# Лабораторна робота 2.
# Стеки мережевих протоколів.
# Аналізатор мережевого трафіку Wireshark

**Мета роботи:** засвоєння функцій модулів різних рівнів еталонної моделі OSI, процедури інкапсуляції та формування повідомлень для передачі в мережу; ознайомлення та вивчення аналізатора мережевого трафіку Wireshark.

**План виконання лабораторної роботи**

1. Ознайомитися та засвоїти теоретичні відомості про еталонну модель взаємодії відкритих систем OSI та стек мережевих протоколів TCP/IP.
2. Ознайомитися з можливостями аналізатора мережевого трафіку Wireshark.
3. За допомогою аналізатора Wireshark виконати захоплення та провести аналіз мережевих пакетів.

## 1. ТЕОРЕТИЧНІ ВІДОМОСТІ

Дивитись конспект лекції.

## 2. РОБОТА З АНАЛІЗАТОРОМ ТРАФІКУ Wireshark

При виконанні системного адміністрування комп'ютерних мереж доволі часто доводиться мати справу зі складними ситуаціями, в яких не допомагають ані інструменти збору статистики (наприклад, netstat), ані стандартні утиліти на основі протоколу ICMP (ping, traceroute тощо). В таких випадках часто використовуються спеціалізовані діагностичні утиліти, які дозволяють прослуховувати мережевий трафік і аналізувати його на рівні блоків передачі окремих протоколів. Вони називаються аналізаторами трафіку або сніферами) і дозволяють, по-перше, локалізувати проблеми мережі та точніше їх діагностувати, а по-друге, виявляти в мережі не тільки різні типи трафіків, а й шкідливе програмне забезпечення.

На сьогодні існує значна кількість систем моделювання роботи комп'ютерних мереж і аналізаторів мережевого трафіку. В даній роботі необхідно ознайомитись з особливостями роботи аналізатора мережевого трафіку Wireshark.

### 2.1. Аналізатор мережевого трафіку Wireshark

Одним з найбільш поширених і популярних аналізаторів трафіку є Wireshark. Існують версії для різних операційних систем: Linux, Windows, MacOS, FreeBSD, Solaris. Wireshark також використовує бібліотеку libpcap для збирання пакетів. Wireshark має зручний графічний інтерфейс.

Для того щоб перехоплювати протокольні одиниці інформації (PDUs), які

передаються мережею, потрібно мати фізичне підключення до відповідної мережі, а також запущену копію Wireshark на комп'ютері. Початкове вікно програми виглядає так, як показано на рисунку 1.8.

Щоб розпочати захоплення пакетів потрібно, при необхідності, ввести фільтр захоплення в поле «Enter a capture filter» та вибрати інтерфейс. У вікні,що відкриється, можна спостерігати за процесом захоплення трафіку на вибраному інтерфейсі (рис. 1.9).

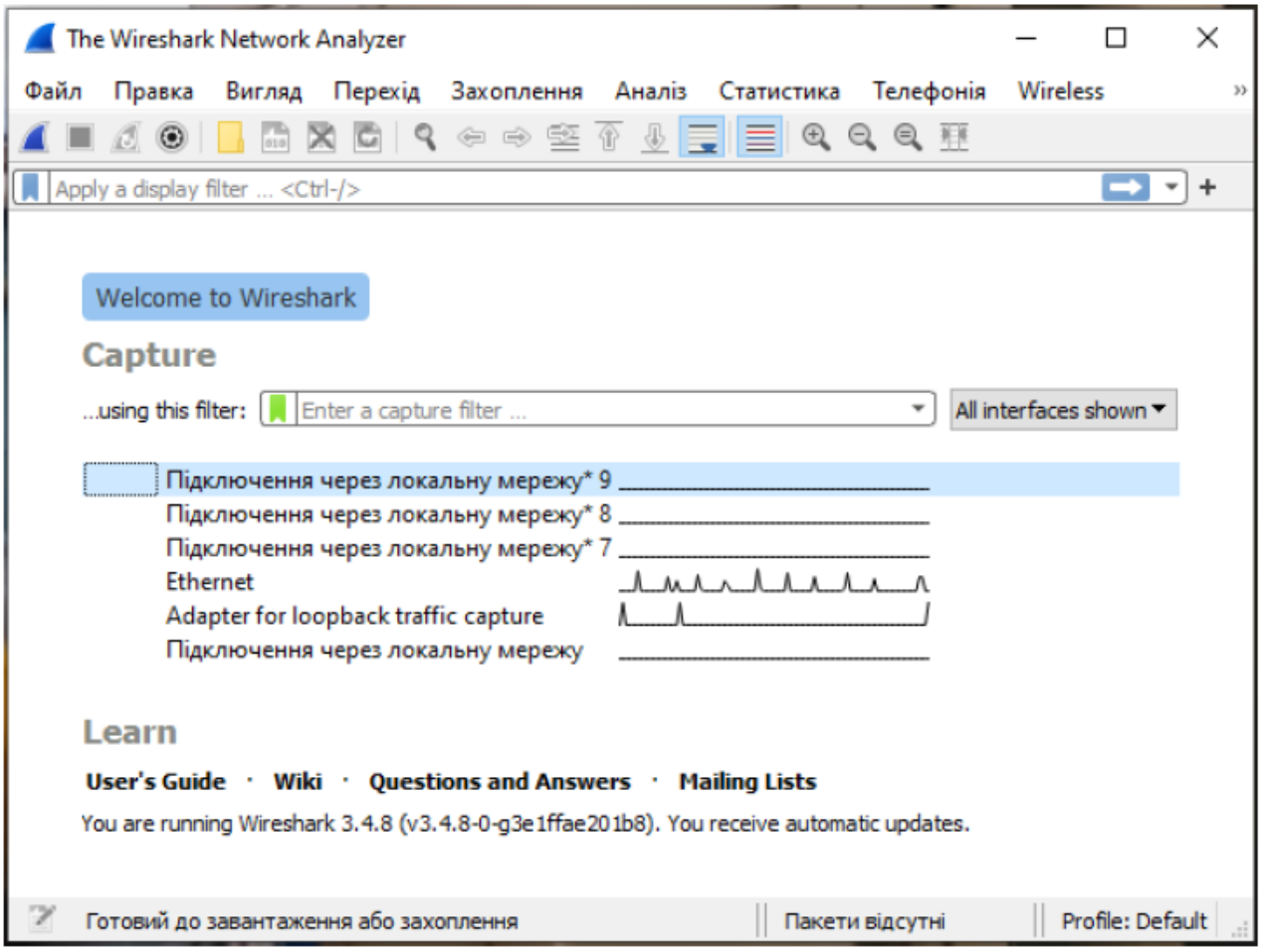


Рисунок 1.8 - Початкове вікно програми Wireshark

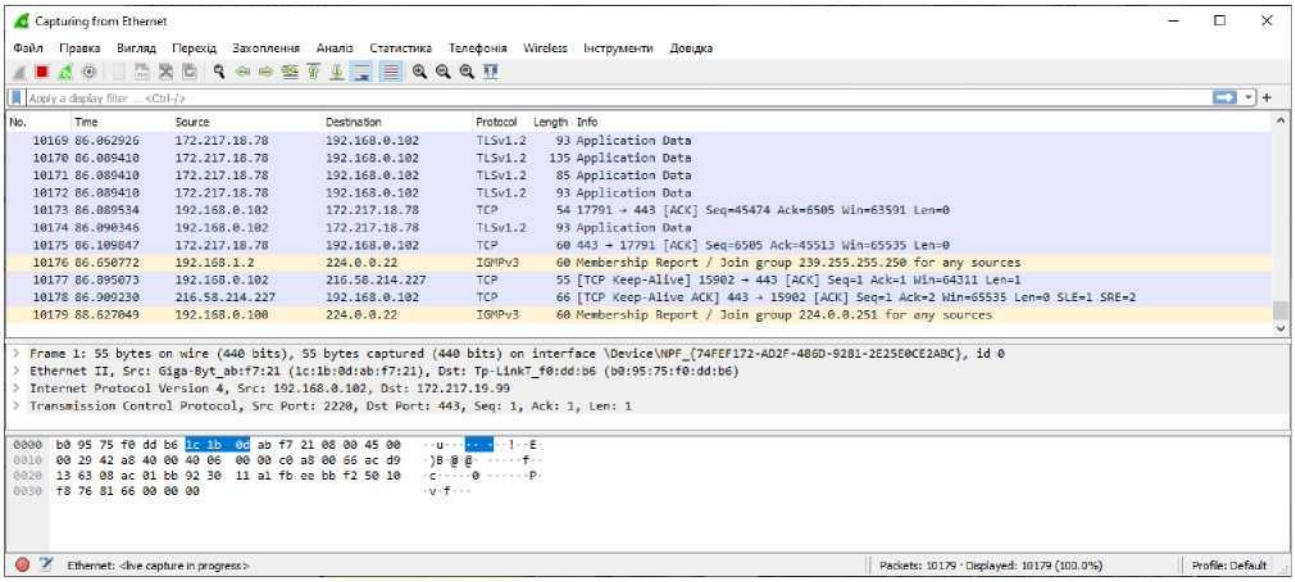


Рисунок 1.9 - Головне вікно програми Wireshark

Потрібно перевірити такі параметри:

- чи працює мережевий адаптер у, так званому, promiscuous mode;. якщо даний режим не обрано, то програма зможе захоплювати лише ті пакети, які були адресовані комп'ютеру, на якому вона встановлена (рис. 1.10);
- чи ввімкнено Resolve MAC addresses; дана опція дозволяє Wireshark транслювати знайдені мережеві адреси в імена (рис. 1.11).

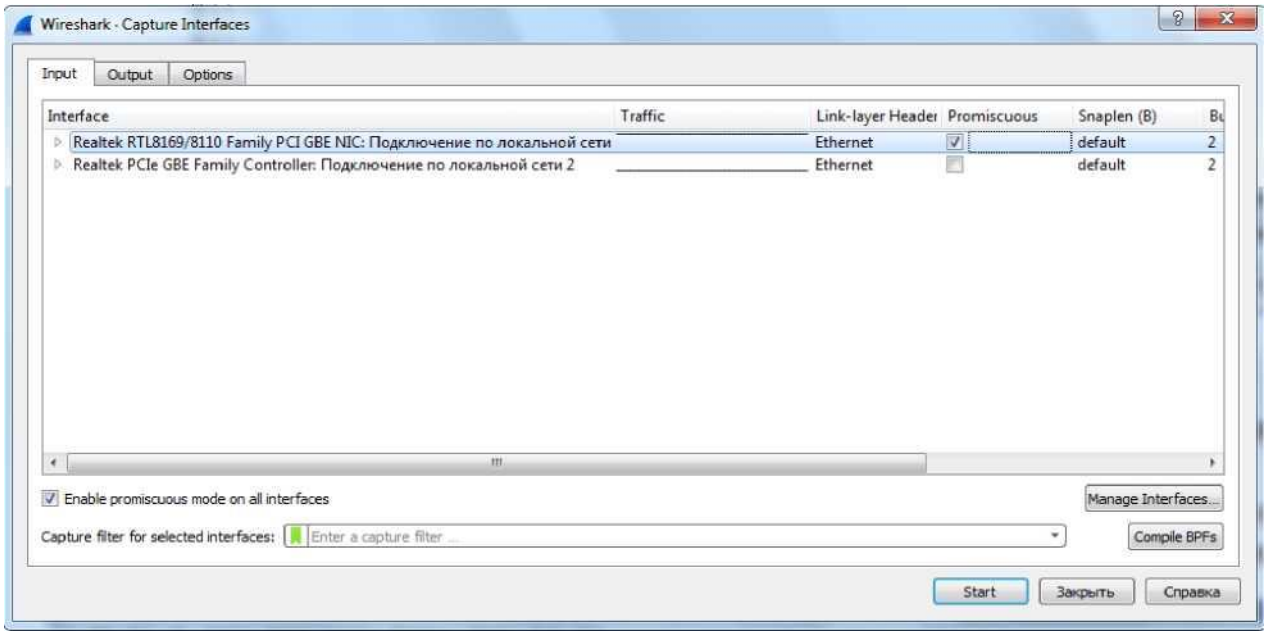


Рисунок 1.10 - Вікно налаштування параметрів захоплення трафіку

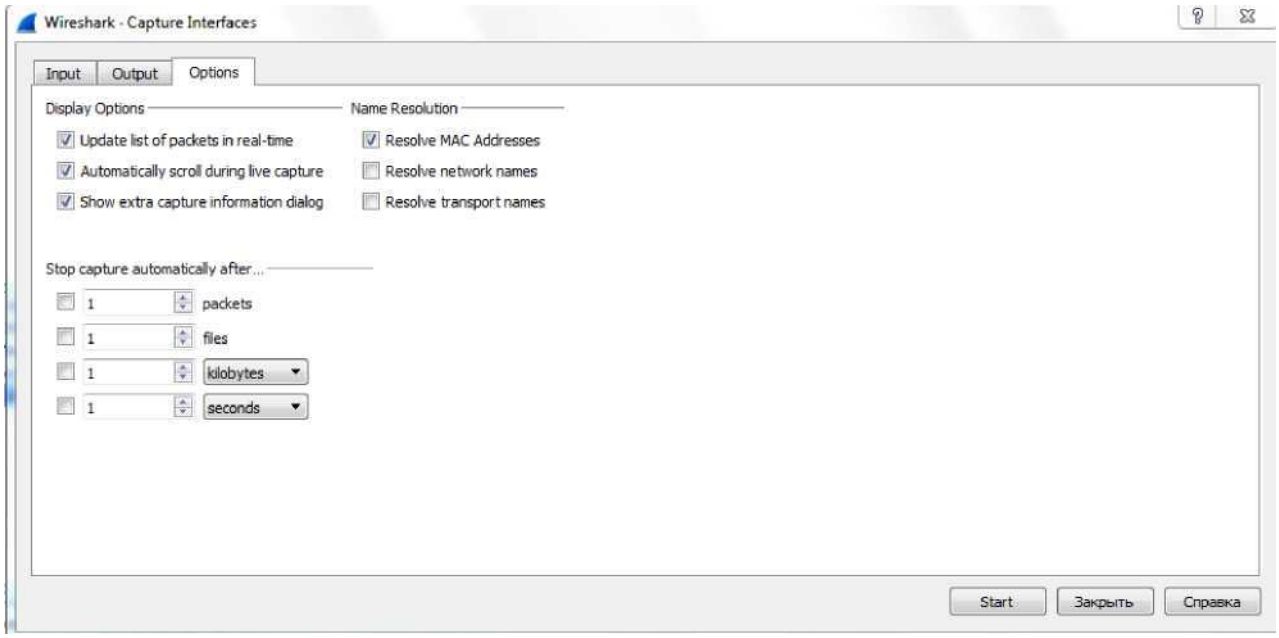


Рисунок 1.11 - Вікно налаштування параметрів перетворення імен

Після того, як налаштування завершені, потрібно натиснути кнопку Start, у результаті чого розпочнеться процес перехоплення пакетів.

Для того щоб перервати процес, потрібно натиснути на кнопку Stop. У результаті захоплення пакетів буде припинено (рис. 1.12). У даному вікні

зображено результат виконання команди ping. Застосувати фільтр захоплення можна і після запуску програми. Для цього потрібно зупинити захоплення кнопкою Stop, увійти в меню Capture, потім у меню Options і у вікні, що відкриється, ввести необхідний фільтр.

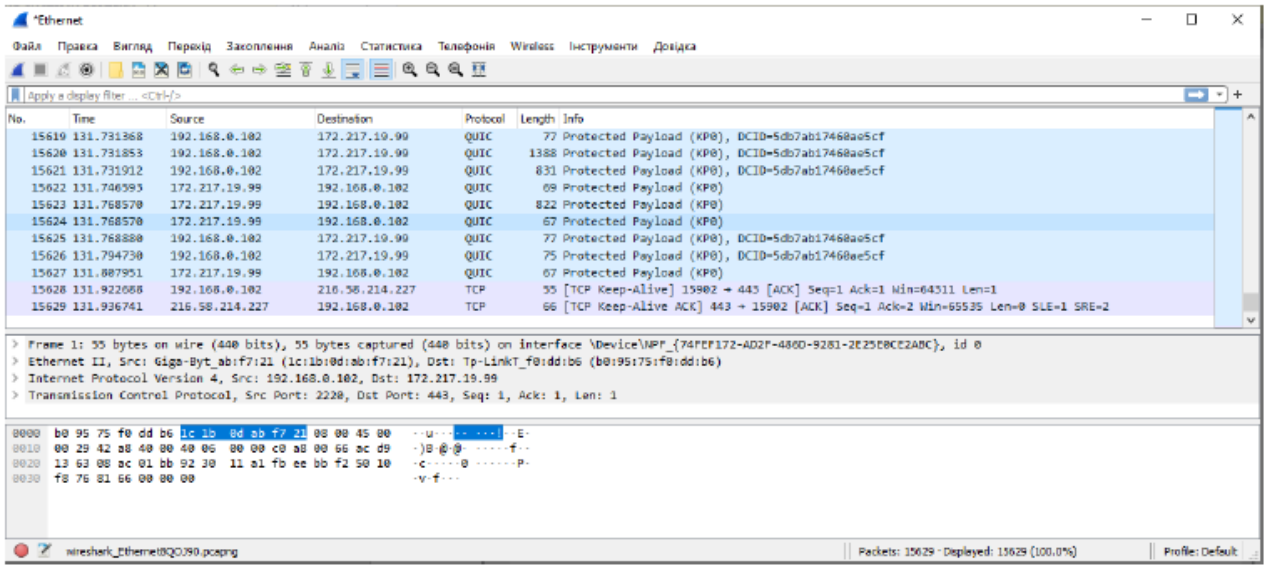


Рисунок 1.12 – Головне вікно програми з результатами захоплення мережевого трафіку

Головне вікно програми складається із трьох частин:

- поле з переліком пакетів, що містить стислу інформацію про кожний захоплений пакет; на цій панелі можна обрати потрібний пакет і його вміст відобразиться у двох інших панелях;
- поле зі структурою пакетів, яке дозволяє вивчити вміст пакету більш детально (інформація про протокол, службові поля пакету);
- поле з байтовою/символьною структурою пакетів показує фактичний вміст обраного пакету в шістнадцятковому/символьному коді.

У Wireshark використовується велика кількість різних фільтрів. Вони поділяються на два види:

- фільтри перехоплення трафіку (capture filters);
- фільтри відображення (display filters).

Перехоплення дозволяє економити оперативну пам’ять і місце на жорсткому диску. Фільтр - це вираз із групи примітивів, об’єднаних, при необхідності, логічними функціями (and, or, not). Записується цей вираз у полі «Capture Filter» діалогового вікна «Capture options». Фільтри, які використовуються часто, можна зберігати в профілі для повторного використання.

На рисунку 1.13 наведені скріншоти екрану, отримані при роботі програми Wireshark, що ілюструють взаємне розташування Ethernet-, IP- та TCP- заголовків у кадрі.

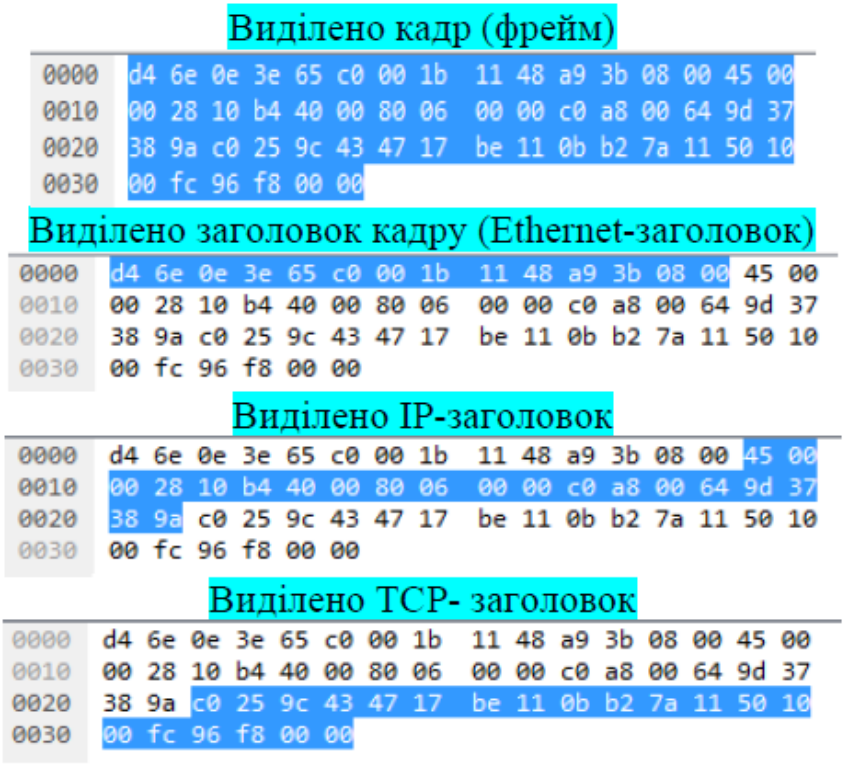


Рисунок 1.13 – Скріншоти екрану, отримані при роботі програми Wireshark

Фільтри відображення працюють з вже перехопленим трафіком. Вписати фільтр відображення можна прямо у відповідне поле (працює випадаючий список-підказка) головного вікна програми після кнопки «Filter» (цією кнопкою можна викликати профіль для фільтрів, які часто використовуються). При натисканні кнопки «Expression...» відкривається конструктор фільтрів.

Захоплені дані можна зберегти у файл. З цими даними можна працювати пізніше. Для цього потрібно натиснути «Файл», потім «Відкрити» і вибрати збережений файл.

Щоб добре розуміти значення фільтрів і що саме вони показують, необхідно добре розуміти роботу мережі.

Фільтри можуть складатись із ключових слів та скорочень, десяткових та шістнадцяткових чисел. Наприклад, можна відобразити тільки ті TCP пакети, які містять рядок **kpi** , якщо використовувати наступний фільтр:

**tcp contains kpi**

Тут **tcp** застосовується для вибору протоколу, **contains** - ключове слово, яке означає, що знайдені пакети міститимуть шуканий вміст.

У таблиці 1.1 перелічені оператори фільтрів.

Логічні оператори, приклади яких наведені в таблиці 1.2, дозволяють

створювати детальні фільтри з використанням відразу декількох умов.

Рекомендується додатково використовувати дужки для отримання тих значень, які необхідні.

Таблиця 1.1 - Логічні оператори фільтрів Wireshark

| Англомовний синтаксис | С-подібний синтаксис | Опис оператора | Зразок застосування |
|---|---|---|---|
| eq | == | Дорівнює | ip.src==10.0.0.5 |
| ne | != | Не дорівнює | ip.src!=10.0.0.5 |
| gt | > | Більше ніж | frame.len > 10 |
| lt | < | Менше ніж | frame.len < 128 |
| ge | >= | Більше чи дорівнює | frame.len ge 0x100 |
| le | <= | Менше, чи дорівнює | frame.len <= 0x20 |
| contains | | Протокол, поле, або зРІз(зІІСЄ) включає значення | sip.To contains "a1762" |
| matches | ~ | Протокол або текстове поле збігається з Perl-сумісним регулярним виразом | http.host matches "acme\\.(org\|com\|net)" |
| bitwise_and | & | Симетричне І | tcp.flags & 0x02 |

Таблиця 1.2 - Приклади логічних операторів

| Оператор | Опис |
|---|---|
| **and/&&** | Логічне І, дані виводяться, якщо вони відповідають обом частинам фільтру. Наприклад, фільтр ip.src==192.168.1.1 and tcp покаже тільки пакети, які надходять від 192.168.1.1 і які пов’язані з протоколом TCP. Будуть показані тільки дані, що задовольняють обом умовам. |
| **or/\|\|** | Логічне АБО, достатньо щоб тільки одна умова була істиною; якщо обидві умови є істиною, то це теж підходить. Наприклад, фільтр tcp.port==80 or tcp.port==8080 покаже TCP пакети, які пов’язані з портом 80 або 8080. |
| **not/!** | Логічне Ні використовується, коли потрібно виключити деякі пакети. Тобто будуть показані всі пакети, крім тих, які задовольняють умові, що слідує після Ні. Наприклад, фільтр !dns покаже всі пакети, крім DNS. |

Приклади логічних операторів:
показати HTTP або DNS трафік:

**http or dns**

Показати будь-який трафік, крім ARP, ICMP и DNS:

**!(arp or icmp or dns)**

Слід зазначити, що в аналізаторі Wireshark існує значна кількість фільтрів, які можуть використовуватись для фільтрації трафіку конкретного протоколу. Основні типи цих фільтрів наведено у Додатку А.

## Завдання

1. При виконанні роботи використовується програмне забезпечення для аналізу протоколів комп'ютерних мереж Wireshark. Запустити відповідну програму.
2. Вибрати інтерфейс для захоплення трафіку (меню Capture/Interface) та активізувати режим захоплення.
3. Скопіювати через мережу файл розміром кілька десятків Мбайт.
4. Завершити захоплення трафіку та перейти до режиму аналізу.

## Запитання

1. Поясніть поняття еталонної моделі взаємодії відкритих систем OSI?
2. Яке призначення еталонної моделі взаємодії відкритих систем?
3. Які рівні містить еталонна модель OSI та їх основні функції?
4. На які протокольні одиниці розбивається інформація для передачі даних мережею?
5. Які функції виконуються на транспортному рівні?
6. Які функції виконуються на мережевому рівні?
7. Які функції виконуються на фізичному рівні?
8. Які функції виконуються на канальному рівні?
9. Поясніть різницю між кадром, пакетом, дейтаграмою, сегментом.
10. На які рівні поділяється стек мережевих протоколів TCP/IP?
11. Поясніть процедуру інкапсуляції?
12. Призначення та функції аналізатора мережевого трафіку Wireshark.
13. Які види фільтрів підтримує програма Wireshark?
14. Наведіть приклади застосування різних видів фільтрів Wireshark.