МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
Фізико-технічний інститут

ОСНОВИ КОМП'ЮТЕРНИХ МЕРЕЖ

методичні вказівки до виконання комп'ютерного практикуму

версія тестової лабораторії: v0.3

Київ – 2021

1 Тестова лабораторія

1.1 Архітектура тестової лабораторії

Робота базується на симуляції мережної архітектури на основі ізольованих середовищ у вигляді docker контейнерів. Варто зазначити, що така симуляція має багато обмежень порівняно з фізичною архітектурою. Наприклад, за необхідності створення мережного інтерфейсу відбувається підняття мосту між контейнером і хостом, причому хост стає шлюзом за замовчуванням (для повноцінної маршрутизації, ця проблема вирішується перепризначенням шлюзу за замовчуванням під час розгортання контейнеру).

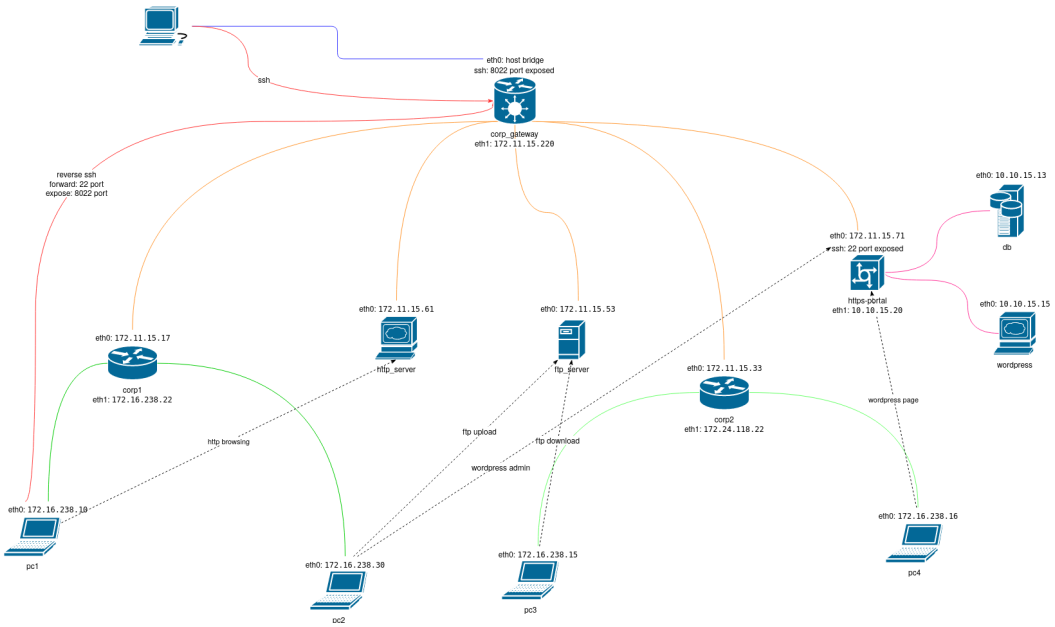
Проте така архітектура доволі адекватно симулює більшість мережних технологій, в тому числі протоколи мережного рівня та вище, технологію трансляції адрес NAT та протоколи канального рівня (з певними обмеженнями).

1.1.1 Топологія мережі

Тестова лабораторія складається з вузлів різного призначення:

- маршрутизатори локальних підмереж (`corp1` та `corp2`): призначені для логічної сегментації мережі на дві внутрішні підмережі з групами робочих станцій `pc1,pc2` та `pc3,pc4`.
- WAN шлюз `corp_gateway`, який є виходом корпоративної мережі на WAN.
- Внутрішньо корпоративні сервіси: http сервер, ftp сервер.
- Внутрішній https зворотній проксі для веб сервісу Wordpress (з окремим сервером бази даних). Цей сервіс також має внутрішній SSH доступ.
- Робочі станції `pc1-pc4`, які періодично (автоматично) симулюють роботу з корпоративними сервісами. При цьому, на станції `pc1` відкритий прокидання портів (port forwarding) на шлюз `corp_gateway`, на якому піднятий зворотній SSH, що дає змогу отримати SSH доступ до робочої станції ззовні корпоративної мережі.

Топологія мережі:



Варто зазначити, що завдяки вбудованому DNS в Docker контейнерах, до вузлів всередині однієї підмережі можна отримати як за IP (`172.16.238.10, 172.16.238.30`) адресою так і за хостнеймом (`client_pc1, client_pc2`)

1.1.2 Розгортання лабораторії

Лабораторія базується на технології ізольованих контейнерів Docker з засобами керування розгортанням Docker-compose, під управлінням Лінукс дистрибутиву Xubuntu. Практично весь процес розгортання та налаштування середовища автоматизований.

Для розгортання лабораторії необхідно:

1. Розгорнути віртуальну машину (Virtualbox, Vmware) з образом Лінукс дистрибутиву Xubuntu 20.04 (на інших дистрибутивах Ubuntu теж, ймовірно, можна, але не тестувалося. На інших лінукс дистрибутивах, ймовірно, можна, але можуть бути нюанси).
2. Інсталювати `git`:

```
$ sudo apt update
$ sudo apt install -y git
```

3. Клонувати репозиторій зі скриптами розгортання та конфігурацією архітектури:

```
$ git clone https://github.com/animant/ftilabs_comnetworks.git
```

4. Налаштувати середовище:

```
$ cd ftilabs_comnetworks
$ ./setup_env.sh
```

5. Розгорнути мережну архітектуру:

```
$ ./start_lab.sh
```

   Після закінчення розгортання, в терміналі виводиться версія лабораторії – її необхідно вказувати у звіті, щоб точно розуміти, на якій версії інфраструктури виконується робота.
6. Почекати 1-2 хвилини для ініціалізації сервісів.
7. Отримати доступ до робочої станції `pc1` можна через зворотний SSH:

```
$ ssh -i ./assets/pc1_access root@127.0.0.1 -p 8022
```

8. Для зупинки (згортання) лабораторії є відповідний скрипт:

```
$ ./stop_lab.sh
```

2 Лабораторна робота №1

2.1 Мета роботи

Метою даної лабораторної роботи є отриманя розуміння роботи базових мережних протоколів канального та мережного рівня (Ethernet, ARP, ICMP, IP), а також отримання навичок роботи з розповсюджении утилітами для аналізу і конфігурування у комп'ютерних мережах: `ifconfig, ip, netstat, ping, tcpdump, wireshark`.

2.2 Теоретичний матеріал

2.2.1 Збір інформації по архітектурі

1. Спершу, необхідно розгорнути лабораторію та отримати SSH доступ до робочої станції `pc1`:

```
$ ssh -i ./assets/pc1_access root@127.0.0.1 -p 8022
```

2. Отримайте інформацію про налаштування інтерфейсів (ip адресу, мак адресу, broadcast адресу) за допомогою команди

```
$ ifconfig -a
```

3. Канальний рівень передачі даних відбувається між сусідніми вузлами, тобто вузлами, що знаходяться в одній локальній підмережі. Оскільки ідентифікаторами сторін в цьому випадку є MAC адреси, то адреси сусідів мають зберігатися на кожному вузлі. Перевірити збережені адреси для сусідніх вузлів можна за допомогою команди `arp`:

```
$ arp -e
```

   Отримуємо мак адресу шлюзу за замовченням 172.16.238.22 (corp1):

```
root@f2ec0f8bbc3b:~# arp -e
Address                  HWtype  HWaddress           Flags Mask            Iface
corp1.ftilabs_comnetwor  ether   02:42:ac:10:ee:16   C                     eth0
root@f2ec0f8bbc3b:~#
```

Якщо спробувати знайти маршрут до іншого вузлу цієї підмережі, то після відпрацювання протоколу ARP `client_pc1` отримає та збереже в arp кеші відповідний мак:

```
$ ping -c 4 172.16.238.30
$ arp -e
```

```
root@f2ec0f8bbc3b:~# ping client_pc2
PING client_pc2 (172.16.238.30): 56 data bytes
64 bytes from 172.16.238.30: icmp_seq=0 ttl=64 time=0.179 ms
64 bytes from 172.16.238.30: icmp_seq=1 ttl=64 time=0.123 ms
64 bytes from 172.16.238.30: icmp_seq=2 ttl=64 time=0.124 ms
^C--- client_pc2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.123/0.142/0.179/0.026 ms
root@f2ec0f8bbc3b:~# arp -e
Address                  HWtype  HWaddress           Flags Mask            Iface
corp1.ftilabs_comnetwor  ether   02:42:ac:10:ee:16   C                     eth0
client_pc2.ftilabs_comn  ether   36:3b:86:f7:9d:5e   C                     eth0
root@f2ec0f8bbc3b:~#
```

Інший спосіб дізнатися мак адреси сусідніх вузлів – через команду `ip`:

```
$ ip neigh show
```

```
root@f2ec0f8bbc3b:~# ip neigh show
172.16.238.22 dev eth0 lladdr 02:42:ac:10:ee:16 REACHABLE
172.16.238.30 dev eth0 lladdr 36:3b:86:f7:9d:5e STALE
root@f2ec0f8bbc3b:~#
```

4. Перейдіть по SSH (пароль SSH з'єднання – `P@ssw0rd`) на https portal та очистіть ARP кеш:

```
$ ssh root@172.11.15.71
$ ip -s neigh flush all
```

5. Шлюз за замовчуванням – це машина в тій самій локальній підмережі через яку даний хост буде встановлювати IP шлях до машин з інших підмереж. Дізнатися про свій шлюз за замовчуванням можна за допомогою команди `route -n`:

```
$ route -n
```

```
root@f2ec0f8bbc3b:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.16.238.22   0.0.0.0         UG    0      0        0 eth0
172.16.238.0    0.0.0.0         255.255.255.0   U     0      0        0 eth0
root@f2ec0f8bbc3b:~#
```

Окрім шлюзу, в таблиці маршрутизації відображаються поточні підмережі для всіх мережних інтерфейсів.
Примітка: під час розгортання docker контейнеру, шлюзом за замовчуванням є фізичний хост. Для коректного перенаправлення трафіку, при розгортанні інфраструктури лабораторії, видаляється встановлений шлюз, а сусідній роутер призначається новим шлюзом за замовчуванням:

```
$ route add default gw 172.16.238.22
$ route del default gw 172.16.238.1
```

2.2.2 Аналіз трафіку через tcpdump і wireshark

`Wireshark` – це розповсюджена утиліта для аналізу мережного трафіку зі зручним GUI і доступна для різних платформ. Проте в реальних ситуаціях, часто ми маємо лише віддалений доступ через термінал чи SSH, без підтримки графічної оболонки та з обмеженими ресурсами. У таких випадках є сенс використовувати консольні утиліти `tcpdump` (уже встановлений на робочі станції лабораторії) та `tshark` (консольна версія `wireshark`).

1. Для початку перехоплення трафіку, на цільовій машині використовуйте команди:

```
$ tcpdump -i eth0 # прослуховувати трафік на інтерфейсі eth0
$ tcpdump -i eth0 port 80 # прослуховувати http трафік (порт 80)
$ tcpdump -i eth0 dst 10.10.13.15 # фільтрувати трафік за адресою одержувача
$ tcpdump -i eth0 -w /tmp/out.cap # зберегти трафік в /tmp/out.cap
```

Примітка: більше інформації про утиліту можна знайти, наприклад, тут: `https://danielmiessler.com/study/tcpdump/`

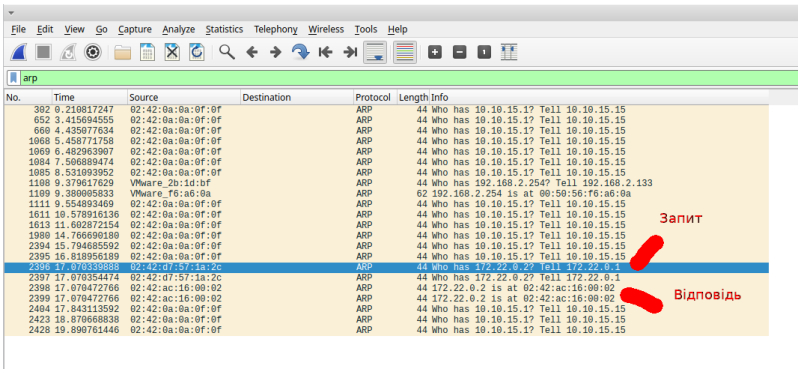
2. Для того, щоб вивантажити збережений дамп по SSH можна користуватися командою:

```
$ scp -i assets/pc1_access -P8022 root@127.0.0.1:/tmp/out.cap .
```

3. Далі, можна проаналізувати отриманий дамп більш зручним в користуванні `wireshark`. Документацію по `wireshark` можна знайти за посиланнями: `https://www.cise.ufl.edu/~helmy/F14/Wireshark_Intro_v6.0.pdf` та `https://www.wireshark.org/docs/`

2.2.3 ARP протокол, атака ARP-spoofing

Задача протоколу ARP – отримання MAC адрес вузлів, за відомою IP адресою, для встановлення каналів зв'язку всередині логічного сегменту (тому це протокол канального рівня). Нехай вузол A з адресою 10.10.15.a намагається встановити канал з вузлом B з адресою 10.10.15.b. В такому випадку, A розсилає на широкомовну адресу 10.10.15.255 ARP запит, що містить IP адресу та MAC вузла A, IP адресу вузла B. Коли запит потрапляє до вузла B, він повертає відповідь на 10.10.15.a зі своєю MAC адресою.



Протокол доволі простий, проте він є незахищеним: будь який вузол підмережі може видати себе за цільовий, повернувши ARP відповідь із своєю MAC адресою і цільовою IP. Більш того, зловмисному вузлу навіть не треба чекати запит – він сам може сформувати ARP запит зі своєю MAC та IP жертви A і відправити за адресою жертви B. Таким чином, тепер трафік від B йтиме через зловмисника і буде йому доступний. Цю атаку можно провести за допомогою утиліти `arpspoof`:

1. Перше, що потрібно зробити, це включити на машині атакуючого просування трафіку (ip forwarding):

```
$ echo 1 > /proc/sys/net/ipv4/ip_forward
```

2. Зазвичай, атакують одразу два вузли: жертву та шлюз, щоб і прямий і зворотній трафік проходив через атакуючого. Нехай, адреса шлюзу 10.10.10.1, а жертви – 10.10.10.3. Тоді атака виглядатиме таким чином:

```
$ arpspoof -i eth0 -t 10.10.10.1 10.10.10.3 2> /dev/null &
$ arpspoof -i eth0 -t 10.10.10.3 10.10.10.1 2> /dev/null &
```

   Після початку отруєння arp таблиці (arp poisoning), трафік між цими вузлами починає проходити через атакуючого.

3. Щоб припинити атаку, потрібно знайти PID відповідних процесів та вбити їх:

```
$ ps -e|grep arpspoof
 $ kill 50 53
```

2.2.4 Трансляція мережних адрес (NAT)

Механізм трансляції мережних адрес використовується для створення двостороннього каналу передачі між вузлом, що знаходиться у підмережі з вузлом, що знаходиться за межами цієї підмережі. Суть такого механізму в тому, що IP адреса відправника (до якої отримувач не має маршруту) змінюється на IP адресу шлюза (який доступний отримувачу), а на зворотньому шляху IP адреси знову замінюються на оригінальні. Таким чином, отримувач бачить лише шлюз, але при цьому IP пакети передаються до вузла, що знаходиться за NAT.

2.2.5 Дослідження протоколів канального та мережного рівнів

Ethernet, ICMP, IP,

2.3 Хід роботи

1. Отримайте SSH доступ до машини `client_pc1`.
2. Зберіть інформацію про налаштування мережі, адреси, маршрути, тощо.
3. Перейдіть за SSH на вузол `https-portal` (172.11.15.71). Очистіть arp кеш (та пересвідчитися в цьому) та спробувати пінгувати вузли `corp_gateway, corp1, corp2, http_server, ftp_server`, проаналізуйте зміни в arp кеші та дайте оцінку отриманому результату.
4. Відкрийте дві SSH сесії на `client_pc1`. В одній з них запустіть `tcpdump`, а у іншій – очистіть arp кеш та спробуйте пінгувати сусідні машини `client_pc2-client_pc4`. Вивантажте дамп трафіку та проаналізуйте за допомогою `wireshark`. Відфільтруйте трафік за ICMP та ARP. Опишіть функціонування цих протоколів на прикладі перехоплених пакетів.
5. Відкрийте у дампі будь який TCP пакет та проаналізуйте кожен рівень стеку TCP/IP: опишіть на прикладі перехопленого пакету, які поля містяться у заголовках кожного рівня, яке їх призначення.
6. Знову відкрийте дві сесії на `client_pc1` і запустіть `tcpdump`. Одночасно, запустіть `wireshark` для перехопленя трафіку зі сторони хосту, на мостовому інтерфейсі з контейнером `corp_gateway`. У другій SSH сесії спробуйте доступитися до зовнішнього ресурсу (наприклад, `curl google.com`). Поясніть роботу механізму NAT на прикладі порівняння двох дампів трафіків – зовнішнього і внутрішнього.
7. На вузлі `client_pc1` запустіть `tcpdump` та збирайте трафік впродовж 1-2 хвилин. Проведіть arp spoofing атаку на вузли `client_pc2` `client_pc3`, `ftp_server`, під час кожної з них зберіть трафік впродовж 1-2 хвилин. Порівняйте отримані дампи та надайте їм оцінку, поясніть, який реультат дала атака в кожному випадку і чому саме такий.

2.4 Вимоги до звіту

1. Звіт має містити титульну сторінку (щоб розуміти, хто його написав). Вона має включати щонайменше Ім'я виконавця, групу, дату виконання, емейл, за яким підключаєтесь до конференцій.
2. Титулка має також включати номер лабораторної роботи та версію тестової лабораторії. Це важливо, бо лабораторія буде змінюватись, іноді суттєво і потрібно буде розуміти в якому середовищі зроблені старі звіти.
3. Звіт має містити детальний опис ходу роботу із підкріпленням результів скріншотами. В т.ч., скріншоти повинні захоплювати, по можливості, ідентифікаційну інформацію – MAC-IP адреси, доменні імена, порти, ідентифікатори сесій тощо.

2.5 Додаткові питання

1. Що таке маска підмережі (netmask)
2. Що таке broadcast адреса, як вона використовується
3. Що відбудеться, якщо шлюзом за замовчуванням призначити сам вузол, на якому проводяться налаштування
4. Для чого потрібен протокол ARP і як він працює
5. Які задачі виконують протоколи канального та мережного рівня
6. Як вирішується проблема у випадку, коли розмір пакету вищого рівня (наприклад, TCP) перевищує розмір тіла повідомлення (payload) нижчого (наприклад, IP)
7. Як працює механізм NAT
8. Чи можливо провести arp-spoofing атаку на вузли з іншої логічної підмережі
9. Чи можливо провести arp-spoofing атаку на вузли, що знаходяться за межами NAT атакуючого