Tema 2

* **Publicare**:

- ***2024-11-30 13:00***

* **Termen de predare**:

- ***2024-12-20 23:55 - deadline HARD!***

Revizii:

- ***2024-12-06 09:05*****:** Lansat t2check `v2024.6`: fix la task 8 + unele erori erau ascunse :(;
- ***2024-12-05 22:30*****:** Lansat t2check `v2024.5`: fix la task 1 (adresă pe Milano la anumite variabile) și punctajul la task 2;
- ***2024-12-05 12:00*****:** VM-ul local poate fi acum descărcat (VMware + VirtualBox);
- ***2024-12-04 21:00*****:** Lansat t2check `v2024.4`: rl-watchdog + dezactivat test rutare la IPv6;
- ***2024-12-02 16:30*****:** Lansat t2check `v2024.3` cu erori mai detaliate, mai ales la task-urile 6-8;
- ***2024-12-01 22:50*****:** Lansat t2check `v2024.2`, acum merge dat `t2check TASK_NR` pentru verificare individuală a task-urilor;
- ***2024-11-30 14:00*****:** Câteva clarificări prin enunț (la infra + checker)!
- ***2024-11-30 13:00*****:** Tema a fost lansată! *God help us*!

Tema constă în realizarea configurației unui set de exerciții pe o topologie (vedeți mai jos) simulată folosind containere (implementare în ContainerNet + Docker) într-o mașină virtuală (ori în cloud - OpenStack, ori descărcată și rulată local).

Notare

Fiecare exercițiu are un punctaj propriu. Nota pe întreaga temă este dată de suma punctajelor acumulate în urma rezolvării fiecărui exercițiu.

Punctajul maxim care se poate obține pe întreaga temă este 100 de puncte (tot ce este peste se trunchiază strict, per temă). Acest punctaj este echivalent cu **2 puncte** din nota finală.

Există **și** exerciții bonus, cu ajutorul cărora puteți obține un total de 125 de puncte (remember: se trunchiază, dar pot fi folosite pentru a completa punctaj parțial la alte task-uri ne-esențiale – vedeți mai jos).

Nu este obligatorie rezolvarea tuturor exercițiilor. Exercițiile pot fi rezolvate în orice ordine, mai puțin în situația în care un exercițiu depinde de rezolvarea unui alt exercițiu (de obicei, primele 4 de stabilire a conectivității containere – Internet).

Mașina virtuală

- Puteți rezolva tema ori folosind infrastructura OpenStack (aveți proiect special numit `rl_tema_prj` - meniu stânga sus, lângă logo OpenStack - unde găsiți și imaginea), ori o mașină virtuală locală (citiți mai jos).
- Notă: pe OpenStack, autentificarea cu username și parolă nu este posibilă! Folosiți autentificarea cu chei publice, prezentată aici (ATENȚIE: cheile SSH utilizate la laborator rămân disponibile pe TOATE proiectele, deci aveți setupul deja făcut!).
- Pentru autentificare pe VM local, utilizați credențialele **student/student**.

Pași pentru accesare OpenStack

- Urmăriți pașii de aici.
- Link dashboard: https://cloud.grid.pub.ro/ [https://cloud.grid.pub.ro/]
- Schimbați proiectul din dropdown-ul de stânga-sus al OpenStack, alegeți `rl_tema_prj` !!
- Ar trebui să aveți cheia publică deja importată pe `rl_tema_prj` (verificați!);

- Nume imagine (de selectat la *Sources*): **RL Tema2 v2024.0**;
- Tip instanță: **m1.small** (NU aveți nevoie de mai mult, iar celelalte vor fi șterse fără notificare);
- Puneți la VM nume cu prefixul `TEMA2_` (OBLIGATORIU! Altfel riscați să vă fie ștearsă) apoi username-ul vostru de Moodle (dacă veți avea nevoie de ajutor din partea unui asistent, să vă găsim ușor);
- **OBLIGATORIU:** autentificarea cu user și parolă a fost dezactivată, folosiți EXCLUSIV chei publice ssh (pe care ar trebui s-o aveți deja configurată în cadrul laboratoarelor).
- **NU modificați parola la conturile root / student!** Dacă vă tăiați accesul la VM din greșeală, cereți ajutorul unui asistentului preferat pe Teams ;)

Atenție! NU distrugeți instanța mașinii virtuale până nu ați încărcat arhiva finală pe Moodle și ați obținut punctajul dorit (și să fiți siguri că nu veți mai avea nevoie să modificați nimic).

Rulare în VM local

- Imaginea locală este acum disponibilă!
- Pentru a rula mașina virtuală a temei local, **o puteți descărca de la acest URL [https://repository.grid.pub.ro/cs/rl/RL_Tema2_2024_LocalVM_cdc15e2d.7z]** (*7GB dezarhivat*).
  - Va trebui să vă autentificați cu username + parola contului de la universitate!
- VM-ul este compatibil atât cu VirtualBox (testat cu 7.1), cât și VMWare (Workstation >= 17). Pe Linux, poate fi rulat și prin qemu+kvm.
- În arhivă sunt incluse ambele proiecte ce se pot deschide cu aplicația hipervizor (`.vmx` pentru VMware + `.vbox`).
- Accesul prin ssh cu parolă (`student:student`) este activat, deoarece VM-ul rulează pe o rețea privată.
- **Atenție: Imaginea VM-ului diferă de cea a laboratorului**, asigurați-vă că îl folosiți pe cel corect (ar trebui să aveți scripturile cu `t2*`)!

Infrastructură temă

- Înainte de a rula checkerul, se recomandă descărcarea ultimului update prin rularea:

```
root@host$ t2update
```

Personalizarea temei

Rezolvările sunt particularizate pentru fiecare student (pe baza contului Moodle). Pentru aceasta, rulați pe mașina temei comanda `t2start USERNAME_MOODLE` cu numele utilizatorului de Moodle ca argument.

După rulare, datele de particularizare vor fi regăsite în fișierul **`/root/assignment.txt` de pe host**. **Nu modificați aceste valori**, ele fiind verificate cu strictețe de către checker-ul de pe Moodle!

Ca și exemplu de pornire:

```
# asigurați-vă că aveți actualizat checkerul / infrastructura:
t2update
# inițializați-vă VM-ul:
t2start alex.cutarescu1337
# vedeți-vă variabilele:
cat /root/assignment.txt
```

Informațiile generate anterior vor fi folosite în enunțul temei cu următoarele notații:

- `$A` - valoarea variabilei A
- `$B` - valoarea variabilei B
- `$C` - valoarea variabilei C
- ...

Datele acestea sunt generate determinist din utilizatorul vostru. Dacă doriți să porniți cu mașina virtuală de la zero, rulând scriptul anterior vă va furniza exact aceleași valori ale variabilelor (cu excepția cazului în care ați greșit

numele!).

Verificare (checker)

- Pentru verificarea temei este disponibil un checker local. Atenție: nu este substitut pentru depanare!
  - Altfel spus, contează soluția voastră, nu checker-ul.
  - Dacă, dintr-o greșeală, checker-ul dă rezulat pozitiv în cazul unui exercițiu rezolvat greșit nu înseamnă că se va puncta. O eventuală actualizare a checker-ul va puncta corect (însă veți fi notificați să re-verificați pe canalele obișnuite, forum și Teams).
  - Obiectivul trebuie să fie rezolvarea corectă a enunțului. Checker-ul vine ca o confirmare (ne dorim cât mai sigură) a acelei rezolvări.
  - Punctajul afișat în arhiva salvată și urcată pe Moodle va fi și cel acordat în catalog la final (dacă acesta dat de ultima versiune a checkerului local și DOAR dacă nu s-a constatat că se trișează).
- Play fair: orice tentativă de fraudare / atac la infrastructură va avea ca efect pierderea definitivă a punctajului (sau chiar repetarea materiei, în anumite cazuri), chiar dacă checkerul este *păcălit* :P.
- Comenzile de mai jos pot fi rulate indiferent de directorul in care ne aflăm.
- Pentru a actualiza checker-ul + alte elemente de infrastructură:

```
t2update
```

- Pentru a rula checkerul:

```
t2check
```

- Exemple de folosire:

```
root@host:~# t2check
task01       ......................  10.0/10.0
task02       ......................  10.0/10.0
task03       ......................  10.0/10.0
task04       ......................  10.0/10.0
...

root@host:~# t2check 10
task10       ......................  0.0/10.0
   mail not received
```

- Dacă sunt probleme, puteți să postați un mesaj pe thread-ul aferent de pe forum [https://curs.upb.ro/2024/mod/forum/view.php?f=8654];

Predarea (upload-ul) soluției

1. Pentru împachetarea soluției, rulați:

```
root@host$ t2check --save
```

(atenție: această comandă va reseta rețelistica și reporni toate serviciile, echivalent cu un reboot mai rapid – asigurați-vă că ați lucrat persistent (*vedeți mai jos, la task-uri, cum*)!)

2. Rezultatul este o arhivă semnată în directorul în care invocați comanda (`$PWD`).
3. Folosind utilitarul `scp`, copiați acest fișier pe stația voastră locală (atenție să nu încurcați directoarele și să copiați o arhivă veche / salvată în altă parte!). Dacă sunteți conectat la OpenStack prin serverul intermediar `fep.grid.pub.ro`, va trebui să copiați mai întâi acolo, apoi s-o preluați pe stația voastră de lucru

(alternativ, puteți folosi funcționalitatea de `JumpHost` (opțiunea `-J`) a clientului `scp` pentru o conexiune directă).

4. Încărcați arhiva pe Assignment Tema 2 [https://curs.upb.ro/2024/mod/assign/view.php?id=75143] (Moodle).

Este **obligatoriu** ca rezolvarea exercițiilor să se facă în mod persistent. La o repornire a mașinii virtuale, rezolvările trebuie să rămână active, altfel puteți întâmpina dificultăți la o revenire ulterioară asupra temei.

Folosiți comanda `reboot` înainte să testați de-a întregul (checkerul local doar simulează ceva mai rapid).

De asemenea, puteți folosi comanda `systemctl restart rl-topology` pentru a reporni rapid toate containerele (sistemul lor de fișiere este și el persistent).

Pentru a rezolva tema este suficient să prelucrați fișiere doar din căile `/etc`, `/home` și `/root` (atât pe host, cât și pe containere). NU instalați alte pachete în plus!

NU opriți / ștergeți manual containerele Docker (decât dacă doriți să luați de la zero cu configurarea acestora, vedeți mai jos comanda).

Pentru a **șterge și reseta** configurația de pe un container, se folosiți secvența:

```
systemctl stop rl-topology && docker container rm <nume container> && systemctl restart rl-topology;
```

Configurații persistente

Toate configurațiile să fie **persistente**. Trebuie să fie active și după repornirea mașinii virtuale.

Pentru modificarea unor fișiere speciale din containerele Docker (`/etc/hosts`, `/etc/resolv.conf`) va trebui să folosiți un script de init sau hook de ifupdown-ng. Găsiți mențiuni adecvate la exercițiile unde va fi necesar să faceți asta!

Pentru configurație persistentă, atât host-ul, cât și containerele vin cu ifupdown-ng [https://github.com/ifupdown-ng/ifupdown-ng] pre-instalat.

Pentru a începe să configurați, verificați fișierele existente la căile `/etc/network/interfaces*` (mai ales `/etc/network/interfaces.d/rl.conf`) ⇒ RTFM here ;) [https://manpages.debian.org/bookworm/ifupdown-ng/interfaces.5.en.html].

Pentru a configura persistent **rute adiționale** (și nu numai!), puteți folosi hook-ul `up` din sintaxa de configurare `interfaces`. Exemplu (fragment):

```
iface <intf>
    address A.B.C.D/xx
    up ip route add X.Y.Z.T/yy via Q.W.E.R
```

Fișierele `/etc/network/interfaces*` se parsează și execută linie cu linie, scripturile `if[up|down]` oprindu-se la prima eroare întâlnită. De exemplu, dacă aveți într-o secțiune `iface` un hook "`up ip ro add invalid route...`" și pe rândurile următoare aveți alte declarații, acestea nu vor mai ajunge să fie aplicate!

Puteți folosi următorul oneliner pentru a verifica rapid o interfață:

```
ifdown --force <intf>; ifup <intf>; ip a sh; ip ro sh
# hint: folosiți ifdown și ifup cu parametrul -a pentru a porni TOATE interfețele declarate!
```

NU RULAȚI `ifdown -a` pe host! VĂ VEȚI PIERDE CONECTIVITATEA PE `eth0` (deci la mașina virtuală, dacă sunteți pe OpenStack) !!!

Pentru a salva/restaura regulile `iptables`, urmați pașii de aici: https://devops.stackexchange.com/questions/11991/how-to-save-and-restore-the-iptables-rule-and-configuration-from-file [https://devops.stackexchange.com/questions/11991/how-to-save-and-restore-the-iptables-rule-and-configuration-from-file] (pentru `iptables-restore`, există mai multe modalități, e.g. puteți pune hook-uri de `up` la o interfață etc.). SUB NICI O FORMĂ SĂ NU INSTALAȚI PACHETELE DESCRISE DIN TUTORIALE (mai ales `ifupdown` – vă strică VM-ul, aveți deja `ifupdown-ng`!!).

Dacă folosiți mai multe fișiere în scripturi (e.g., apelați dintr-un script alt script), folosiți căi absolute. Adică folosiți `/root/scripts/make-juju.py` în loc de `./make-juju.py` pentru a nu se baza pe directorul actual de lucru (`working directory`). NU uitați să le faceți executabile și să includeți shebang-ul!

Discuții legate de temă

Toate discuțiile legate de probleme/întrebări/exerciții din tema de RL trebuie puse pe forumul temei [https://curs.upb.ro/2024/mod/forum/view.php?f=8654]. Reguli de utilizare ale acestui forum:

- Nu se pun rezolvări directe pe forum.
- Fiecare întrebare legată de un task trebuie pusă pe thread-ul dedicat acelui task.
- În momentul în care puneți o întrebare, includeți în mesaj:
  - contextul în care a apărut problema pe care o semnalați
  - ce ați încercat să faceți pentru a repara problema
  - alte informații care pot descrie mai bine problema
  - dacă este vorba de rezolvarea unui task, cum ați verificat rezolvarea task-ului
- Verificați dacă cineva nu a mai întâmpinat aceeași problemă și i-a fost oferit un hint sau o soluție (lurk before you leap).
- Post-uri care nu sunt puse pe thread-ul corespunzător vor fi șterse.
- **Nu creați thread-uri noi (off-topic) de discuție! Vor fi șterse.**

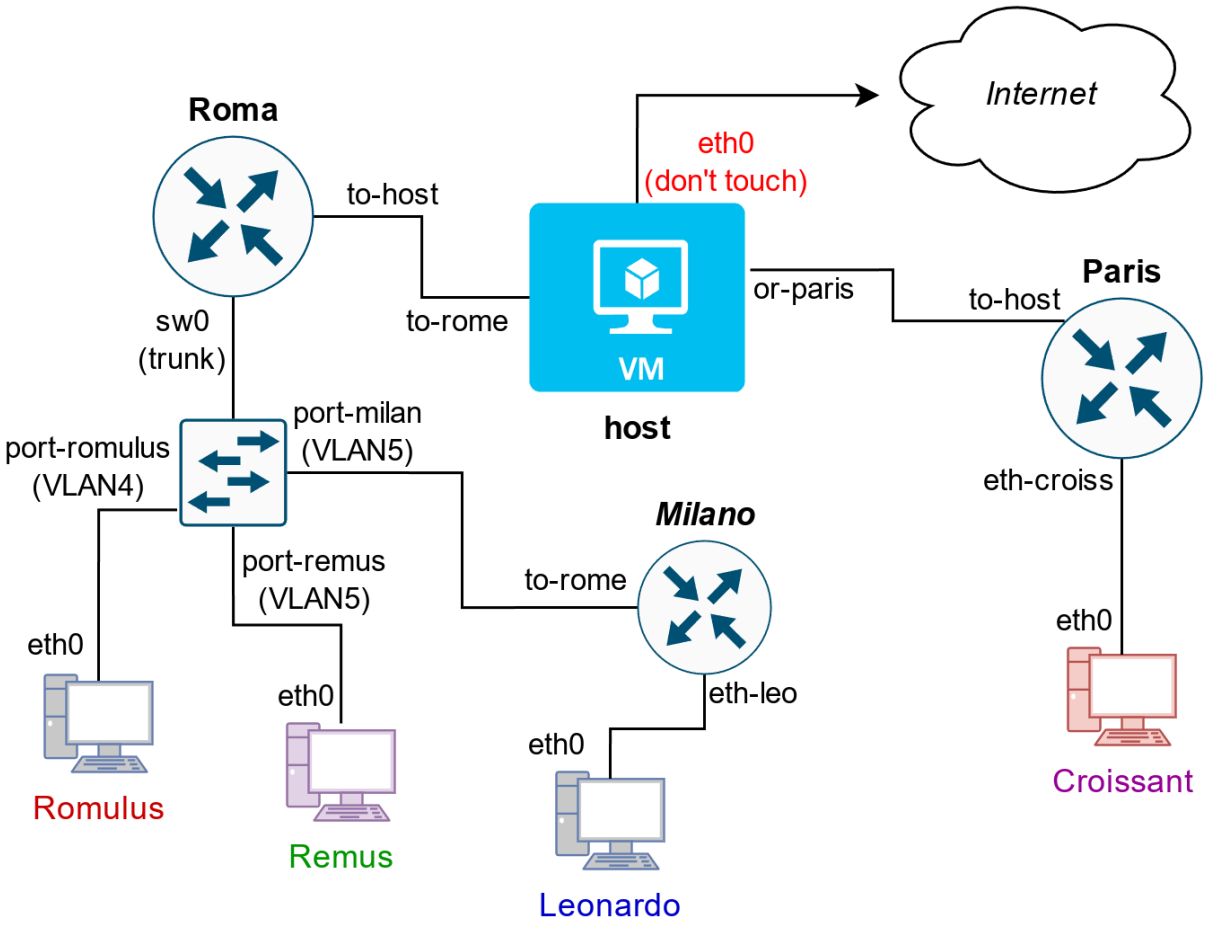
Subiecte

Este recomandat să citiți enunțul în întregime prima oară și de oricâte ori aveți întrebări!. Și, desigur, recitirea completă a task-ului următor de care vă apucați / după o pauză îndelungată de la rezolvarea acestuia.

Topologie & infrastructură

În topologie aveți ruterele `host` (fiind VM-ul care găzduiește toată infrastructura), `Roma`, `Paris` și `Milano`; stațiile `Romulus` și `Remus` sunt conectate la `Roma` printr-un bridge virtual pe router (în VLAN-urile `VLAN4` respectiv `VLAN5` – porturile sunt deja configurate); adițional, ruterul `Milano` este conectat tot la `VLAN5` (deci partajează rețeaua cu `Remus`); `Leonardo` este la `Milano`, iar `Croissant` la `Paris`.

Pentru a accesa un echipament, folosiți comanda de la laborator: `go NUME_ECHIPAMENT` (atenție: numele este case-sensitive!). Dați `docker ps` pentru a vedea denumirea containerelor (și eliminați prefixul `mn.` pentru a obține denumirea folosibilă prin `go`).



Va trebui să realizați **primele 4** exerciții în ordine. Întrucât aceste exerciții oferă, în final, conectivitate la Internet, restul vor depinde de acestea!

Ex. 1 [25p] Adresare + rutare IPv4

Recomandăm citirea primelor 4 cerințe în întregime înainte de a vă apuca de lucru efectiv! De asemenea, v-ar fi [extrem de] util să re-citiți TOATE instrucțiunile de mai sus!

1. Subnetați FIX (i.e., dimensiuni egale, maximizare nr. de stații) spațiul `10.$A.$B.0/24` și configurați cu adrese IPv4 toate legăturile din topologie în ordinea cerută (începând cu PRIMA adresă asignabilă), astfel:
   - prima subrețea alocată va fi `VLAN4`, asignare în ordinea: `Roma`, `Romulus`;
   - a doua subrețea alocată va fi `VLAN5`, asignare în ordinea: `Roma`, `Milano`, `Remus`;
   - a treia subrețea alocată va fi cea dintre `Milano` și `Leonardo` (asignare în această ordine);
   - a patra subrețea alocată va fi cea dintre `Paris` și `Croissant` (la fel, în această ordine);
2. Subnetați OPTIM spațiul `172.30.$C.240/28` + configurați echipamentele (`host` va avea mereu prima adresă asignabilă) astfel:
   - o rețea între `host` și `Roma`;
   - cealaltă (ultima rămasă): `host` și `Paris`.
3. Configurați rutarea IPv4 (default GWs și/sau rute statice) astfel încât toate stațiile să se poată accesa unele pe altele prin adresă IP!

Denumirea interfețelor pe Linux este similară ca în toplogia de mai sus, cu precizarea că sub-interfețele de pe router ce trebuiesc configurate au forma `<intf>.<vlan_id>` (cele cu `port-<X>` de pe switch sunt doar Layer 2 și pot fi ignorate). NU stricați VLAN ID-urile / redenumiți interfețele bridge-ului!

Atenție mare: adresa IP de pe `eth0` a sistemului `host` este asignată dinamic de către hipervizor, prin DHCP; NU vă atingeți (*never go full ifdown!*) de această interfață, altfel riscați să vă pierdeți accesul la VM!

Ex. 2 [15p] Adresare IPv6

1. Configurați adrese IPv6 pentru rețeaua `VLAN4` și `VLAN5` (notă: variabila `$VLANID` va avea valoarea 4, respectiv 5, cu zero-uri în față până la completarea segmentului de 16 biți):
    - Folosiți spațiul `2024:baba:$B:$A:$VLANID::/96`.
    - Aceeași ordine de asignare ca la IPv4.
2. Configurați conectivitate IPv6 între `Roma` și `host`:
    - Folosiți spațiul `fdee:dada:$C:$D::/64`.
    - Prima adresă asignabilă este pentru `host`, a doua a lui `Roma`.
3. Configurați rutarea IPv6 pentru a permite comunicarea între toate sistemele cu adresă IPv6.
4. *Atenție*: echipamentele `Leonardo`, `Paris` și `Croissant` **NU** vor avea adresă IPv6!

Ex. 3 [5p] Hosts

- Realizați configurațiile necesare astfel încât echipamentele să poată fi accesate prin numele lor (folosiți numele `host`, `Roma`, `Milano`, `Paris`, `Romulus`, `Remus`, `Leonardo` și `Croissant` - atenție la MAJUSCULE!). Adăugați intrări doar pentru adresele IPv4.

Fișierul `/etc/hosts` din containere este mai special (montat ca volum `--bind` de către Docker). Acest lucru face ca orice modificare a acestuia să se piardă la fiecare restart al containerului (aka reboot al VM-ului sau restart al serviciului `rl-topology`). Dacă (și sigur) vreți să persiste, puteți să-l salvați în altă cale (e.g., `/etc/hosts.orig` și să-l restaurați mereu când pornește containerul printr-un hook de `up` în `/etc/network/interfaces.d/rl.conf`). Și nu folosiți `cp -f` (acesta vrea să șteargă fișierul și nu puteți face asta cu un volum montat), folosiți `cat` + redirectare simplă în bash (va trunchia & suprascrie corect), e.g.:

```
iface <intf>
        up cat /etc/hosts.orig >/etc/hosts
```

Ex. 4 [5p] Internet connectivity

- Realizați configurațiile necesare pentru ca cele 7 echipamente sa aibă acces la Internet:
    - Configurați translatarea pe sistemul host astfel încât containerele să poată primi răspunsuri din Internet.
    - Configurați containerele pentru a putea accesa resurse din Internet pe baza numelor de domeniu al acestora (DNS).
    - **Atenție**: nu translatați orice pachet este rutat (e.g., cele între containere ar trebui să-și vadă IP-urile originale)!

Fișierul `resolv.conf` este gestionat ca volum de către Docker :(… aceeași poveste ca la exercițiul anterior ⇒ aceeași soluție (e.g., faceți un fișier `/etc/resolv.conf.orig` pe care îl veți suprascrie peste `/etc/resolv.conf` cu un hook pe `up cat …` din fișierele interfaces).

Ex. 5 [10p] Network Address Translation

- Configurați reguli de **DNAT** pe sistemele `Roma` și/sau `host` (după caz), astfel:
    - Conexiunile pe `Roma` la porturile `(24000 + $E)`, `(24000 + $F)` și `(24000 + $G)` să conducă la conectarea ssh pe sistemele `Romulus`, `Remus` respectiv `Leonardo`.
    - Conectarea pe `host` la portul `(9000 + $K)` să conducă la conectarea pe tracker-ul de pe sistemul `Milano`.
- **Sfat**: aveți grijă cum testați: DNAT-ul va funcționa DOAR dacă veniți dintr-o rețea externă ruterului (e.g., `host` sau `Paris` vs `Roma`)!

Ex. 6 [10p] Filtrare pachete (iptables)

- Configurați **filtrarea** de pachete pe `Roma` / `Paris` / `Milano` (după caz) astfel încât:

- conexiunile SMTP și Telnet inițiate de pe sistemul `Remus` în afara rețelei lui să fie blocate *(inclusiv către alte rutere!)*;
- conexiunile către tracker-ul ce rulează pe sistemul `Milano` **să nu** fie permise de la `Croissant`.
- blocați TOATE conexiunile externe (i.e., de pe IP-urile din afara stației) către `Leonardo`, mai puțin protocoalele `icmp` și `ssh`.
- **atenție**: NU blocați conexiunile inițiate de `Leonardo` **și nici răspunsurile** de la acestea! folosiți reguli **stateful** (i.e. connection tracking)!

Ex. 7 [10p] Chei SSH

- Configurați serviciul de **SSH** pe sistemele `Romulus`, `Remus`, `Leonardo` și `host` astfel încât autentificarea cu utilizatorul `student` de pe oricare sistem să fie permisă pe toate celelate sisteme (tot în cadrul utilizatorului `student`) folosind chei publice;
- Folosind alias-uri SSH [https://collectiveidea.com/blog/archives/2011/02/04/how-to-ssh-aliases/], configurați SSH-ul pe `host` pentru a vă putea conecta rapid folosind următoarele comenzi:
  - `ssh romu` să ducă către sistemul `Romulus` cu utilizatorul `student`;
  - `ssh remu` să ducă către sistemul `Remus` cu utilizatorul `student`;
  - `ssh leo` să ducă către sistemul `Leonardo` cu utilizatorul `student`;

Atenție: NU VĂ ȘTERGEȚI CHEIA AUTORIZATĂ DE LA OPENSTACK PE HOST! În mod normal, nu aveți nevoie să operați pe acest fișier la temă!

Ex. 8 [10p] Support ticketing

- Creați, pe sistemul `Romulus`, scriptul `/home/student/scan-support-tickets` care să scanane și preia de pe serverul de IMAP al `Milano` toate mesajele ce conțin `Support Ticket` în subiect și să le trimită înapoi un reply (de pe același server, către mailul sender-ului!) de forma:

```
Hi,

Your ticket named '<SUBJECT>' was registered as #<X>.

Thank you for your patience!
```

- Unde în locul lui `<SUBJECT>` să fie subiectul inițial, iar la `<X>` va fi interpolat un cod numeric unic (orice strategie se acceptă, e.g., nr. crescător începând cu `#1`).
  - Mesajul poate fi în orice formă, atâta timp cât conține informațiile cerute!
- Scriptul poate fi realizat în orice limbaj de scripting e instalat pe stație, însă va trebui să fie executabil direct (e.g., prin shebang). De asemenea, scriptul va fi rulat din checker cu working directory necunoscut (deci nu vă bazați pe el!).
- Va trebui să vă autentificați pe serverul de mail (atât IMAP cât și SMTP) de pe `Milano` cu credențialele:
  - *username*: `support`;
  - *password*: `Pierdut$Cont1337` (caracterul dolar face parte din parolă, nu denotă o variabilă);

Ex. 9 [10p] Serviciu de sincronizare automată

- Pe `Remus`, realizați un script la `/home/student/scripts/auto-backup` care să sincronizeze automat fișiere locale către remote, cu următorii parametri:
  - recursiv, din directorul `/home/student/Documents/` (de pe `Remus`) către serverul+calea `Roma:/var/remus-backup/`!
  - a se păstra structura directoarelor, începând cu rădăcina (e.g., `~/Documents/ceva/test` să ajungă la `/var/remus-backup/ceva/test`);

    - ignorați (i.e., nu copiați!) fișierele ascunse (cele cu `.`) la sursă sau cele care conțin `VIRUS` în denumire;
    - NU aveți voie să modificați ownerul/permisiunile directorului destinație (verificați-le înainte pentru a vedea cum trebuie să ajungă);
    - NU aveți parola de autentificare a utilizatorului țintă (defapt, nici nu are), dar tot trebuie să faceți ceva în această privință (i.e., să vă autentificați cu succes!)…
- Folosi un utilitar eficient! un fișier nou apărut ar trebui să se sincronizeze în max. 3 secunde!
- Scriptul va trebui să stea pornit până la primirea SIGINT/SIGKILL. La testare manuală, să îl porniți / opriți manual…
    - *Comportament checker*: dacă checkerul vede că procesul `auto-backup` este deja pornit, nu va mai face lucrul acesta, însă NU va acorda punctajul (însă va face sincronizarea și verificarea, lăsându-vă să depanați scriptul vostru cu stdout/stderr);
    - Dacă folosiți un infinite loop, ar trebui să vă faceți scriptul să iasă la primirea SIGTERM / SIGINT și să nu rămână agățat!
- Convenția de apel a scriptului (de către checker) este similară cu cea descrisă în exercițiul anterior.

Ex. 10 [Bonus - 10p] Wireguard tunnel

- Dorim să conectăm rețelele `Milano-Leonardo` și `Paris-Croissant` împreună printr-un tunel Wireguard:
    - Pentru rețeaua din tunel, folosiți spațiul `10.$H.$J.96/30`; prima adresă asignabilă este a lui Milano, iar ce-a de-a doua a lui `Paris`;
    - Va trebui să puteți accesa adresele IP ale capetelor de tunel și de pe `Leonardo` și `Croissant`!
    - Denumiți interfețele de tunel `wg-rl` la ambele capete.
    - Tunelul wireguard trebuie să fie persistent! (folosiți hook-uri în `interfaces`).

Ex. 11 [Bonus - 15p] Pin your hair

- Pe stația `Milano` va rula un serviciu securizat pe portul TCP `1000+$K` care va asculta DOAR pe adresa wireguard creată la task-ul anterior (checkerul îl va porni automat; pentru testare puteți folosi `nc` cu argumentul `-l` și IP-ul WireGuard (+ desigur, portul), trebuie să puteți trimite mesaje bidirecționale prin portul forwardat).
- Configurați DNAT pe `Paris` astfel încât să poată primi conexiuni pe portul `1000+$K` și să le forwardeze prin tunelul Wireguard către `Milano`, același port.
    - Accesul la serviciu prin `Paris:1000+$K` va trebui să fie funcțional din orice rețea!
- **Restricție:** este obligatoriu să folosiți DOAR iptables și/sau rutare (care ar trebui să fie deja configurată la ex. anterior) pentru a rezolva acest exercițiu, e.g., nu e voie să folosiți un serviciu auxiliar care să asculte pe portul `1000+$K` și să redirecționeze pachetele la `Milano`!
- *Hint: folosiți `tcpdump` cu încredere când nu funcționează ceva ;)*