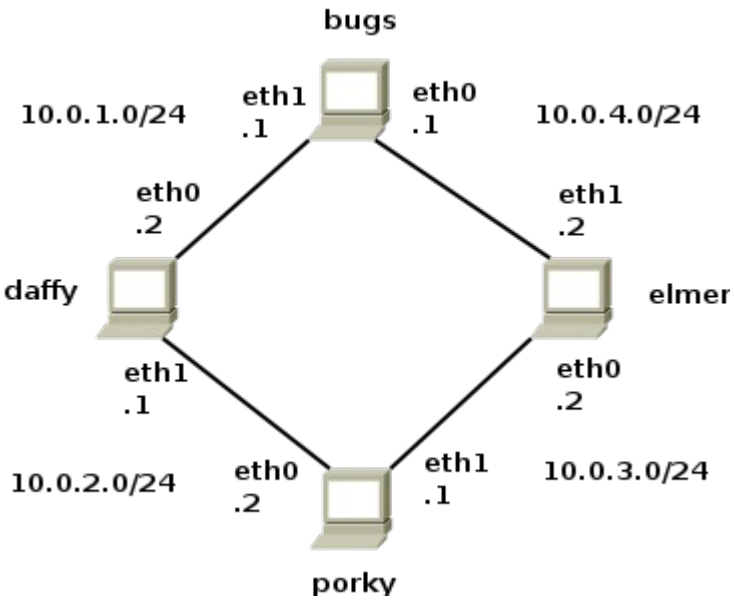
Indicații

- Testul conține **10** subiecte. Fiecare subiect este notat cu maxim **10p**. Punctajul total este de maxim **100p**.
- Se acordă punctaje parțiale doar în cazul subiectelor cu subpuncte.
- Puteți rezolva subiectele în orice ordine.
- Pentru a fi punctată, o rezolvare trebuie să includă şi **metoda de verificare** a funcționalității acesteia.
- Înainte de a începe testul, porniti mașinile virtuale, fiecare într-o consolă separată, folosind comenzile:

bugs:	**daffy:**	**porky**:	**elmer:**
`vzctl start 100`	`vzctl start 200`	`vzctl start 300`	`vzctl start 400`
`vzctl enter 100`	`vzctl enter 200`	`vzctl enter 300`	`vzctl enter 400`

- Pe toate mașinile, există:
  - utilizatorul **`root`**, cu parola **`student`**
  - utilizatorul **`student`**, cu parola **`student`**

Topologie



1. Adresare IP

(10p) Configurați în mod permanent adresele IP pe stațiile **bugs**, **daffy**, **porky** și **elmer**, conform figurii cu topologia:

bugs:	**daffy:**	**porky:**	**elmer:**
eth**0: 10.0.4.1**/24	eth**0: 10.0.1.2**/24	eth**0: 10.0.2.2**/24	eth**0: 10.0.3.2**/24
eth**1: 10.0.1.1**/24	eth**1: 10.0.2.1**/24	eth**1: 10.0.3.1**/24	eth**1: 10.0.4.2**/24

Rezolvare:

```
# vim /etc/network/interfaces

auto eth0
iface eth0 inet static
        address 10.0.4.1
        netmask 255.255.255.0
auto eth1
iface eth1 inet static
        address 10.0.1.1
        netmask 255.255.255.0

# /etc/init.d/networking restart
```

Asemanator pentru celelalte statii.

Verificare:

```
# cat /etc/network/interfaces
# ip address show
```

---

2. Rutare

(2p) **a)** Activați rutarea în mod permanent pe stațiile **bugs**, **daffy**, **porky** și **elmer**.

(8p) **b)** Asigurați conectivitatea în rețea, adăugând rute statice pe cele 4 stații de mai sus. Rutele vor fi configurate astfel:

pe **bugs:**	pe **daffy:**	pe **porky:**	pe **elmer:**
10.0.2.0/24 via **daffy**	**10.0.3.0/24** via **porky**	**10.0.4.0/24** via **elmer**	**10.0.1.0/24** via **bugs**
10.0.3.0/24 via **elmer**	**10.0.4.0/24** via **bugs**	**10.0.1.0/24** via **daffy**	**10.0.2.0/24** via **porky**

Rezolvare:

a)

```
# vim /etc/sysctl.conf
# net.ipv4.ip_forward=1
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

b)

```
# ip route add 10.0.2.0/24 via 10.0.1.2
```

(analog pentru restul rutelor)

Verificare:

a)

```
# cat /etc/sysctl.conf | grep ip_forward
# cat /proc/sys/net/ipv4/ip_forward
```

b)

```
# ip route list
```

Ping de pe o statie pe toate celelalte

---

3. Tunelare

(10p) Creați un tunel GRE între stațiile **daffy** și **elmer**, astfel încât:

pe **daffy**:	pe **elmer**:
• tunelul să plece de pe **eth0** • adresa IP a interfeței tunel să fie **192.168.100.1**/24	• tunelul să plece de pe **eth1** • adresa IP a interfeței tunel să fie **192.168.200.2**/24

Rezolvare:

daffy:

```
# ip tunnel add tun0 mode gre remote 10.0.1.2 local 10.0.4.2 dev eth1
# ip addr add 192.168.100.1/24 dev tun0
# ip link set tun0 up
```

elmer:

```
# ip tunnel add tun0 mode gre remote 10.0.4.2 local 10.0.1.2 dev eth0
# ip addr add 192.168.0.2/24 dev tun0
# ip link set tun0 up
```

Verificare:

```
# ip tunnel show
```

`# ping 192.168.0.1` sau `# ping 192.168.0.2`

4. DNS – rezolvare directă

(1p) **a)** Pe **bugs**, instalați serverul DNS **Bind9**.

(8p) **b)** Configurați serverul DNS pentru a răspunde la cereri pentru domeniul **cake.ro** cu următoarele informații:

- numele **cake.ro** are adresa IP **10.0.3.1**
- serverul de nume asociat domeniului este **ns.cake.ro**
- adresa IP a server-ului de nume este **10.0.2.1**
- serverul de mail asociat domeniului este **mail.cake.ro**, avand prioritatea **5**
- adresa IP a server-ului de mail este **10.0.3.2**
- numele **ftp.cake.ro** are adresa IP **10.0.4.2**
- numele **www.cake.ro** este un alias pentru **cake.ro**

Notă: Fișierul de zonă se va numi **db.cake.ro**.

(1p) **c)** Aplicati modificările făcute.

Hint: Folositi **named-checkzone** si **named-checkconf** pentru a depista eventualele erori.

Rezolvare:

a)

```
# apt-get install bind9
```

b)

```
# vim /etc/bind/name.conf.local
zone "cake.ro" {
        type master;
```

```
        file "/etc/bind/db.cake.ro";
};

# vim /etc/bind/db.cake.ro
$ORIGIN ro.
$TTL 36000
cake IN SOA ns.cake.ro. admin.cake.ro. (
                        2007092001
                        8H
                        2H
                        1W
                        1D
)
cake            IN NS ns.cake.ro.
cake            IN MX 5 mail.cake.ro.
cake            IN A 10.0.3.1
ns.cake.ro.     IN A 10.0.2.1
mail.cake.ro.   IN A 10.0.3.2
ftp.cake.ro.    IN A 10.0.4.2
www.cake.ro.    IN CNAME cake.ro.
```

c)

```
# /etc/init.d/bind9 restart
```

Verificare:

a)

```
# netstat -tlnp | grep 53
```

b) c)

(pe bugs)

```
# host -t NS cake.ro localhost
# host -t MX cake.ro localhost
# host -t A cake.ro localhost
# host -t CNAME cake.ro localhost
```

5. DNS – rezolvare inversă

(9p) **a)** Configurați serverul DNS pentru a răspunde și la cereri de rezolvare inversa pentru rețeaua 10.0.0.0, astfel:

- **`1.2.0.10.in-addr.arpa`** să răspundă cu **`ns.cake.ro`**
- **`2.3.0.10.in-addr.arpa`** să răspundă cu **`mail.cake.ro`**
- **`2.4.0.10.in-addr.arpa`** să răspunda cu **`ftp.cake.ro`**

Notă: Fișierul de zonă se va numi **`db.0.10.in-addr.arpa.`**

(1p) **b)** Aplicați modificările făcute.

Hint: Folositi **`named-checkzone`** si **`named-checkconf`** pentru a depista eventualele erori.

Rezolvare:

a)

```
# vim /etc/bind/name.conf.local
zone "0.10.in-addr.arpa"

{

        type master;

        file "/etc/bind/db.10.in-addr.arpa";
```

```
};
```

```
# vim /etc/bind/db.10.in-addr.arpa
$ORIGIN 0.10.in-addr.arpa.
$TTL 1D
@ IN SOA ns.cake.ro. admin.cake.ro. (
                2008091301 ; Serial
                8H ; Refresh
                2H ; Retry
                1W ; Expire
                1D ; TTL
)

NS ns.cake.ro.
1.2 IN PTR ns.cake.ro.
2.3 IN PTR mail.cake.ro.
2.4 IN PTR ftp.cake.ro.
```

b)

```
# /etc/init.d/bind9 restart
```

Verificare:

a) b)

```
# host -t PTR 10.0.2.1 localhost
# host -t PTR 10.0.3.2 localhost
# host -t PTR 10.0.4.2 localhost
```

---

6. Iptables

Scrieti o regula de firewall pe **bugs** care să:

(3p) **a)** accepte pachetele **echo-request** venite pe interfața **eth1**

(7p) **b)** limiteze numarul acestor pachete la 1 pe secundă, astfel incat sa se permita testarea conectivitatii prin **ping** dar sa se evite o forma de atac de tip DoS.

Hint: **--limit rate**

Rezolvare:

```
# iptables -t filter -I INPUT -i eth0 -p icmp --icmp-type echo-request -m limit --limit 1/s -j
ACCEPT
```

Verificare:

```
# iptables -t filter -L
```

---

7. SSH

(1p) **a)** Creați utilizatorul **duck** pe **daffy** și utilizatorul **fudd** pe **elmer**.

(1p) **b)** Instalați serverul SSH pe **daffy**.

(2p) **c)** Configurați serverul SSH să asculte conexiuni doar pe portul **2200**.

(6p) **d)** Configurați autentificare cu chei, astfel încât utilizatorul **fudd** de pe **elmer** să se poata autentifica pe contul **duck** de pe **daffy** folosind o cheie **DSA**.

Rezolvare:

a)

```
# adduser duck
# adduser fudd
```

b)
```
# apt-get install openssh-server
```

c)
```
# vim /etc/ssh/sshd_config
Port 2200
```

d)
```
fudd@elmer:$ ssh-keygen -t dsa
fudd@elmer:$ scp /.ssh/*.pub duck@10.0.2.1:
fudd@elmer:$ ssh duck@10.0.2.1
duck@daffy:$ mkdir .ssh
duck@daffy:$ touch .ssh/authorized_keys
duck@daffy:$ cat id_dsa.pub >> .ssh/authorized_keys
```

Verificare:

b) c)
```
# netstat -tlnp
```

d)
```
# fudd@elmer:$ ssh duck@10.0.2.1 -p 2200 -i /.ssh/id_dsa
```

---

8. Iptables

(8p) **a)** Pe **daffy**, adaugati o regula in **iptables** care sa blocheze toate pachetele destinate statiei pe portul **2200** cu exceptia pachetelor primite de la **localhost**.

(2p) **b)** Listati tabela de restrictii modificata si apoi stergeti-o in prezenta asistentului.

Rezolvare:

a)
```
# iptables -t filter -I INPUT -p tcp --dport 2200 -j DROP
# iptables -t filter -I INPUT -p tcp --dport 2200 -s 127.0.0.1 -d 127.0.0.1 -j ACCEPT
```

b)
```
# iptables -t filter -L
# iptables -t filter -F
```

Verificare:
```
# telnet localhost 2200
# telnet 10.0.2.1 2200
```

---

9. Captura trafic

(9p) **a)** Pe **bugs**, folositi **tcpdump** pentru a captura primele 3 pachete destinate masinii fizice, pe portul **22**.

(1p) **b)** Generati trafic pentru a verifica functionarea capturii.

Rezolvare:

a)
```
# tcpdump -n -vvv dst 10.38.x.y and port 22 -c 3
```

b)
```
# ssh 10.38.x.y
```

Verificare:
Output-ul comenzii tcpdump

10. Port forwarding

(10p) Pe statia **porky**, configurati **Port Forwarding** (PAT) astfel încât pachetele ce sosesc pe portul **953**, pe interfața **eth1**, să fie redirectate pe portul **53**, pe **daffy**, pe interfata **eth1**.

Rezolvare:

```
# iptables -t nat -A PREROUTING -p tcp -i eth1 --dport 953 -j DNAT --to 10.0.2.1:53
```

Verificare:

```
# iptables -t nat -L
```