Лабораторна робота 5.
Аналіз мережевого трафіку за допомогою програми Wireshark

Мета роботи: ознайомитись з програмним забезпеченням для аналізу мережевого трафіку, отримати практичні навички використання програмного аналізатора протоколів Wireshark, закріпити знання з архітектури комп'ютерних мереж, поглиблено вивчити стек протоколів TCP/IP.

Теоретичні відомості

Sniffer (від англ. to sniff – нюхати) – це мережний аналізатор трафіку, програма або програмно-апаратний пристрій, призначений для перехвату та наступного аналізу, або тільки аналізу мережного трафіку, призначеного для інших вузлів. Перехват трафіку може відбуватися:

- звичайним "прослуховуванням" мережного інтерфейсу; метод ефективний при використанні в сегменті концентраторів (hub) замість комутаторів (switch), в іншому випадку метод малоефективний, оскільки на сніфер попадають лише окремі кадри, які призначені приймаючому вузлу;
- підключенням сніферу в розрив каналу;
- відгалуженням (програмним чи апаратним) трафіку і направленням його копії на сніфер;
- через аналіз побічних електромагнітних випромінювань та відновленням таким чином трафіку, що прослуховується;
- через атаку на канальному рівні (2-му) або мережному рівні (3-му), яка приводить до перенаправлення трафіку жертви або всього трафіку сегменту на сніфер з наступним поверненням трафіку на потрібну адресу .

Сніфери застосовуються як в благих, так і деструктивних цілях. Аналіз трафіку, що проходить через сніфер, дозволяє:

- відслідковувати мережну активність додатків;
- відлагоджувати протоколи мережних додатків;
- локалізувати несправність або помилку конфігурації;
- знаходити паразитний, вірусний та закільцьований трафік, наявність котрого збільшує навантаження мережного обладнання та каналів зв'язку;
- виявити в мережі шкідливе ПО, наприклад, мережні сканери, флудери, троянські програми, клієнти піринговых мереж та інші;
- перехватити любий незашифрований (а інколи і зашифрований) трафік користувача з ціллю дізнавання паролів та іншої інформації

Поступово із інструментів, призначених тільки для діагностики, сніфери перетворилися в засоби для дослідження та навчання. Наприклад, вони постійно використовуються для вивчення динаміки і взаємодії в мережах. Зокрема, вони дозволяють легко та наглядно вивчати тонкощі мережних протоколів. Спостерігаючи за даними, які відправляє протокол, можна глибше зрозуміти його функціонування на практиці, а заодно побачити, коли деяка конкретна реалізація робить не у відповідності зі специфікацією. На сьогоднішній момент існує достатня кількість хороших реалізацій сніферів. Деякі з них:

Tcpdump (http://www.tcpdump.org/) – консольний варіант сніферу. Працює на найбільш поширених на сьогоднішній день ОС;

Wireshark (http://www.wireshark.org/) до недавнього часу був відомий під іменем Ethreal;

WinDump http://www.winpcap.org/windump;

Програма Wireshark являеться однією з самих зручних реалізацій сніферів, яка доступна для багатьох ОС і поширюється безкоштовно. На рис.1 зображена структура мережної підсистеми ОС. Уся базова інфраструктура реалізована у вигляді драйверів та працює в режимі ядра. Процеси користувача та реалізації прикладних протоколів, зокрема інтерфейс сніферу працюють в режимі користувача.

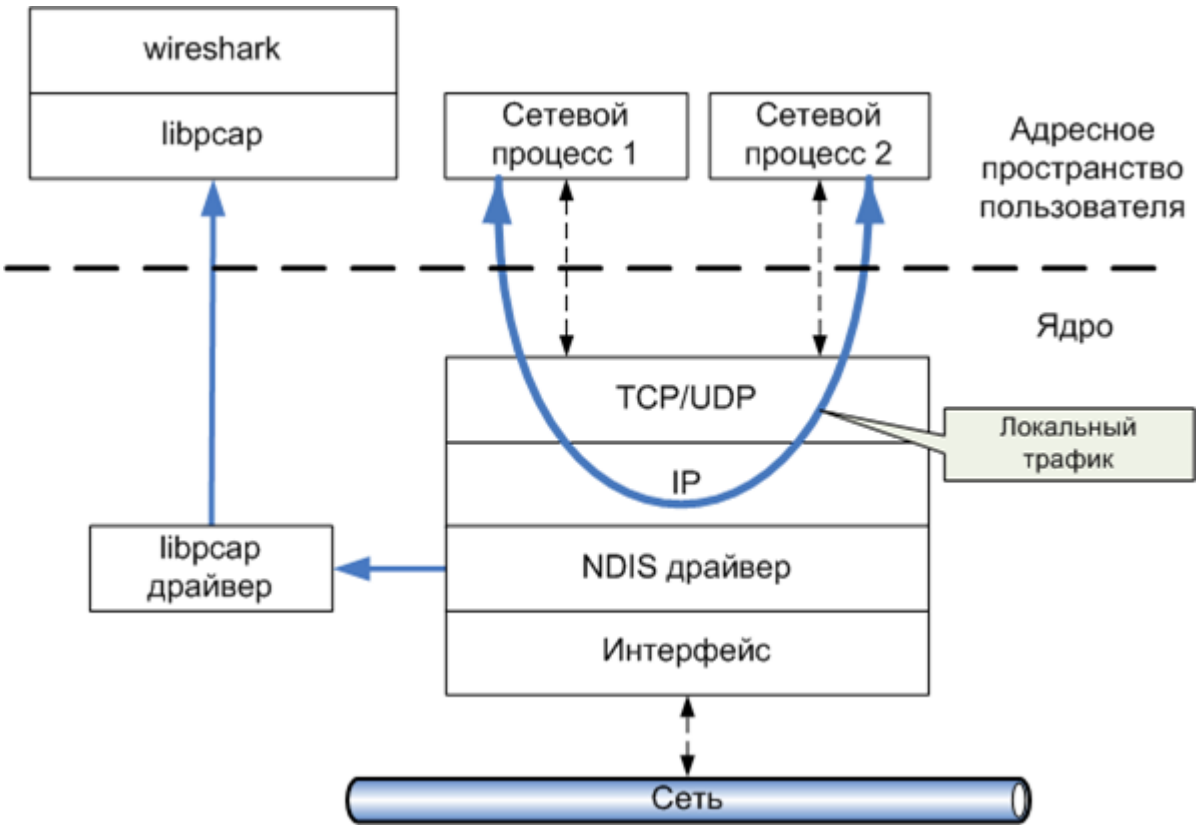

Рис.4.1. Принцип захвату мережного трафіку.

Основними компонентами сніферу є: драйвер для захоплення пакетів (libpcap драйвер), інтерфейсна бібліотека (libpcap) та інтерфейс користувача (Wireshark). Бібліотека libpcap (реалізація під ОС Windows носить назву WinPcap - http://www.winpcap.org) – універсальна мережна бібліотека, яка самостійно реалізує велику кількість мережних протоколів та працює безпосередньо з NDIS драйверами (Network Driver Interface Specification) мережних пристроїв. На базі даної бібліотеки реалізована велика кількість мережних програм, в тому числі сніфер Wireshark.

Сніфери використовують бібліотеку в режимі "захоплення" пакетів, тобто може отримувати копію всіх даних що проходить через драйвер мережного інтерфейсу. Зміни в самі дані не вносяться. Якщо локальний трафік не проходить через драйвер мережного пристрою (див. рис.Д9), то він не буде видимий сніфером.

Wireshark дозволяє в режимі реального часу захватувати пакети з мережі, та аналізувати їх структуру. Також можна аналізувати структуру

пакетів з файлу, який вміщує трафік, отриманий, наприклад програмою «tcpdump» (unix/linux).

Головне вікно програми

Wireshark, рис. 4.2.

Завантажити програму безкоштовно можна з офіційного ресурсу: https://www.wireshark.org/download.html .

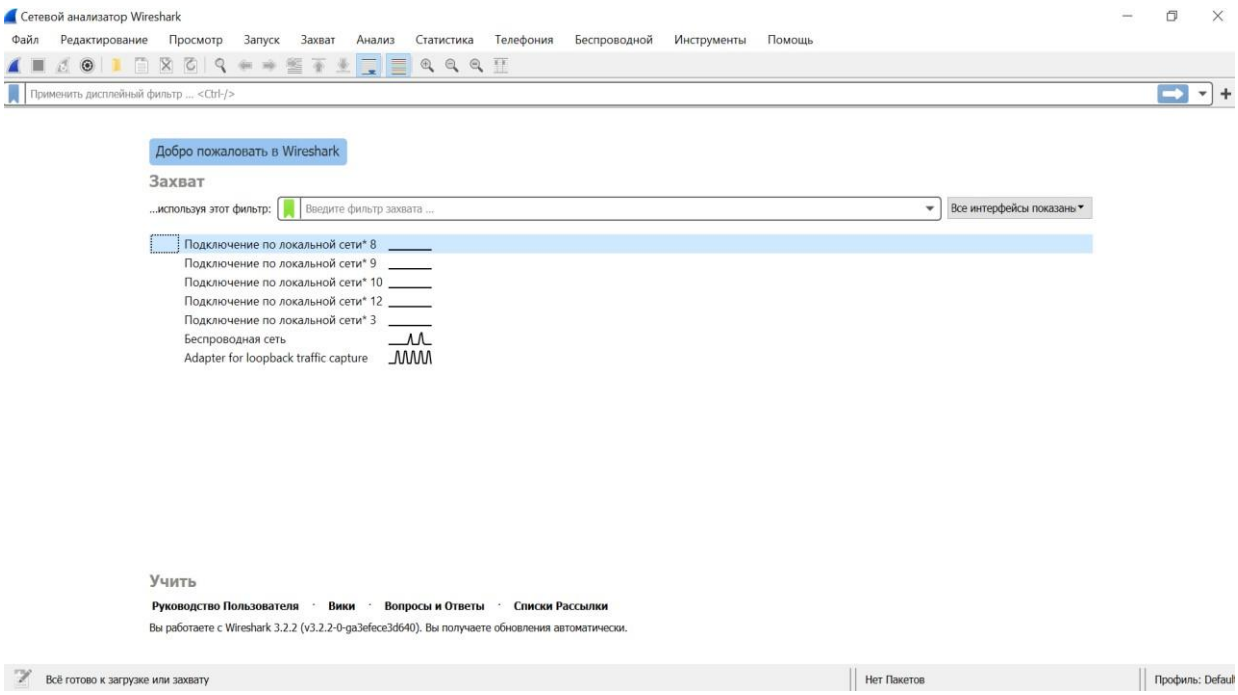

Рис. 4.2. Головне вікно програми Wireshark

У верхній частині вікна знаходиться меню і панель інструментів. Нижче розміщений фільтр, у якому можна задавати критерії фільтрації.

Для того, щоб мати уявлення про можливості програми Wireshark коротко розглянемо її інтерфейс (рис. 4.3).

Як бачимо з рис. 4.3, інтерфейс програми містить три робочі області (панелі), які забезпечують різний ступінь деталізації відомостей про перехоплені пакета:

- верхня панель *(Packet List)*, що містить список перехоплених кадрів з коротким описом;

- середня панель *(Packet Details*), на якій показано дерево протоколів, що використовувалось під час передачі кадру, вибраного у верхньому вікні;
- нижня панель *(Packet Bytes)*, на якій можна побачити вміст вибраного кадру у шістнадцятковому (зліва) та текстовому (справа) поданні у кодах ASCII.

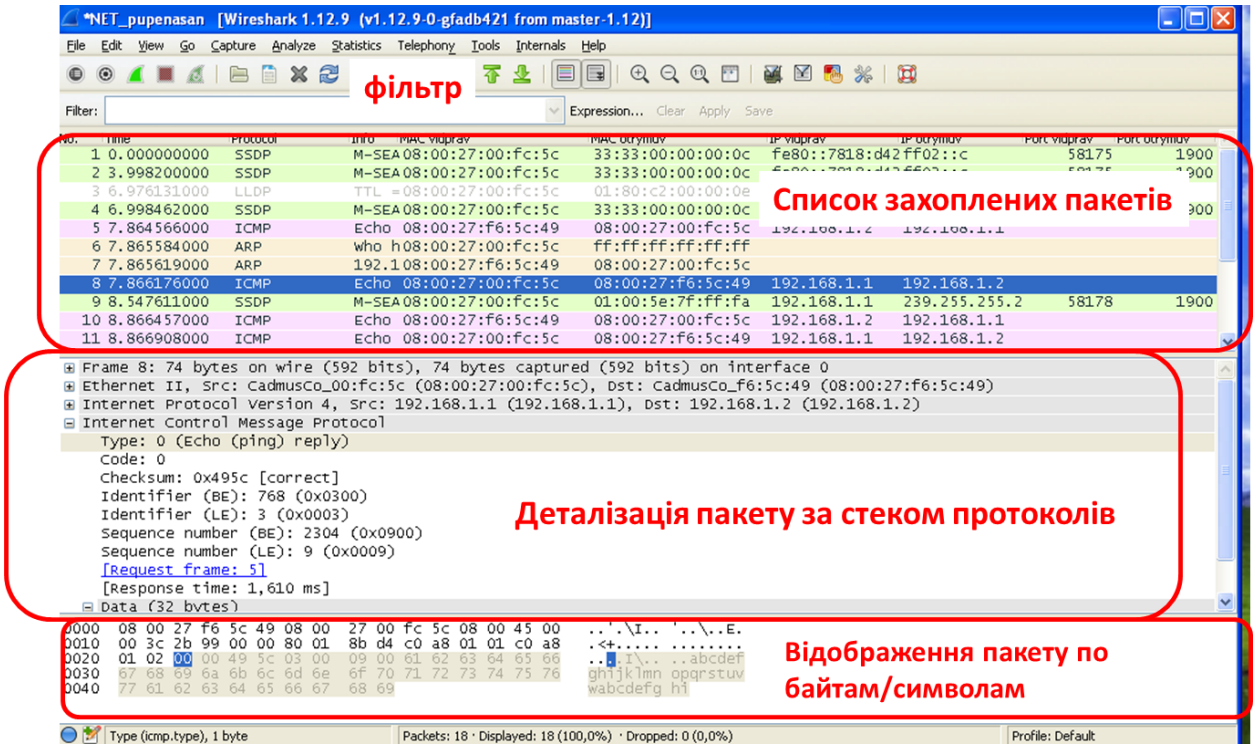

Рис. 4.3. Інтерфейс програми Wireshark

Панель списку пакетів вміщує всі пакети, які захоплені з мережі, попередньо відфільтровані через умову фільтру (Filter). Список можна відсортувати по будь-якому полю (в прямому або зворотному порядку) – для цього треба натиснути по заголовку відповідного поля. Кожний рядок вміщує наступні поля (по замовченню)

- порядковий номер пакету (No.);
- час надходження пакету (Time);
- джерело пакету (Source);
- пункт призначення (Destination);
- протокол (Protocol);
- інформаційне поле (Info).

Список полів які будуть відображатися в панелі списку як колонки налаштовується в Edit/Perferencis/Columns (рис.4.4). Для того, щоб зміни вступили в силу, після добавлення колонок треба натиснути “Apply”.

Панель деталізації пакетів показує ієрархічну протокольну структуру вибраного в панелі списку пакету. Тобто вказується вкладеність PDU (пакетів) один в одного, згідно відомих Wireshark протоколів. При виділенні PDU, його вміст в байтах показується на панелі побайтового відображення. При розкриванні PDU (+), виводиться детальний аналіз його полів. Так, наприклад, на рис.4.3 видно, що в кадр EthernetII вкладений пакет IPv4, в який в свою чергу вкладений пакет ICMP (Internet Control Message Protocol). Для виділених пакетів видно поля та їх значення, які розділені через двокрапку. Так, наприклад, на рис.4.3 видно, що поле ICMP-пакету з назвою Type має значення 0, що вказує на тип «Echo (ping) reply».

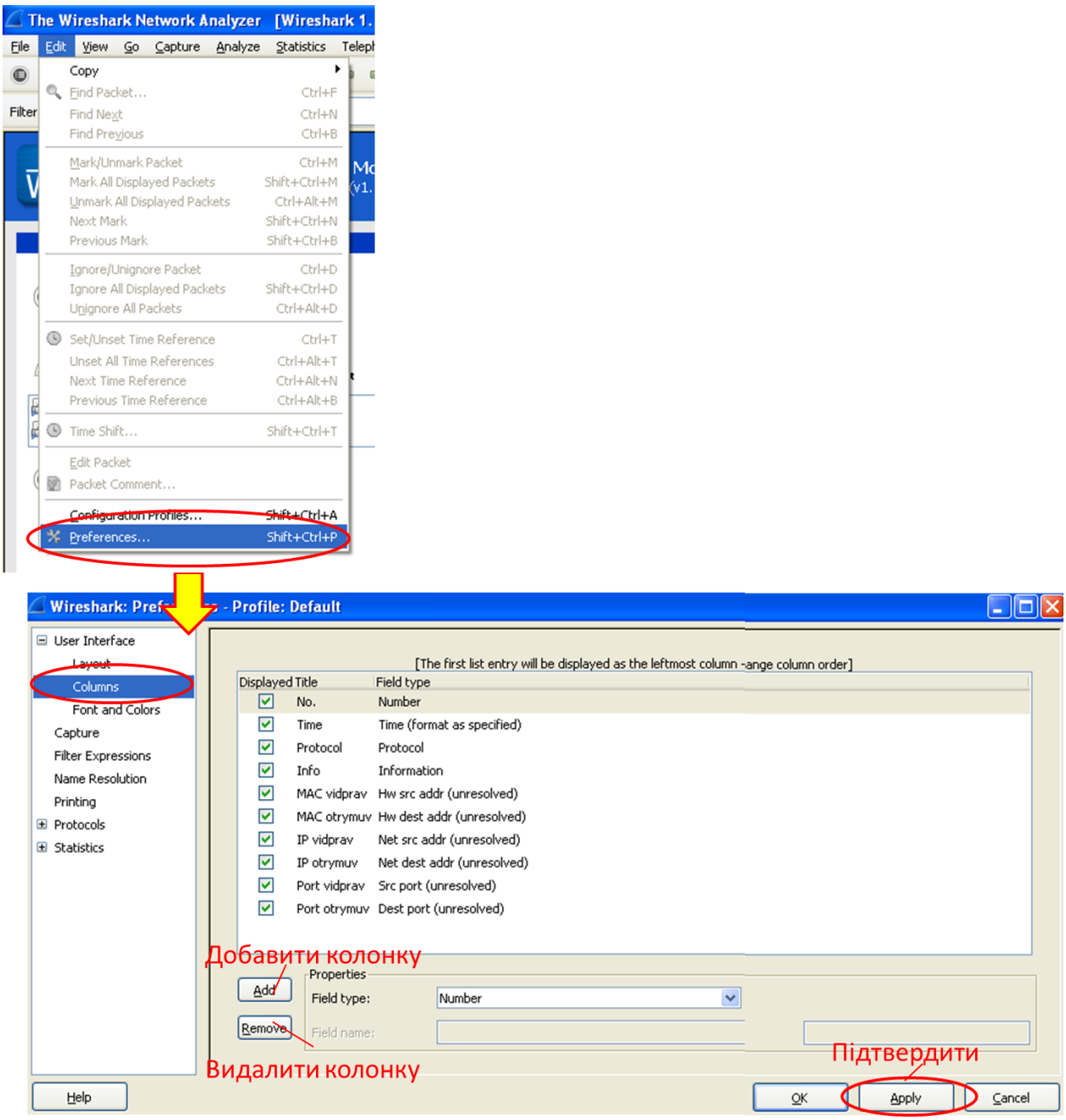

Рис.4.3. На панелі побайтового відображення пакетів для виділених пакетів показується його 16-кове та символьне представлення.

Кількість параметрів, що виводяться на екран, може бути змінена (зменшена) у процесі налаштування.

Вибравши певний кадр зі списку, що міститься у верхній панелі, можна переглянути детальнішу інформацію про нього на середній та нижній панелях.

Вибраний кадр відзначається підсвічуванням.

Параметр *Time* у списку пакетів є відносним часом отримання пакета. Відлік проводиться за відношенням до часу прийому першого пакета. Параметри відображення часу можна змінити у налаштуваннях.

Параметри *Source* та *Destination* містять відповідно адреси джерела та віддаленого об'єкта. Здебільшого - це *IP-адреси,* а у разі перехоплення кадрів *ARP*- та *RARP*-протоколів - *MAC-адреси.*

Різні протоколи на панелі *Packet List* підсвічуються різними кольорами, що додає наочності та спрощує аналіз.

На середній панелі - *Панелі Деталей* - виводиться детальна інформація про кадр *(frame*) з номером, що був вибраний на панелі *Packet List.* Інформація подана у структурованому за рівнями вигляді дерева протоколів. Конфігурацію інтерфейсу можна змінювати за допомогою меню *View.*

Детальніше про програму *Wireshark* можна прочитати на сайті розробників www.wireshark.org.

Формат пакета протоколу IPv4

Протокол Internet створений для використання в об'єднаних системах комп'ютерних комунікаційних мереж з комутацією пакетів. Протокол Internet забезпечує передавання пакетів від відправника до одержувачів без встановлення з'єднання, а отже, без забезпечення гарантії доставки. За необхідності протокол забезпечує фрагментацію та дефрагментацію пакетів.

Перше поле (Version) займає 4 біти і відведене під версію протоколу IP, яка своєю чергою, однозначно визначає формат заголовка.

Поле HLEN (Header Length) також займає 4 біти і вказує довжину заголовка в 32-бітних словах. Мінімальна довжина заголовка - 20 байт, тобто п'ять 32-бітних слів, максимальна - 60 байт. Більшість IP-дейтарам мають заголовок мінімальної довжини, для них HLEN=5.

Поле Total Length IP-пакета (загальна довжина) займає 2 байти і вказує на загальну довжину пакета у байтах з урахуванням довжини заголовка і поля даних. Максимальне значення загальної довжини - 65535 байт, проте рідко дейтаграма має розмір більший, ніж 1500 байт.

Поле Identification (ідентифікатор пакета) займає два байти і визначає номер конкретного пакета. Він повинен бути унікальним для цієї пари відправ- ник-адресат упродовж усього часу існування пакета. Кожен пакет, проходячи через Internet, можливо буде фрагментовано на менші частини (фрагменти).

Ідентифікатор слугує для того, щоб віддалена машина могла визначити пакет, до якого належить фрагмент. Усі фрагменти пакета мають той самий ідентифікатор.

Далі йде резервне поле в 1 біт, в якому записано нуль, і два однобітні прапорці: DF (Don’t Fragment - не фрагментувати) і MF (More Fragments - більше фрагментів).

Встановлений в 1 біт DF забороняє маршрутизатору фрагментувати пакет і використовується, якщо віддалена машина не здатна його зібрати. У цьому випадку пакет мусить бути направлений поза мережею, що пропонує фрагментацію, або буде відкинений.

Наступний біт MF встановлюється вів усіх проміжних фрагментах. В останньому фрагменті він встановлюється в 0, і це говорить про те, що цей фрагмент є останнім фрагментом пакета, що був фрагментований. Треба зазначити, що версія протоколу IPv6 забороняє фрагментацію у маршрути-заторах.

Поле зміщення фрагмента Fragment Offset займає 13 біт і показує зміщення у байтах поля даних цього пакета від початку загального поля даних пакета, що був фрагментований.

Поле Time to Live (час життя) займає один байт і визначає граничний термін, упродовж якого пакет може переміщатись мережею. Час життя вимірюється у секундах і задається станцією-джерелом, його максимальне значення - 255 секунд. Маршрутизатори та інші вузли мережі після закінчення кожної секунди віднімають одиничку від поточного значення часу життя; одиниця віднімається і в тому випадку, коли затримка менша, ніж одна секунда. Оскільки сучасні маршрутизатори рідко опрацьовують пакет більше однієї секунди, то час життя можна вважати таким, що дорівнює максимальній кількості вузлів (пересилань, hops), через які дозволено пройти пакету до місця призначення. Якщо час життя пакета закінчиться до того, як він прибуде до вузла адресата, пакет знищується.

Поле протоколу верхнього рівня Protocol займає один байт і вказує, якому протоколу верхнього рівня належить пакет, наприклад, протоколу TCP (значення поля - 6) або UDP (значення поля - 17), ICMP, OSPF.

Поле контрольна сума Header Checksum займає 2 байти і використовується для контролю достовірності заголовка. Визначається контрольна сума по усіх 16-бітових словах заголовка. Оскільки деякі поля в процесі передачі пакета змінюються (наприклад, час життя), то контрольна сума заново знаходиться за кожної обробки заголовка IP-пакета. Якщо під час прийому пакета при перевірці контрольної суми виявлено помилку, то пакет відкидається.

Поля IP-адреси джерела (Source Address) і IP-адреси призначення (Destination Address) займають по 4 байти.

Поле опції Options є необов’язковим і переважно використовується тільки під час відлагодження мережі. Це поле складається з підполів, кількість яких може бути довільною. У підполях може бути вказаний маршрут для передачі пакета, можуть реєструватись маршрутизатори, через які проходив пакет, можуть задаватись часові позначки, дані для

системи безпеки. За допомогою нулів поле опцій (padding) вирівнюється до 32-бітової границі.

ARP-повідомлення, формат та аналіз

ARP (Address Resolution Protocol, протокол визначення адреси) - це протокол канального рівня, призначений для визначення MAC-адреси за відомою IP-адресою. Визначення локальної адреси відбувається тільки для IPпакетів, що відправляються, оскільки заголовки створюються у момент відправлення.

Існує також протокол, що розв'язує зворотну задачу, - знаходження IPадреси за відомою локальною адресою. Він називається реверсний ARP {Reverse Address Resolution Protocol, RARP) і використовується під час старту станцій, що в початковий момент не знають своєї IP-адреси, але знають MACадресу свого мережевого адаптера.

У локальних мережах для пошуку у мережі вузла із заданою IP-адресою протокол ARP використовує широкомовні кадри протоколу канального рівня (MAC-адреса - FF:FF:FF:FF:FF:FF). Вузол, який повинен виконати відображення IP-адреси на локальну адресу, формує ARP-запит, вкладає його у кадр протоколу канального рівня, наприклад, кадр Ethernet. В ARP-запиті вказується MAC-адреса відправника {Sender Hardware address, SHA) та IP-адреса призначення {Target protocol address, TPA), для якої треба знайти MAC-адресу. Кадр розсилається широкомовно. Усі вузли локальної мережі отримують ARP- запит і порівнюють вказану там шукану IP-адресу з власною. У разі їх збігу вузол формує ЛЛР-відповідь, у якій вказує свою IP-адресу {Sender protocol address, SPA) і свою локальну адресу {Sender hardware address, SHA). ARP- відповідь відсилається вже направлено, оскільки в ARP-запиті відправника була вказана його локальна адреса.

ARP-запити і відповіді використовують один і той самий формат кадру. Оскільки локальні адреси можуть у різних типах мереж мати різну довжину, то формат пакета протоколу ARP залежить від типу мережі.

Протокол TCP

Протокол TCP використовується у тих випадках, коли потрібна надійні доставка повідомлень. Він звільняє прикладні процеси від необхідності використовувати тайм-аути і повторні передачі для забезпечення надійності. Найтиповішим прикладним процесом, що використовує протокол TCP, є протокол передачі файлів FTP (File Transfer Protocol).

Протокол TCP вимагає, щоб усі відправлені дані були підгверджені стороною, що прийняла їх. Він використовує тайм-аути і повторні передачі для забезпечення надійної доставки. Відправникові можна передавати деяку кількість даних, не чекаючи підтвердження прийому раніше відправлених даних. Отже між відправленими і підтвердженими даними існує вікно (Window) вже відправлених, але ще не підтверджених даних. Кількість байт, які можна передавати без підтвердження, називається розміром вікна W. Як правило, розмір вікна встановлюється у стартових файлах мережевого програмного забезпечення. Оскільки TCP-канал є дуплексним, то підтвердження для даних, що йдуть в одному напрямку, можуть передаватися разом з даними, що йдуть у протилежному напрямку. Приймачі на одному і другому боці віртуального каналу у правляють потоком для того, щоб не допускати переповнювання буферів.

Порт відправника (Source port) займає 2 байти та ідентифікує процес-відправник джерела.

Порт призначення (Destination port) займає 2 байти та ідентифікує процесодержувач.

Порядковий номер (Sequence number) займає 4 байти і вказує номер байта, який визначає зміщення сегмента стосовно потоку даних, що відправляється.

Номер квитанції про підтвердження (Acknowledgment number) займає 4 байти, містить збільшений на одиницю максимальний номер байта в отриманому сегменті.

Поле Зсув даних (Data offset) визначає розмір заголовка сегмента TCP в 32-бітових словах. Інколи це поле називається Довжина заголовка і позначається HLEN (Header Length). Мінімальний розмір заголовка становить 5 слів, а максимальний - 15, що становить відповідно 20 і 60 байт. Зсув рахується від початку заголовка сегмента TCP.

Три біти (Reserved) зарезервовано для майбутнього використання і повинні встановлюватися в нуль.

Прапорці (управляючі біти) займають 9 біт. Шість останніх прапорців містять службову інформацію про тип сегмента:

- URG (Urgent pointer) - термінове повідомлення;
- ACK (Acknowledgement) - квитанція на прийнятий сегмент;
- PSH (Push) - повідомляє приймальну сторону про те, що дані з прийомного буфера необхідно передати програмі, якій вони призначені, без очікування заповнення буфера;
- RST (Reset) - запит на відновлення з'єднання;
- SYN (Synchronize sequence numbers) - повідомлення, що

використовується для синхронізації номерів послідовності під час встановлення з'єднання;

- FIN (final) - вказує на завершення з'єднання - ознака досягнення

передаючою стороною останнього байта у потоці даних, що передається.

Прапорці CWR (Congestion Window Reduced) та ECE (Explicit Congestion Notification-EcAo, ECN-Echo) рекомендовані для управління перевантаженнями з метою підтримки гарантованої якості обслуговування QoS. Прапорець NS запропоновано (RFC 3540) для необов'язкового використання з метою забезпечення стійкості роботи системи контролю перевантажень (насиченості).

Поле Вікно (Window Size) займає 2 байти, містить значення розміру вікна, що оголошується, у байтах.

Поле контрольної суми (Checksum) займає 2 байти і слугує для контролю помилок у заголовку і даних сегмента. Якщо сегмент містить непарне число октетів, останні октети доповнюються справа 8 нулями для вирівнювання по 16-бітовій межі. Біти заповнення (0) не передаються у сегменті і слугують тільки для розрахунку контрольної суми. Під час розрахунку контрольної суми значення самого поля контрольної суми приймається таким, що дорівнює 0.

Вказівник терміновості (Urgent pointer) займає 2 байти і використовується для визначення порядкового номера октета, з якого починаються важливі (urgent) дані. Це поле береться до уваги тільки для пакетів із встановленим прапорцем URG.

Поле Опції (Options) має змінну довжину і може взагалі бути відсугнім.

Використовується, наприклад, під час вибору максимального розміру сегмента.

Заповнювач (Padding) - фіктивне поле змінної довжини, що використовується для доведення розміру заголовка до цілого числа 32-бітових слів.

Протокол UDP

Протокол UDP набагато простіший, ніж TCP. Він корисний у ситуаціях, коли механізми забезпечення надійності протоколу TCP не є обов'язковими.

Заголовок UDP (UDP Header) має усього чотири поля:

- поле порту джерела (Source Port);
- поле порту пункту призначення (Destination Port);
- поле довжини (Length);
- поле контрольної суми (Checksum).

Поля порту джерела і порту призначення виконують ті самі функції, що і відповідні поля в заголовку сегмента TCP. Поле довжини визначає довжину заголовка і даних, поле контрольної суми забезпечує перевірку заголовка і даних на наявність помилок. Контрольна сума у протоколі UDP є факультативною можливістю.

Захоплення пакетів.

Для початку захоплення пакетів необхідно задати параметри, зокрема, вказати мережний інтерфейс, з якого і буде відбуватися захват пакетів. Ця дія доступна через меню "Capture -> Interfaces..." (рис 4.4)., для більш детального налаштування - "Capture -> Options..." (рис 4.5). У полі «Interface:» вибирається доступний інтерфейс (наприклад мережний адаптер).

У якості додаткових параметрів захвату можна вказати:

«Capture Filter» – фільтр захвату. Тут можна добавляти той чи інший фільтр із існуючих, або вказати умови фільтрації.

«Update list of packets in real time» – оновлення списку захоплених пакетів в режимі реального часу.

«Stop Capture» – набір параметрів, які дозволяють задати те чи інше значення, при досягненні якого процес захвату закінчиться.

«Name Resolution» – набір параметрів вирішення імен дозволяє визначити які із способів вирішення імен повинні використовуватися.

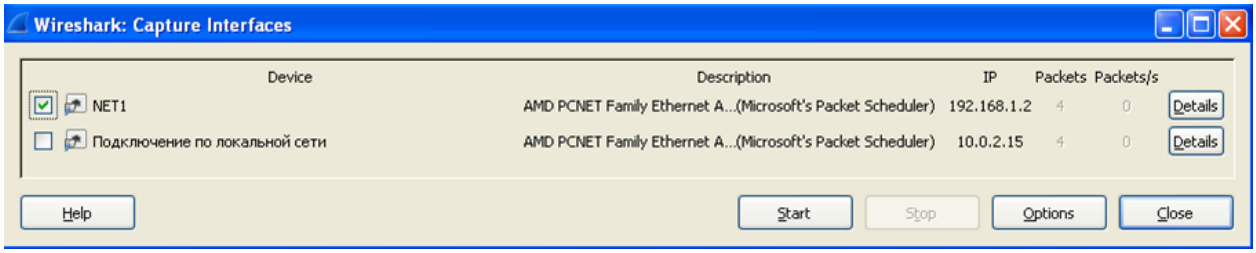

Рис 4.4

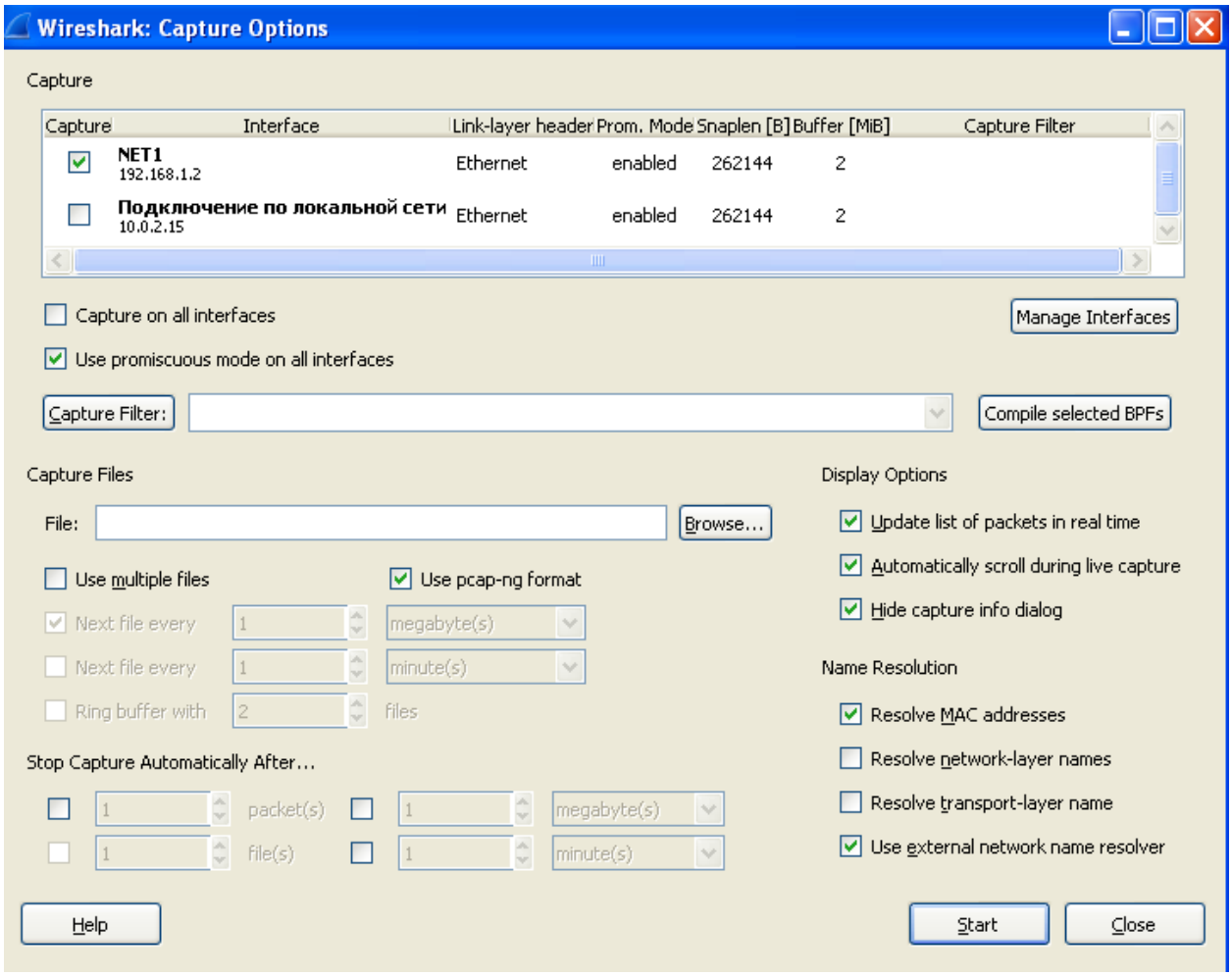

Рис 4.5. Вибір інтерфейсів та параметрів захвату пакетів.

Для початку моніторинга мережної активності необхідно натиснути «Start». Після вибору інтерфейсу, який цікавить, можна починати і закінчувати захват пакетів через команди в меню «Capture».

Налаштування фільтрів.

Якщо запустити сніфер без додаткових налаштувань, то він буде захоплювати всі пакети, які проходять через мережний інтерфейс. Найбільш часто користувача цікавлять тільки деякі пакети, які відповідають заданим критеріям. Для визначення цих критеріїв служать фільтри відображення та захоплення.

Існує два варіанти фільтрації пакетів: на етапі захоплення і на етапі відображення користувачу. У першому випадку ефективність роботи сніферу та споживані ним системні ресурси значно нижче, ніж у 2-му випадку.

При визначенні фільтру вказуються умови, яким повинні відповідати ті пакети, які повинні бути пропущені через фільтр. Назва полів та протоколів в фільтрі вводиться тільки маленькими літерами. Якщо текст фільтру має коректний зміст, поле фільтру підсвічується зеленим, якщо некоректний – червоним. Для задіяння фільтру необхідно його підтвердити кнопкою "Apply".

Фільтрацію можна проводити за протоколами або за значеннями полів в PDU (пакеті). Для застосування фільтрації за протоколом необхідно в поле вводу фільтра ввести назву протоколу. Наприклад фільтр за протоколом "arp" виводить тільки ARP-кадри (рис 4.6).

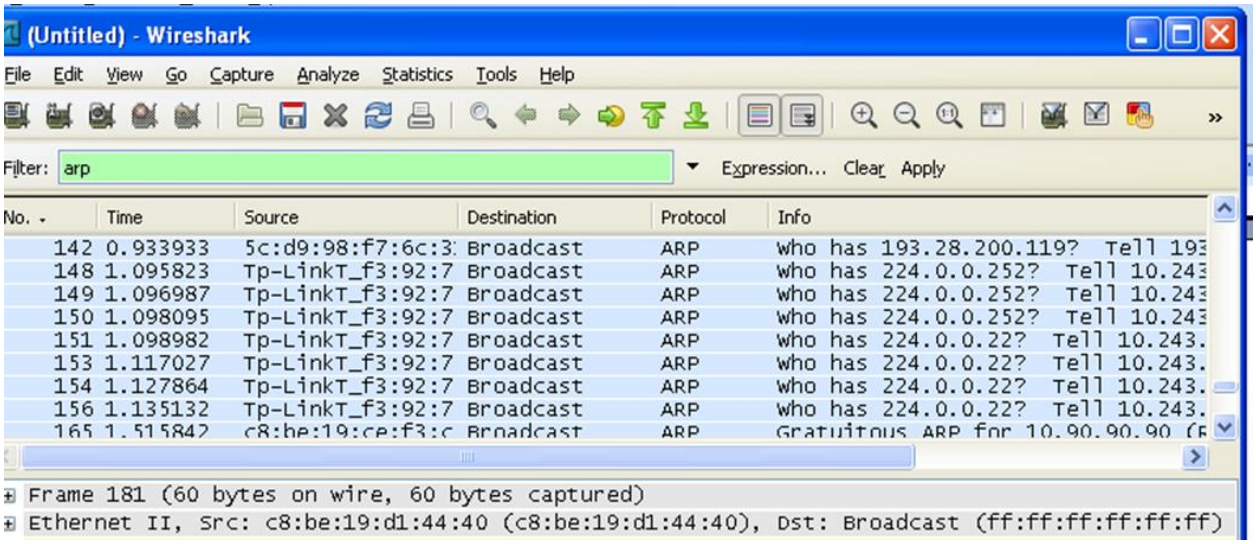

Рис 4.6. Приклад задавання фільтру через назву протоколу.

Для фільтрації за значенням поля в PDU, записується назва поля, оператор порівняння та значення з яким порівнюється це поле. Для формування тексту фільтру можна скористатися майстром побудови виразів фільтру, який викликається через кнопку "Expression...". Наприклад результат роботи майстра з рис 4.7 буде текстове значення фільтру

modbus_tcp.func_code == 3

У таблиці нижче наводиться перелік найбільш вживаних для лабораторних робіт полів.

Поле	Опис
eth.addr	Фізична адреса джерела або отримувача в кадрі Ethernet
eth.dst	Фізична адреса отримувача в кадрі Ethernet
eth.src	Фізична адреса джерела в кадрі Ethernet
eth.len	Довжина кадру Ethernet
ip.addr	Мережна адреса джерела або отримувача в пакеті протоколу IP
ip.dst	Мережна адреса отримувача в пакеті протоколу IP
ip.src	Мережна адреса джерела в пакеті протоколу IP

ip.proto	Позначення протоколу, який був інкапсульований в пакет IP
tcp.ack	Підтвердження (ACK) протоколу TCP
tcp.port	Порт джерела і отримувача в сегменті протоколу TCP
tcp.dstport	Порт отримувача в сегменті протоколу TCP
tcp.srcport	Порт джерела в сегменті протоколу TCP
udp.port	Порт джерела або отримувача в сегменті протоколу UDP
udp.dstport	Порт отримувача в сегменті протоколу UDP
udp.srcport	Порт джерела в сегменті протоколу UDP
dns.qry.name	Ім’я мережного ресурсу в DNS запиті
dns.resp.name	Ім’я мережного ресурсу в DNS відповіді

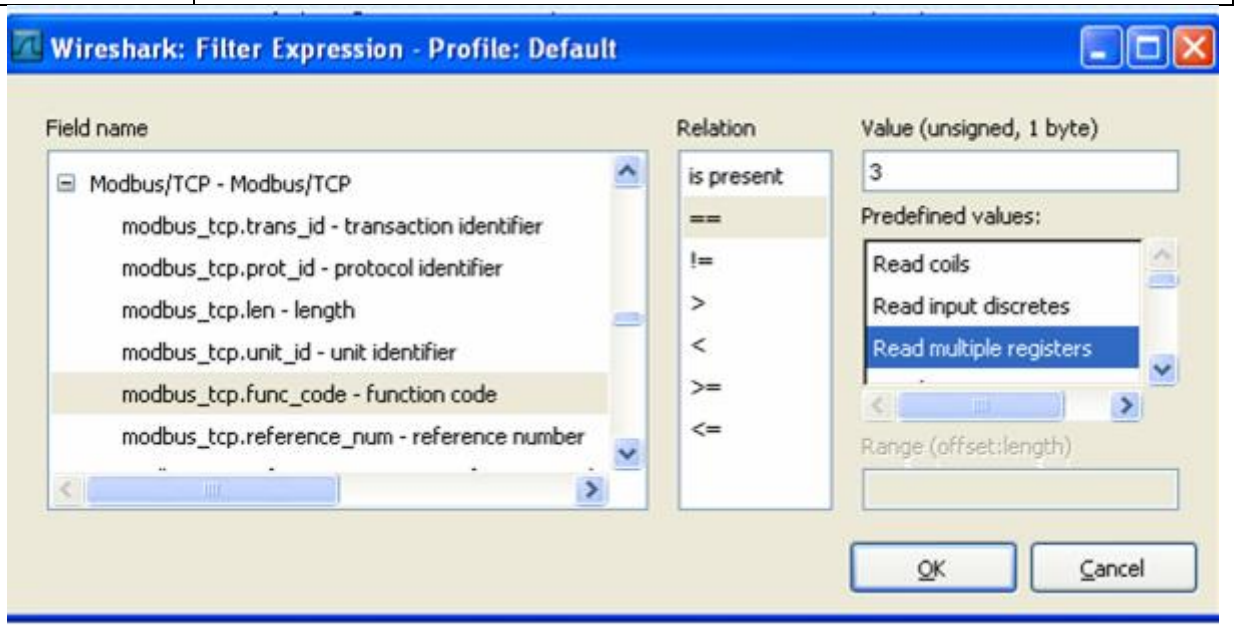

Рис 4.7. Приклад задавання фільтру через поле протоколу.

Оператор порівняння може являти собою С-подібний оператор або символьне скорочення оператору, які наведені в таб.1

символьне скорочення	C-like	Опис та приклад
Eq	==	Рівне ip.src==10.0.0.5

Ne	!=	Нерівне ip.src!=10.0.0.5
gt	>	Більше ніж frame.len > 10
lt	<	Менше ніж frame.len < 128
ge	>=	Більше або дорівнює frame.len >= 0x100
le	<=	Менше або дорівнює frame.len <= 0x20

Значення поля типу без-знакового цілого можна вводити в різних системах числення: 10-кова, 8-кова (починається з 0), 16-кова (починається з 0x). Наприклад, наступні значення еквівалентні.

ip.len le 1500

ip.len le 02734

ip.len le 0x436

Для того щоб задати умову для булевого поля, достатньо вказати це поле. Наприклад, наступне значення фільтру пропустить тільки ті пакети, для яких прапорець SYN в сегменті TCP буде виставлений

tcp.flags.syn

Байти або два байти адреси Ethernet можуть бути розділені розділювачами ":", "-" або ".". Наприклад:

eth.dst == ff:ff:ff:ff:ff:ff

eth.dst == ff-ff-ff-ff-ff-ff

eth.dst == ffff.ffff.ffff

Для визначення в умові текстового поля, його значення береться в лапки. Наприклад:

http.request.uri == "http://www.wireshark.org/"

При побудові фільтру можна комбінувати дві та більше умови, використовуючи логічні оператори. У якості умови може виконуватися як фільтрація за протоколами, так і фільтрація за значенням полів.

символьне скорочення	C-like	Опис та приклад
not	!	заперечення not llc
and	&&	конкатенація (логічне «І») ip.src==10.0.0.5 and tcp.flags.fin
or	||	альтернатива (логічне «АБО») ip.scr==10.0.0.5 or ip.src==192.1.1.1

Порядок виконання роботи

1. Ознайомитись з теоретичними матеріалами за тематикою практичної роботи.
2. Запустити програму WireShark, ознайомитись з її інтерфейсом, налаштувати параметри.
3. Виконати завдання.
4. Оформити звіт з практичної роботи.

Завдання

Завдання 1. Аналіз заголовків протокольних блоків даних канального, мережевого та транспортного рівнів.

1. Запустити перехоплення пакетів.

2. Перехопити кадр, у якому міститься пакет протоколу IPv4, що переносить блок даних протоколу TCP.

3. Розкрити заголовок кадру Ethernet та записати значення його полів у шістнадцятковій системі. Пояснити зміст кожного параметра.

4. Розкрити заголовок IP-пакета. У табл. 4.1 занести значення тих полів пакета протоколу IPv4, після яких стоїть знак “=”. Записи зробити у десятковій системі. Пояснити суть записаних параметрів.

Таблиця 4.1. Значення заголовка пакета протоколу IPv4

0								1								2								3							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Version=				HLEN=				Differentiated Services (Type of Service)								Total Length=															
Identification=																0	D F	M F	Fragment Offset=											
Time to Live=								Protocol=								Header Checksum															
Source Address=																															
Destination Address=																															
Options																															

5. Записати значення прапорців DF, MF та пояснити їх призначення.

Розкрити заголовок сегмента протоколу TCP. У табл. 4.2 занести значення тих полів заголовка, після яких стоїть знак “=”. Записи зробити у десятковій системі. Пояснити суть записаних параметрів.

Таблиця 4.2. Значення заголовка сегмента протоколу TCP

0								1								2								3							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Source port=																Destination port=															
Sequence number=																															
Acknowledgment number=																															
HLEN=				Reserved 000				Flags								Window Size=															
Checksum																Urgent pointer =															

6. Занести значення прапорців поля Flags у табл. 4.3 та пояснити, що

означає кожна опція.

Таблиця 4.3. Значення прапорців поля Flags

URG:	
ACK:	

PSH:	
RST:	
SYN:	
FIN:	

7. Перехопити пакет протоколу IPv4, що переносить блок даних протоколу UDP. Розкрити заголовок блока даних протоколу UDP. Занести значення полів у табл. 4.4 та пояснити, що означає кожне поле.

Таблиця 4.4. Значення заголовка сегмента протоколу UDP

UDP Header																															
0								1								2								3							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Source port=																Destination port=															
Length=																Checksum															

Проаналізувати перехоплений кадр, порядковий номер якого дорівнює вашому порядковому номеру у журналі викладача. Результати аналізу зафіксувати у звіті.

Завдання 2. Аналіз протокольних блоків даних протоколу ARP.

1. Запустити програму Wireshark. Переконатись, що усі опції налаштовані правильно та увімкнути процес перехоплення пакетів.

2. Використати функцію фільтрації та захопити пакети, що переносять блоки даних протоколу ARP.

3. Розкрити заголовок ARP-запиту та занести значення полів у табл.

4.5. Пояснити, що означає кожне поле.

4. Розкрити заголовок ARP відповіді та занести значення полів у табл.

4.5. Які зміни відбулись у значеннях полів ARP повідомлення? Чому?

Таблиця 4.5. Значення ARP повідомлень

0	8	16	24
Hardware type		Protocol type	
Hardware size	Protocol length	Opcode	
Sender hardware address (SHA) (байти 0-3)			
SHA (байти 4-5)		Sender protocol address (SPA) (байти 0-1)	
SPA (байти 2-3)		Target hardware address (THA) (байти 01)	
THA (байти 2-5)			
Target protocol address (TPA) (байти 0-3)			

Завдання 3. Перехоплення ICMP-повідомлень.

1. Запустити програму \Vireshark. Переконатись, що усі опції налаштовані правильно та увімкнути процес перехоплення пакетів.

2. Із командного рядка виконати команду ping <IP-адреса>, або ping <символьне ім'я>. Після виконання утиліти ping, перейти до вікна Wireshark та припинити перехоплення пакетів.

3. Відфільтрувати кадри, що містять ICMP-повідомлення. У разі успішного виконання попереднього етапу знайти ехо-запити та ехо-відповіді.

4. Обрати перший ехо-пакет. Для отримання детальнішої інформації на панелі Packet Detail розгорнути заголовки блоків даних усіх протоколів. Проаналізувати цю інформацію. Перелічити усі протоколи, блоки даних яких містяться у кадрі Ethernet.

5. Знайти усі поля “Source” та “’Destination”. Які два типи адрес використовуються як адреси відправника та одержувача?

6. Перевірити чи є в перехоплених пакетах інші ICMP-повідомлення.

Проаналізувати їх та визначити причину їх появи.

Зміст звіту

1. Теоретичні відомості та скриншоти виконанння завдань.
2. Виконані завдання та заповнені таблиці згідно завдань.
3. Зробити висновки.