Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Навчально-науковий центр заочної форми навчання
(повна назва)

Кафедра *Інформаційно-мережної інженерії*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Управління трафіком у мережах передачі даних на основі протоколу TCP

(тема)

Виконав:
здобувач *4* року навчання,
групи *ТРІМІз-21-1*

Андрій ЧУРСІН
(власне ім'я, прізвище)

Спеціальність *172 Телекомунікації та радіотехніка*
(код і повна назва спеціальності)

Тип програми *освітньо-професійна*

Освітня програма
«Інформаційно-мережна інженерія»
(повна назва освітньої програми)

Керівник доцент *Юрій КОЛТУН*
(посада, власне ім'я, прізвище)

Допускається до захисту

Зав. кафедри _____ *Валерій БЕЗРУК*
(підпис) (власне ім'я, прізвище)

2025 р.

Не містить відомостей заборонених до відкритого публікування.

Здобувач / *Андрій ЧУРСІН* /

Керівник / *Юрій КОЛТУН* /

Харківський національний університет радіоелектроніки

Навчально-науковий центр заочної форми навчання

Кафедра *Інформаційно-мережної інженерії*
(повна назва)

Рівень вищої освіти *перший (бакалаврський)*

Спеціальність *172 Телекомунікації та радіотехніка*
(код і повна назва)

Тип програми *освітньо-професійна*
(освітньо-професійна або освітньо-наукова)

Освітня програма *«Інформаційно-мережна інженерія»*
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«*02*» *травня* 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві *Чурсіну Андрію Віталійовичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Управління трафіком у мережах передачі даних на основі протоколу TCP*
затверджена наказом університету від «*02*» *травня* 2025 р. № 63 *Стз*
2. Термін подання студентом роботи до екзаменаційної комісії *20 червня 2025 р.*
3. Вихідні дані до роботи *Тип мережі ПД – IP-мережі. Метод управління трафіком – передача/запит підтвердження на основі ковзного вікна. Схема роботи методу – запит повторної передачі.*
Розглянути загальні особливості функціонування та формат заголовку TCP. Проаналізувати механізми управління трафіком, що підтримуються TCP за заданим методом. Надати загальну методику оцінки часових характеристик щодо заданого методу у процесі обміну TCP-сегментами.

4. Перелік питань, що потрібно опрацювати в роботі
Вступ
1. Загальні особливості та принципи роботи протоколу TCP.
2. Аналіз механізмів управління потоками трафіку, що реалізуються протоколом TCP у процесі передачі даних.
3. Загальна методика оцінки часових характеристик у процесі передачі підтверджень на доставку TCP-сегментів.
Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) *Слайди у форматі Power Point (назва, мета і актуальність роботи, формування TCP-сегментів із потоку байтів, структура заголовку TCP-сегменту, схема утворення надійного логічного каналу між кінцевими вузлами шляхом встановлення TCP-з'єднання, діаграми встановлення і розриву логічного з'єднання, діаграма роботи механізму простою джерела, загальна концепція механізму ковзного вікна, механізм віконного управління передачею з поверненням на N пакетів, механізм віконного управління передачею і прийомом пакетів у разі вибіркового повторення, схема функціонування управління потоком із виділенням ліміту на передачу даних, висновки)*

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	*Ознайомлення із завданням. Уточнення ТЗ.*	02.05 – 03.05.25	*виконано*
2	*Підбір літератури за темою роботи.*	04.05 – 10.05.25	*виконано*
3	*Виконання розділу 1*	11.05 – 20.05.25	*виконано*
4	*Виконання розділу 2*	21.05 – 29.05.25	*виконано*
5	*Виконання розділу 3*	30.05 – 11.06.25	*виконано*
7	*Оформлення пояснювальної записки*	12.06 – 20.06.25	*виконано*
8	*Оформлення презентаційного матеріалу,*		
	підготовка до захисту у ЕК, захист.	21.06 – 26.06.25	*виконано*

Дата видачі завдання *02 травня 2025 р.*

Здобувач _____
(підпис)

Керівник роботи _____ *(доц. Юрій КОЛТУН)*
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 55 с., 14 рис., – табл., 12 джерел, 1 додаток.

TCP, TCP-СЕГМЕНТ, ПАКЕТ, ПІДТВЕРДЖЕННЯ, КВИТАНЦІЯ, КВИТУВАННЯ, УПРАВЛІННЯ ТРАФІКОМ, ПЕРЕДАЧА З ПІДТВЕРДЖЕННЯМ, ЗАПИТ ПОВТОРНОЇ ПЕРЕДАЧІ, ARQ, МЕТОД КОВЗНОГО ВІКНА, МЕТОД ПРОСТОЮ ДЖЕРЕЛА, ТАЙМЕР, ТАЙМ-АУТ

Об'єкти дослідження – протокол TCP, механізми управління трафіком.

Мета роботи – загальний аналіз підходів і механізмів управління трафіком у мережах передачі даних на основі протоколу TCP.

Зроблено аналіз механізмів управління потоками трафіку, що підтримуються протоколом TCP у процесі передачі даних. За напрям проведення аналізу взятий метод передачі з підтвердженням на основі алгоритму ковзного вікна, робота якого направлена на запит повторної передачі у разі виявлення втрати або помилки при передачі TCP-сегменту. Запропонована загальна методика проведення оцінки середнього часу затримки отримання підтверджень для певної кількості сегментів TCP, що були передані, і визначення тайм-ауту повторної передачі.

THE ABSTRACT

Explanatory note 55 pages, 14 fig., – tab., 12 sources, 1 app.

TCP, TCP SEGMENT, PACKET, ACKNOWLEDGMENT, RECEIPT, ACKNOWLEDGMENT, TRAFFIC MANAGEMENT, ACKNOWLEDGMENT-BASED TRANSMISSION, AUTOMATIC REPEAT REQUEST, ARQ, SLIDING WINDOW METHOD, SIMPLE SOURCE METHOD, TIMER, TIME-OUT

Objects of research – TCP protocol, traffic management mechanisms.

The purpose of work – general analysis approaches and mechanisms for managing traffic in data transmission networks based on the TCP protocol.

An analysis of traffic flow control mechanisms supported by the TCP protocol during data transmission has been performed. The analysis was based on the acknowledgment-based transmission method using a sliding window algorithm, which is designed to request retransmission in case of loss or error during TCP segment transmission. A general methodology is proposed for estimating the average delay time for receiving confirmations for a certain number of TCP segments that have been transmitted and for determining the retransmission time-out.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ARQ (Automatic Repeat reQuest) – запит повторної передачі;

MSS (Maximum Segment Size) – максимальний розмір сегменту;

QoS (Quality of Service) – якість обслуговування;

RTO (Retransmission Time Out) – тайм-аут повторної передачі;

RTT (Round Trip Time) – час проходження туди і назад (час обігу);

SN (Sequence Number) – номер у послідовності;

SRTT (Smoothed Round Trip Time) – усереднена оцінка часу звернення;

TCP/IP (Transmission Control Protocol / Internet Protocol) – протокол управління передачею / протокол передачі в мережі Internet.

МПД – мережі передачі даних;

ОС – операційна система;

ПД – передача даних;

ПЗ – програмне забезпечення;

ТКС – телекомунікаційна система.

ВСТУП

Сучасні мережі передачі даних, що орієнтовані на пакетну комутацію, є складними телекомунікаційними системами (ТКС), які складаються з великої кількості різноманітних компонентів: комп'ютерів, комутаторів, маршрутизаторів, системного прикладного програмного забезпечення (ПЗ), тощо. Основна задача мережних інженерів і системних адміністраторів у таких ТКС полягає в тому, щоб вони максимально ефективно здійснювали обробку потоків інформації та забезпечували можливість користувачам виконувати їх прикладні задачі, на основі надання їм своїх ресурсів. Прикладне ПЗ при цьому звертається до служб, що забезпечують зв'язок з іншими прикладними програмами в мережі через відповідні мережні вузли, тобто реалізуються різні принципи і механізми транспортування інформаційних потоків за допомогою міжмережного обміну.

Особливу вагомість ці принципи і механізми набули в процесі розвитку мережі Internet, яка змінила спосіб подання інформації, об'єднавши можливості транспортування всіх її видів – відео, голосу, графічних зображень і звичайних даних, що дало змогу створити на основі їх комбінацій безліч різноманітних інформаційних послуг. Одними з найважливіших логічних компонентів мережі Internet, як і будь-якої іншої мережі передачі даних (МПД), під час організації міжмережного обміну в процесі передачі даних (ПД) є телекомунікаційні протоколи, які здійснюють управління процесами інформаційного обміну. Основою Internet є стек протоколів TCP/IP (Transmission Control Protocol / Internet Protocol – протокол управління передачею/протокол передачі в мережі Internet), основна задача якого полягає в реалізації міжмережного обміну. Зокрема стек TCP/IP забезпечує надійний зв'язок між мережним обладнанням від різних виробників, а протоколи, що до нього входять, надають механізми передачі повідомлень, описують їх формат і вказують, як здійснювати обробку помилок. Ці протоколи дають змогу описати та зрозуміти процеси ПД незалежно від типу обладнання, на якому ці процеси відбуваються [1].

Слід зазначити, що функціонал щодо забезпечення надійної передачі потоків інформації між кінцевими мережними пристроями покладається на телекомунікаційні протоколи транспортного рівня. Тому для досягнення високої продуктивності всіх вузлів у мережі та кінцевого обладнання користувачів дуже важливо визначитися з конкретним транспортним протоколом, який буде

забезпечувати інтерфейс між додатками та мережним обладнанням. До його функціоналу також має входити запит необхідної якості обслуговування (Quality of Service, QoS), потрібної для додатків, що взаємодіють [1].

Основним транспортним протоколом стека TCP/IP є однойменний протокол управління передачею (TCP), який також є транспортною основою для більшості застосунків і сервісів в мережі Internet. Він використовується для встановлення з'єднань і управління передачею даних між додатками. TCP гарантує, що дані будуть доставлені в правильному порядку і без втрат. Це досягається завдяки використанню механізму підтверджень і повторної передачі втрачених пакетів. Він також забезпечує контроль потоку і управління перевантаженнями, що дає змогу ефективно використовувати мережні ресурси. Завдяки цим властивостям, TCP є кращим вибором для тих додатків, що вимагають високої надійності та точності ПД [2].

Метою цієї кваліфікаційної роботи є аналіз підходів і механізмів управління трафіком у мережах ПД на основі протоколу TCP. Механізми, що реалізовані у протоколі TCP для управління потоками трафіку, дають змогу динамічно адаптувати передачу даних до поточних умов мережі, забезпечуючи надійність та ефективність міжмережного обміну та запобігти виникненню перевантажень. Тому, в умовах постійного зростання мережних навантажень, зростанням запитів на доступ до мережних ресурсів і ускладненням мережної інфраструктури, тема щодо управління трафіком на основі TCP зберігає свою високу актуальність [3].

1 ЗАГАЛЬНІ ОСОБЛИВОСТІ ТА ПРИНЦИПИ РОБОТИ ПРОТОКОЛУ TCP

1.1 Призначення, загальні особливості роботи і структура заголовку протоколу TCP

Протокол TCP призначений для здійснення передачі даних між додатками. Він має засоби управління потоком і корекції помилок і є орієнтованим на встановлення логічного з'єднання, що дає йому змогу забезпечувати надійність і достовірність обміну даними між процесами на комп'ютерах, які входять у загальну мережу. Протокол TCP, з одного боку, взаємодіє з прикладним протоколом додатку користувача, а з іншого – з протоколом, що забезпечує «низькорівневі» функції: маршрутизацію та адресацію пакетів, використовуючи в якості інструменту ненадійний дейтаграмний сервіс IP-протоколу [4, 5].

В операційній системі (ОС) відтворення протоколу TCP являє собою окремий модуль драйверу, що відноситься до системних, через який, як правило, просуваються усі виклики його функцій. Інтерфейс між прикладним процесом і протоколом TCP – це по суті бібліотека викликів, яка є подібною, наприклад, до бібліотеки системних викликів, що забезпечує роботу з файлами. Зокрема, можна виконати операцію відкриття з'єднання і передати дані або операцію закриття з'єднання і прийняти дані із встановленого з'єднання (тобто точно також, як і у разі роботи з файлом: можна виконати операції відкриття або закриття файлу і здійснити читання або запис файлу). Виклики протоколу TCP можуть функціонувати з прикладним додатком в асинхронному режимі. Треба звернути увагу, що використання TCP у тій або іншій системі може застосовувати безліч власних функцій, але будь-яке втілення TCP у конкретній системі має надавати потрібну за стандартами його функціональність [5].

У разі роботи на вузлі-відправнику протокол TCP сприймає інформацію, що пересилається до нього від прикладних процесів, як неструктурований потік байтів (рис. 1.1). Буферизація даних, що надходять, здійснюється засобами TCP. Для здійснення передачі на мережний рівень із буфера виокремлюється деяка безперервна їх частина, яка отримала назву сегменту. Сегмент – це по факту інформаційна одиниця TCP, яка має свій формат і, як і всі форматні одиниці даних, забезпечується заголовком [6].

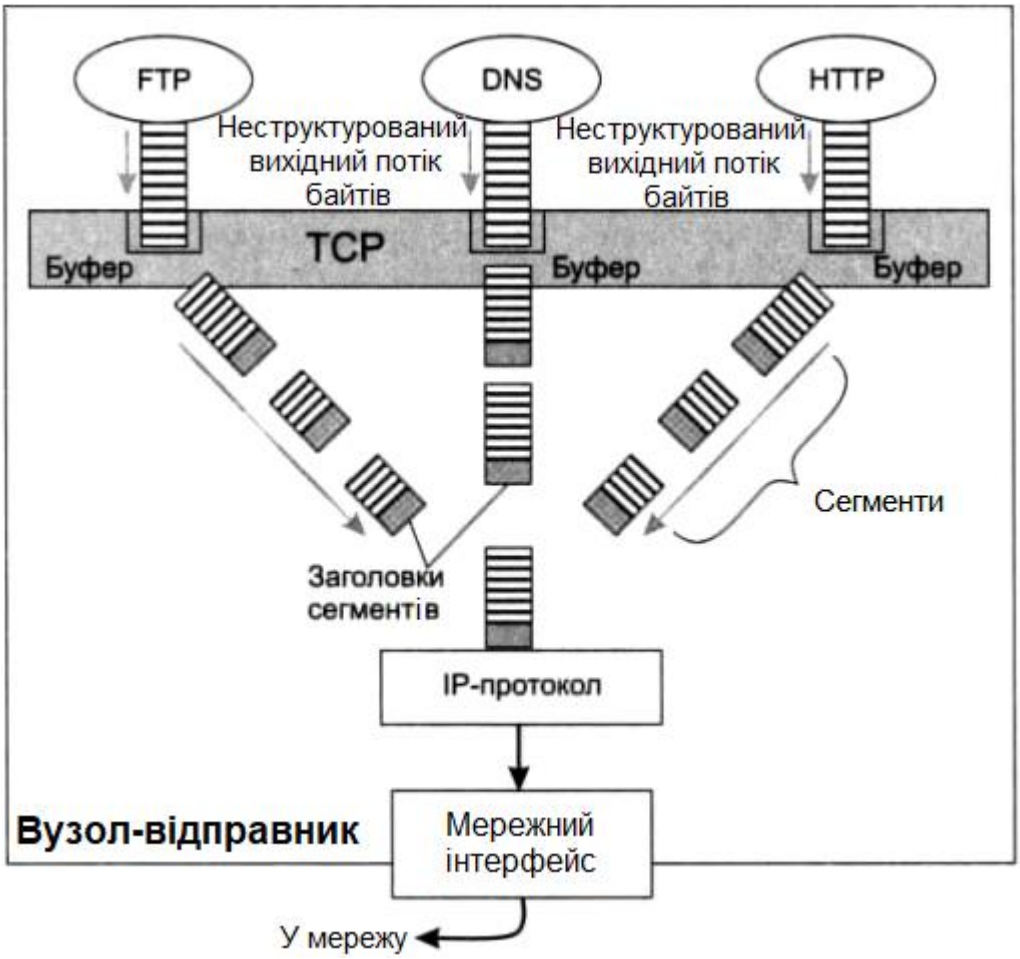

Рисунок 1.1 – Формування TCP-сегментів із потоку байтів

Узагальнено робота додатка користувача у процесі здійснення передачі даних за протоколом TCP реалізується наступним чином. Для реалізації ПД процесу користувача треба зробити виклик відповідної функції TCP та вказати на буфер даних, що передаються. TCP уміщує ці дані в сегменти свого стека і далі викликає функцію передачі протоколу нижнього рівня (як правило, це протокол IP). На приймальній стороні одержувач TCP групує дані, що надійшли від протоколу нижнього рівня, у приймальні сегменти свого буфера, перевіряє їх цілісність і передає відповідному процесу користувача, після чого інформує відправника про їх отримання [5].

Слід зазначити, що сегментом називають як одиницю переданих даних загалом (поле даних і заголовок протоколу TCP), так і окремо поле даних. Крім того, на відміну від протоколу дейтаграм користувача (User Datagram Protocol, UDP), що створює дейтаграми на основі логічно відокремлених одиниць даних

(повідомлень, що генеруються додатками), протокол TCP ділить потік даних на сегменти без урахування їх змісту або внутрішньої структури [6].

Структура заголовка TCP-сегмента показана на рис. 1.2 [4, 6].

Поля «Порт джерела» (Source Port) і «Порт отримувача» (Destination Port) мають довжину по 16 біт кожне та ідентифікують процес або додаток, що використовує протокол TCP [4, 6].

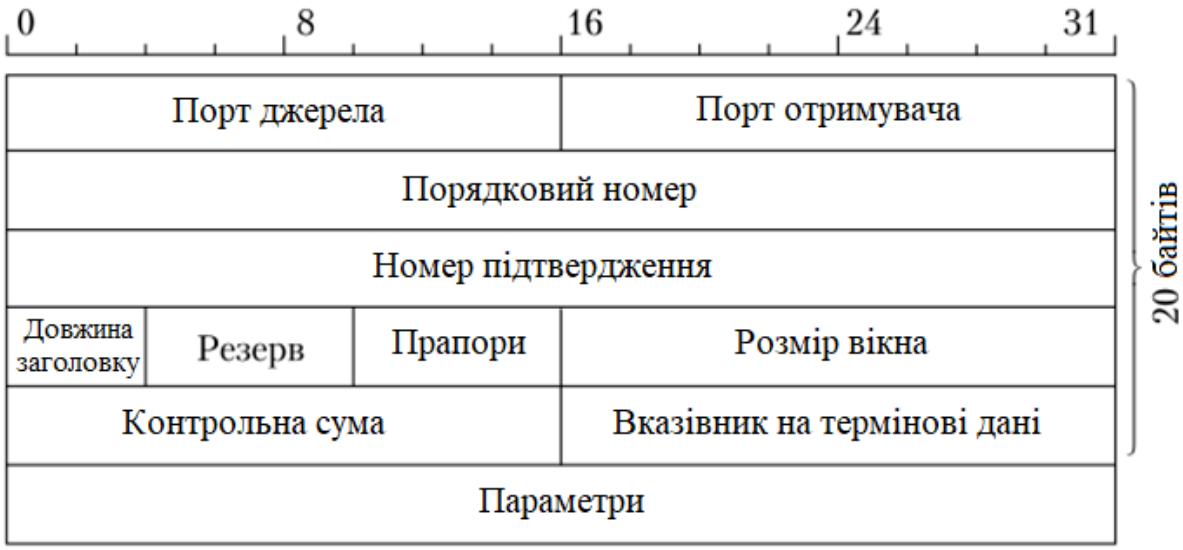

Рисунок 1.2 – Структура заголовку TCP-сегменту

Поле «Порядковий номер» (Sequence Number) довжиною 32 біти визначає номер першого байту даних у сегменті, а також зсув сегмента відносно потоку даних, що відправляються. Відповідно поле «Номер підтвердження» (Acknowledgement Number) також довжиною 32 біти вказує максимальний номер байту в отриманому сегменті, що збільшений на одиницю. Ці поля реалізуються як цілі числа без знаку, які скидаються, коли досягають максимального значення. Кожна сторона веде власну порядкову нумерацію [4, 6].

Поле «Довжина заголовка» (Offset) має довжину 4 біти і містить розмір TCP-заголовка в 32-бітних словах. Ця інформація необхідна, тому що поле «Параметри» (Options) може бути змінної довжини. Можна сказати, що це поле задає зміщення від початку сегмента до початку даних у 32-бітних словах [4].

Далі йдуть поле «Резерв» (Resrved), що не використовується, і поле «Прапори» довжиною по 6 біт кожне. Однобітні значення прапорів або ще їх називають «кодовими бітами» показані на рис. 1.3 [4, 6].

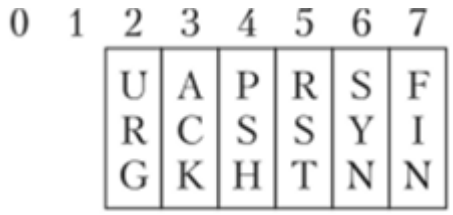

Рисунок 1.3 – Значення прапорів у заголовку протоколу TCP

Поля прапорів містять службову інформацію про тип даного сегмента. Позитивне значення вказує на встановлення цих бітів у «1» [4, 6]:

- прапор URG (Urgent Pointer – вказівник терміновості) встановлюється в «1» у випадку використання поля «вказівник па термінові дані», в разі передачі термінового повідомлення;

- прапор АСК (Acknowledgment – підтвердження) встановлюється в «1», якщо поле «Номер підтвердження» (Acknowledgement Number) містить дані, а в протилежному випадку це поле буде проігнороване;

- прапор PSH (Push) використовується у випадку запиту на відправлення повідомлення або необхідності негайно інформувати додаток про дані, що надійшли, без очікування заповнення буфера. Зазначимо, що більшість сучасних релізів TCP просто ігнорують цей прапор у процесі здійснення прийому пакетів;

- прапор RST (Reset – запит на скидання з'єднання) використовується для скасування з'єднання через помилки додатка, відмови від неправильного сегменту, спроби створити з'єднання у разі відсутності затребуваного сервісу;

- прапор SYN (Synchronize – синхронізація) використовується як повідомлення для синхронізації лічильників переданих даних у разі встановлення з'єднання. Встановлюється в «1» за умови ініціювання з'єднання і синхронізації порядкового номера;

- прапор FIN (Finished – завершення) є ознакою досягнення стороною, яка передає, останнього байту в потоці даних, що передаються. Використовується для розриву з'єднання у випадку, коли відправник закінчив передачу даних.

Управління потоком у протоколі TCP здійснюється за допомогою ковзного вікна змінного розміру. Поле «Розмір вікна» (Window) має довжину 16 біт і містить кількість байт, яка може бути надіслана після байту, отримання якого вже було підтверджено. Якщо значення цього поля становить «0», це означає, що всі байти, аж до байту, номер якого зазначено в полі «Номер підтвердження», були отримані, але одержувач надалі відмовляється приймати наступні дані. Дозвіл на

подальшу передачу видається відправленням сегменту з таким самим значенням поля «Номер підтвердження» і ненульовим значенням поля «Розмір вікна» [4] .

Поле «Контрольна сума» (CheckSum) призначене для забезпечення цілісності повідомлення. Воно також має довжину 16 біт і містить контрольну суму сегменту TCP, яка обчислюється по всьому сегменту за модулем 1. Під час обчислення контрольної суми це поле вважається рівним «0», а поле даних вирівнюється по 32-байтній межі нульовими байтами. Перед визначенням контрольної суми до TCP-сегменту додається псевдозаголовок (рис. 1.4). Він призначений для «страхування» неправильної маршрутизації TCP-сегменту і містить у собі IP-адреси відправника і одержувача (довжинами по 32 біти кожна). Далі йде зарезервоване поле довжиною 8 біт, що заповнене нулями. Поле «Протокол» довжиною 8 біт ідентифікує протокол із заголовка пакета IP (для TCP це значення дорівнює 6). Останнім у структурі псевдозаголовку є 16-ти бітне поле «Довжина TCP-сегменту» [4 - 6].

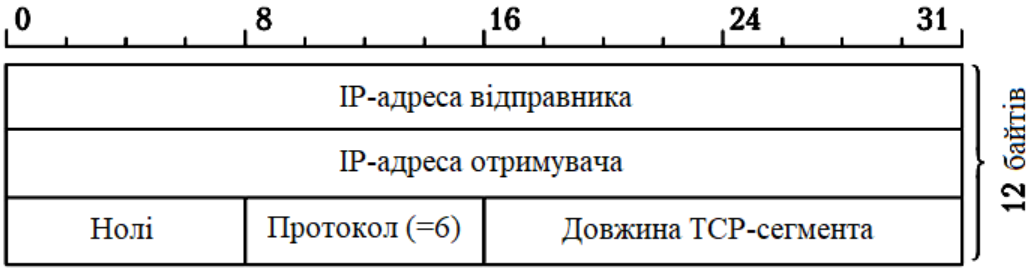

Рисунок 1.4 – Структура псевдозаголовку TCP-сегменту

Поле «Вказівник на термінові дані» має довжину 16 біт і являє собою вказівник останнього байту, що містить інформацію, яка потребує негайного реагування. Це поле береться до уваги у разі значення прапору URG=1, який позначає сегмент із першим байтом «термінової» інформації [7].

Поле «Параметри» або «Опції» (Option) має змінну довжину, що кратна 32-ом бітам. Це поле зарезервоване для майбутнього застосування і в заголовку може бути відсутнім. На цей час визначені наступні опції [4, 7]:

- кінець списку параметрів (опцій);

- жодних операцій – використовується для того, щоб заповнити поля «Параметри» («Опції») до числа октетів, що є кратним 4;

- максимальний розмір сегменту (Maximum Segment Size, MSS) – встановлює верхній розмір поля даних.

Зазначимо, що дані в TCP-сегменті можуть бути навіть відсутніми, а характер і формат інформації, що передається, задаються виключно прикладною програмою. Теоретично максимальний розмір поля «Дані», якщо воно є, становить за відсутності опцій 65 495 байт [5, 6].

Також зазначимо, що в TCP передбачена підтримка режиму повнодуплексної передачі, за яким потоки даних можуть незалежно передаватися в обох напрямках. У разі здійснення такого обміну кожна із сторін повинна відстежувати позиційні номери переданих і прийнятих байтів. Якщо був отриманий сегмент із деяким кодом поля «Номер підтвердження», у якому зазначено номер байту, що має прийти наступним, то це означає, що всі байти з номерами, що є меншими за зазначений у даному полі, були доставлені успішно. Якщо успішно були доставлені байти з номерами 0-*N*, а потім ще додатково був отриманий сегмент з номерами байтів *(N+k) – (N+k+m)*, то такий сегмент буде поміщений у буфер, але підтвердження на його одержання не буде надане. Замість цього буде надіслано відгук, з кодом номеру байту, який має прийти наступним, тобто *(N+1)*. У разі отримання сегменту з невірною контрольною сумою буде надіслано відгук, який буде ідентичний попередньому. Таке дублювання відгуків надає змогу визначати втрату пакета [7].

1.2 Особливості встановлення логічних з’єднань за протоколом TCP

Як було зазначено на початку попереднього розділу, однією з важливих задач протоколу TCP є забезпечення надійної доставки повідомлень, використовуючи для цього дейтаграмний протокол IP, який сам по собі не є надійним. Для виконання цієї задачі TCP протокол застосовує метод передачі даних із здійсненням встановлення логічного з'єднання. Таке з'єднання дозволяє всім користувачам, що приймають участь у обміні даними, стежити за тим, щоб з ними нічого не сталося (тобто, щоб вони не були втрачені, спотворені або продубльовані), а також, щоб дані поступили до вузла-одержувача у тій послідовності, в якій їх було надіслано [6].

Протокол TCP здійснює встановлення логічних з'єднань між різними прикладними процесами, але в кожному такому з'єднанні приймає участь тільки два процеси. Логічне з'єднання TCP є дуплексним, тобто кожен, хто бере в ньому участь, може одночасно отримувати та здійснювати передачу даних. На рис. 1.5

можна бачити мережі, що з'єднані між собою маршрутизаторами та які здійснюють обмін за протоколом IP. Модулі TCP протоколу розгорнуті на кінцевих вузлах. І саме вони вирішують задачу реалізації надійного обміну даними за рахунок організації логічних з'єднань [6].

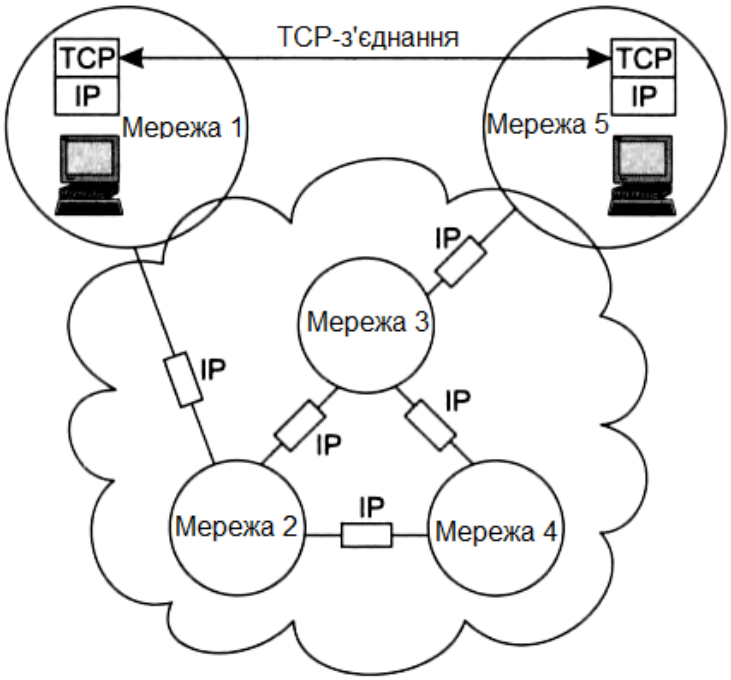

Рисунок 1.5 – Схема утворення надійного логічного каналу між кінцевими вузлами шляхом встановлення TCP-з’єднання

У разі організації такого логічного з'єднання модулі TCP узгоджують між собою різні параметри процедури здійснення обміну даними. Зокрема за протоколом TCP кожний учасник цього з'єднання посилає учаснику з протилежного боку наступні параметри [6]:

- максимальний розмір сегменту, який готовий цей учасник приймати;

- максимальний об'єм даних (це муже бути кілька сегментів), які учасники дозволяють один одному передавати, навіть якщо протилежна сторона ще не одержала підтвердження на попередній означений об’єм даних (так званий «розмір вікна»);

- початковий порядковий номер байту, з якого той або іншій учасник з’єднання починає відлік потоку даних у рамках цього з'єднання.

У результаті такої процедури узгодження модулі TCP передавальної та приймальної сторін з'єднання роблять визначення параметрів з'єднань. При цьому

одні з них будуть постійними протягом усього встановленого сеансу зв'язку, а інші будуть адаптивно змінюватися під поточну ситуацію [6].

У разі організації такого логічного з'єднання TCP формується зв'язок типу «клієнт-сервер». Воно організується за ініціативою клієнта (тобто клієнтської частини додатку). У разі потреби здійснити обмін даними із серверною частиною клієнт звертається до протоколу TCP, що знаходиться нижче за функціональним рівнем, та який у відповідь на цей звернення надсилає свій сегмент-запит на організацію встановлення з'єднання протоколу TCP, який функціонує на боці сервера (рис. 1.6а). У запиті міститься прапор синхронізації (SYN), що встановлений в «1» [6].

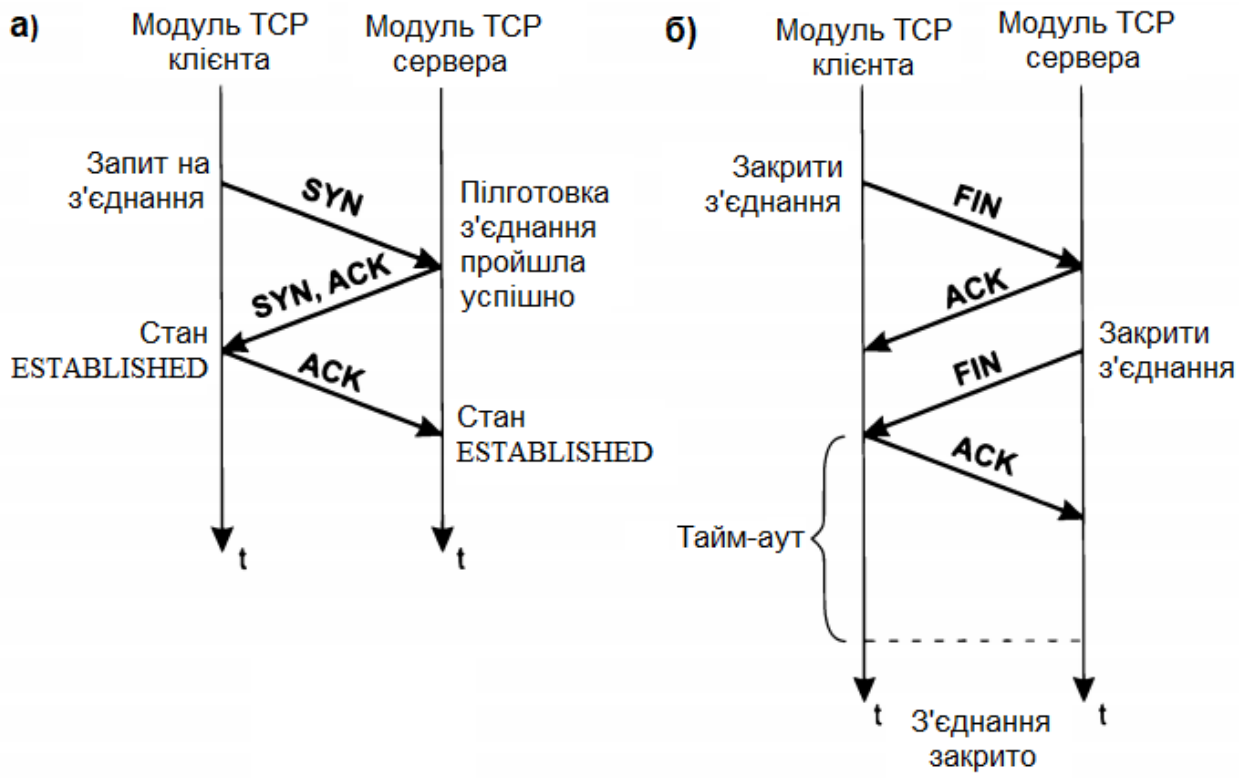

Рисунок 1.6 – Діаграми встановлення (а) і розриву (б) логічного з’єднання

Після того як сервер отримує вхідний запит на здійснення встановлення логічного з’єднання, TCP-модуль на стороні сервера ініціює процес підготовки до здійснення обслуговування нового клієнта. Для цього він формує запит до операційної системи з метою резервування та виділення необхідних системних ресурсів. Зокрема, створюються спеціальні буфери для зберігання вхідних і вихідних даних, налаштовуються таймери для контролю часу очікування

підтверджень і повторних передач, а також ініціалізуються лічильники для відстеження кількості переданих і прийнятих байтів. Усі ці ресурси закріплюються за цим з'єднанням – від моменту його створення до остаточного завершення. Тобто кожне TCP-з'єднання має свій власний набір системних ресурсів, який забезпечує його функціонування [6].

Якщо сервер успішно отримав всі необхідні ресурси та завершив виконання всіх дій щодо встановлення з'єднання, TCP-модуль надсилає клієнту спеціальний сегмент, що містить два прапори – SYN (synchronize) і ACK (acknowledgement). SYN сигналізує про ініціацію нового з'єднання, а ACK підтверджує отримання попереднього повідомлення від клієнта. У відповідь клієнт формує і надсилає серверу TCP-сегмент з прапором ACK, чим підтверджує прийом «SYN-ACK» від сервера. Після цього клієнт переходить у стан ESTABLISHED – встановленого логічного з'єднання. Отримавши від клієнта відповідь з прапором ACK, сервер також переходить у стан ESTABLISHED. На цьому процес встановлення TCP-з'єднання завершується і обидві сторони можуть починати здійснювати обмін даними [6].

Завершення TCP-з'єднання може бути ініційоване будь-якою із сторін, як клієнтом, так і сервером, так і в будь-який момент часу. Для цього ініціатор розриву надсилає сегмент із прапором FIN (finish), який сигналізує про намір здійснити припинення передачі даних. Інша сторона у відповідь надсилає сегмент з прапором ACK, тим самим підтверджуючи отримання повідомлення про завершення. Потім друга сторона також надсилає свій сегмент із прапором FIN, а ініціатор сегмент із прапором ACK. Ця послідовність обміну сегментами із прапорами FIN та ACK показана на рисунку 2.2б (тут ініціатором завершення з'єднання є клієнт). Вона гарантує правильне завершення з'єднання. Зокрема, якщо, як у нашому випадку, клієнт ініціює завершення, то він обов'язково має дочекатися підтвердження від сервера [6].

Після завершення обміну фінальними повідомленнями, логічне TCP-з'єднання вважається повністю закритим. Однак протягом деякого періоду (що відомий як TIME_WAIT («час очікування)) сторона, яка ініціювала завершення, зберігає контекст з'єднання, щоб переконатися в тому, що останній прапор ACK було отримано і, що не з'явились дубльовані або аварійні сегменти. Після витримки TIME_WAIT система остаточно вивільняє всі ресурси, які були задіяні для цього логічного з'єднання [6].

Ідентифікація логічних з'єднань TCP завжди здійснюється парою сокетів, що визначені саме для нього двома процесами, які взаємодіють. При цьому сокет може приймати участь у декількох з'єднаннях [6].

Зауважимо, що на рис. 2.2 процеси організації та завершення з'єднання показані досить схематично. Реальні модулі TCP протоколу функціонують за більш складними алгоритмами, в яких враховуються різні ситуації, зокрема такі, як: виникнення затримок і можливості втрати сегментів, недостатність ресурсів або можлива неготовність серверу до встановлення з'єднання, та інші. Крім того, тут не врахований той факт, що ще на початковому кроці здійснення встановлення з'єднання модулі TCP на стороні відправника і одержувача домовляються про деякі параметри взаємодії між собою, наприклад, про початкові номери байтів, які вони посилають [6].

2 АНАЛІЗ МЕХАНІЗМІВ УПРАВЛІННЯ ПОТОКАМИ ТРАФІКУ, ЩО РЕАЛІЗУЮТЬСЯ ПРОТОКОЛОМ TCP У ПРОЦЕСІ ПЕРЕДАЧІ ДАНИХ

Один із найефективніших і найвідоміших методів, що використовуються для організації надійного обміну даними та управління, є передача з підтвердженням (квитуванням). Суть цього методу полягає в тому, що вузол-відправник відсилає дані, а вузол-одержувач підтверджує їх отримання так званими квитанціями. Якщо вузол-відправник вчасно не отримує підтвердження (квитанції) на передані дані, то він здійснює їх повторну передачу. Така схема, що реалізується таким методом квітування, отримала назву «запит повторної передачі» (Automatic Repeat reQuest, ARQ). У разі її реалізації виникає безліч питань, наприклад [6, 8]:

- чи має відправник чекати, поки надійде повідомлення про підтвердження на відправлений пакет (сегмент, кадр, повідомлення), перш ніж відсилати наступний?

- чи має вузол-одержувач підтверджувати надходження кожного пакета або відразу робити підтвердження декількох?

- який час очікування повідомлення про підтвердження відправником є прийнятним?

- що має статися у випадку, коли повідомлення про підтвердження буде втрачено і вузол-відправник ще раз передасть той же самий пакет?

- яким чином вузол-одержувач має розпізнавати дублікати пакетів, а вузол-відправник – дублікати квитанцій про підтвердження?

Усі можливі підходи щодо надання відповідей на зазначені питання можна згрупувати у дві основні групи методів здійснення управління потоками трафіку на основі підтверджень: простою джерела (Stop-and-Wait) і ковзного (плаваючого вікна) вікна. Метод управління трафіком на основі ковзного вікна складається з двох основних груп механізмів [6, 9]:

- механізми, що працюють з використанням вікна передачі – до них відноситься наприклад передача з поверненням на N пакетів (Go-Back-N);

- механізми, що працюють з використанням вікна передачі і вікна прийому, наприклад, механізм передачі з вибірковим повторенням (Selective Repeat).

Треба мати на увазі, що всі ці механізми функціонують за наступних умов [6]:

- вузол-відправник і вузол-одержувач функціонують в асинхронному режимі та реалізують передачу пакетів по ненадійній лінії зв'язку (тобто в ній можливі спотворення, виникнення непередбачуваних затримок і втрат пакетів;

- вузол-відправник приймає дані від додатку (протоколу верхнього рівня), а вузол-одержувач здійснює передачу отриманих даних до додатку на верхній рівень;

- вузол-одержувач має механізм відстеження і визначення спотворених пакетів, наприклад, за алгоритмом перевірки контрольної суми;

- у разі успішного отримання пакету вузол-одержувач передає до вузла-відправника підтвердження (зокрема, як зазначалося, у протоколі TCP це може бути прапор АСК, що встановлюється в «1»);

- для реалізації відстеження затримок пакетів застосовується таймер, тайм-аут якого має бути рівним граничному часу очікування повідомлення про підтвердження (див рис. 1.6).

Розглянемо ці механізми детальніше поки що без прив'язки до конкретного транспортного протоколу.

2.1 Механізм простою джерела

У випадку застосування цього механізму вузол-відправник здійснює передачу послідовності пакетів без їх нумерації. При цьому вузол-відправник мусить дочекатися від вузла-одержувача повідомлення про підтвердження доставки першого пакета і тільки після цього він надсилає наступний пакет. Кожен відправлений пакет вузол-відправник супроводжує таймером: якщо впродовж визначеного часу підтвердження не надходить, то це говорить про те, що пакет або квитанція втрачена чи пошкоджена, і тоді відправку пакету треба повторити [6].

Як показано на рисунку 2.1, другий пакет передається тільки після отримання повідомлення-підтвердження про доставку першого. Проте між другим і третім пакетами виникла затримка: вузол-відправник, не дочекавшись повідомлення про підтвердження (ймовірно воно було втрачене), надсилає другий пакет повторно. У результаті вузол-одержувач тепер отримає і оригінал другого пакета, і його дублікат. Таким чином, у разі використання цього механізму вузол-

одержувач повинен мати інструментарій для виявлення та відкидання у разі потреби дубльованих пакетів [6].

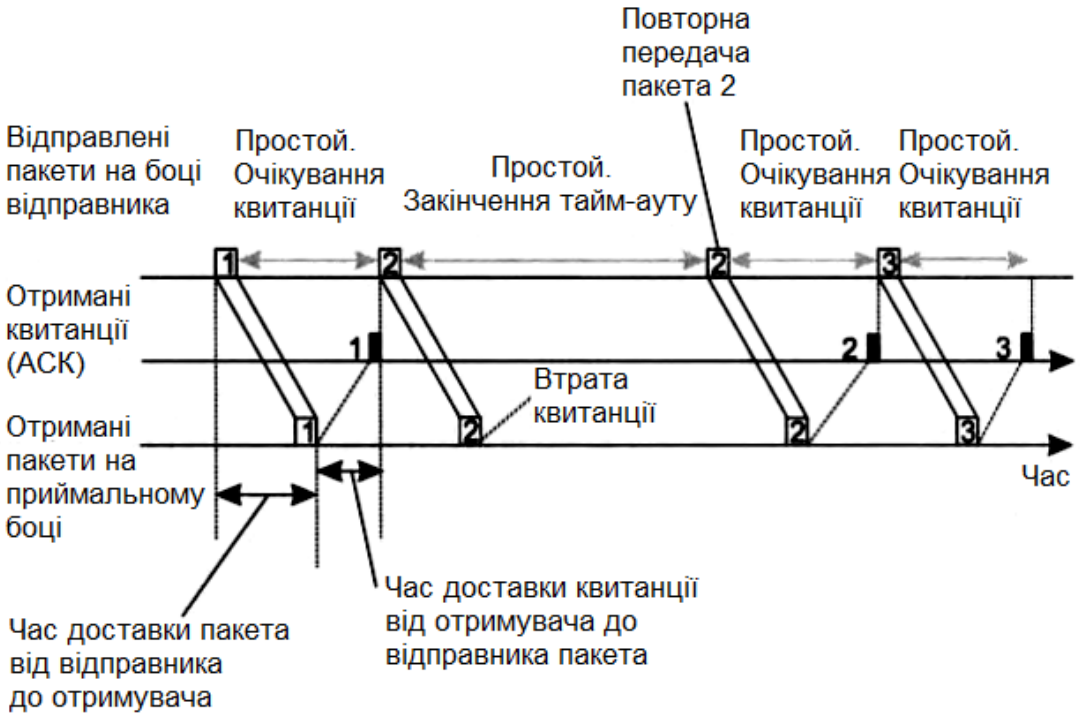

Рисунок 2.1 – Діаграма роботи механізму простою джерела

Якщо ж перше повідомлення про підтвердження не було втрачене, а просто надійшло із запізненням, то може виникнути плутанина. Тобто вузол-відправник може помилково прийняти друге підтвердження (на дублікат) за квитанцію доставки наступного пакета. Це означає, що вузол-відправник також повинен мати інструментарій, що дозволить здійснювати розпізнавання дублікатів повідомлень-підтверджень, щоб уникнути помилкової інтерпретації отриманих відповідей [6].

Цю проблему з визначенням дубльованих повідомлень можна частково вирішити за рахунок включення в заголовок пакета спеціального біта. Він налаштовується вузлом-відправником таким чином, що «0» і «1» в якості його значення мають чергуватися в межах всієї послідовності пакетів, що відправляються. На приймальній стороні перевіряється значення цього біта: якщо у двох пакетів, що послідовно прийшли, його значення будуть дорівнювати двом «1» або двом «0», то ці пакети будуть сприйматися як дублікати. Аналогічним способом працює і сторона передачі з підтвердженнями. Такий спосіб виявлення

дублікатів не є повністю надійним, туму що базується на припущенні про те, що імовірність надходження у вузол-одержувач дублікатів один за одним є достатньо великою. Крім того, цей механізм у цілому забезпечує низький коефіцієнт використання каналу, оскільки сторона, що здійснює передачу, значну частину часу перебуває в очікуванні приходу повідомлення про підтвердження [6].

2.2 Загальна концепція механізму ковзного вікна

Суть цього механізму полягає в тому, що для того, щоб підвищити швидкість ПД, вузол-відправник отримує право передати деяку кількість пакетів, не чекаючи надходження на ці пакети підтверджень [6].

У разі використання цього механізму полягає в тому, що можуть одночасно існувати у мережі кілька непідтверджених пакетів. Ці пакети необхідно вміти розрізняти, щоб вузол-відправник мав змогу з’ясувати, для якого пакета прийшло повідомлення про підтвердження. Для цього пакетам у їх заголовках прописуються послідовні номери, що є унікальними. Діапазон таких можливих унікальних номерів визначає у полі заголовку «Номер пакета». Коли цей діапазон закінчується, то призначення унікальних номерів пакетів знову починається з «0». Тобто, можна бачити, що повністю не можна нівелювати випадки, коли в мережі будуть існуватимуть пакети, що матимуть однакові унікальні номери. Саме вікно визначається якраз на цій послідовності пронумерованих пакетів (рис. 2.2). Воно завжди має нижню границю, що називається «базою вікна» (зокрема, на рис. 2.2 – це пакет, що має номер 9). Також вікно має і верхню границю (на рис. 2.2 – це пакет, що має номер 14). Номери пакетів, що будуть знаходитися у межах між нижньою і верхньою границями вікна, визначають його розмір (зокрема на рис. 2.2 розмір вікна становить 6-ть пакетів). Треба зазначити, що вікно завжди буде переміщуватися в бік пакетів з більшими номерами [6, 9].

Вікно, яке наведено на рис. 2.2, здійснює управління процесом передачі шляхом обмеження кількості пакетів, які вузол-відправник може передати до надходження підтвердження. Треба також зазначити, що вікно може бути окреслене не тільки для управління передачею, але і для управління прийомом пакетів (тут буде обмежуватися кількість пакетів, які вузол-одержувач може прийняти, не здійснюючи їх передачу далі на верхній рівень) [6].

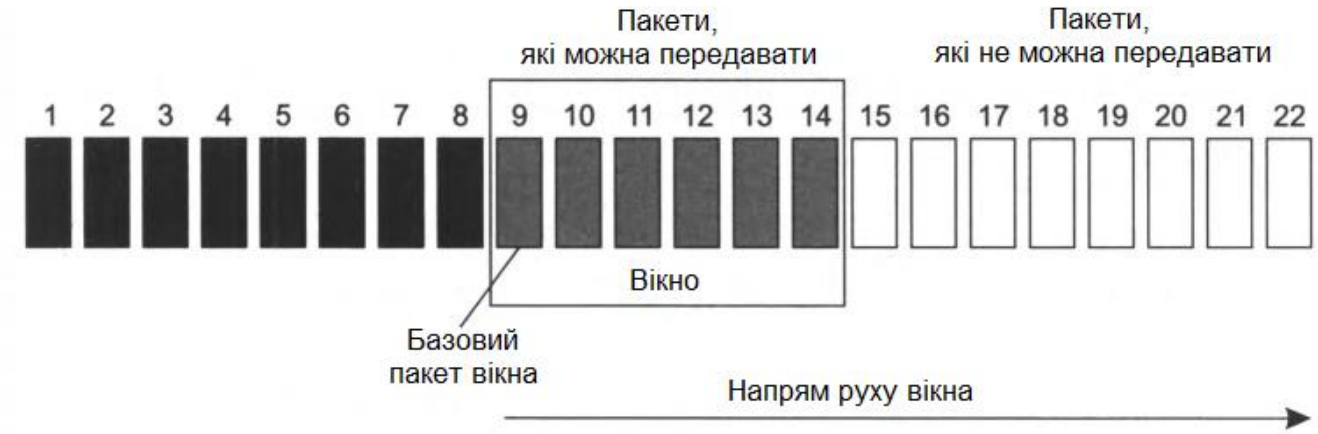

Рисунок 2.2 – Визначення розміру вікна передачі пакетів

На рисунку 2.3 показана типова діаграма роботи класичного механізму ковзного вікна на боці вузла-відправника.

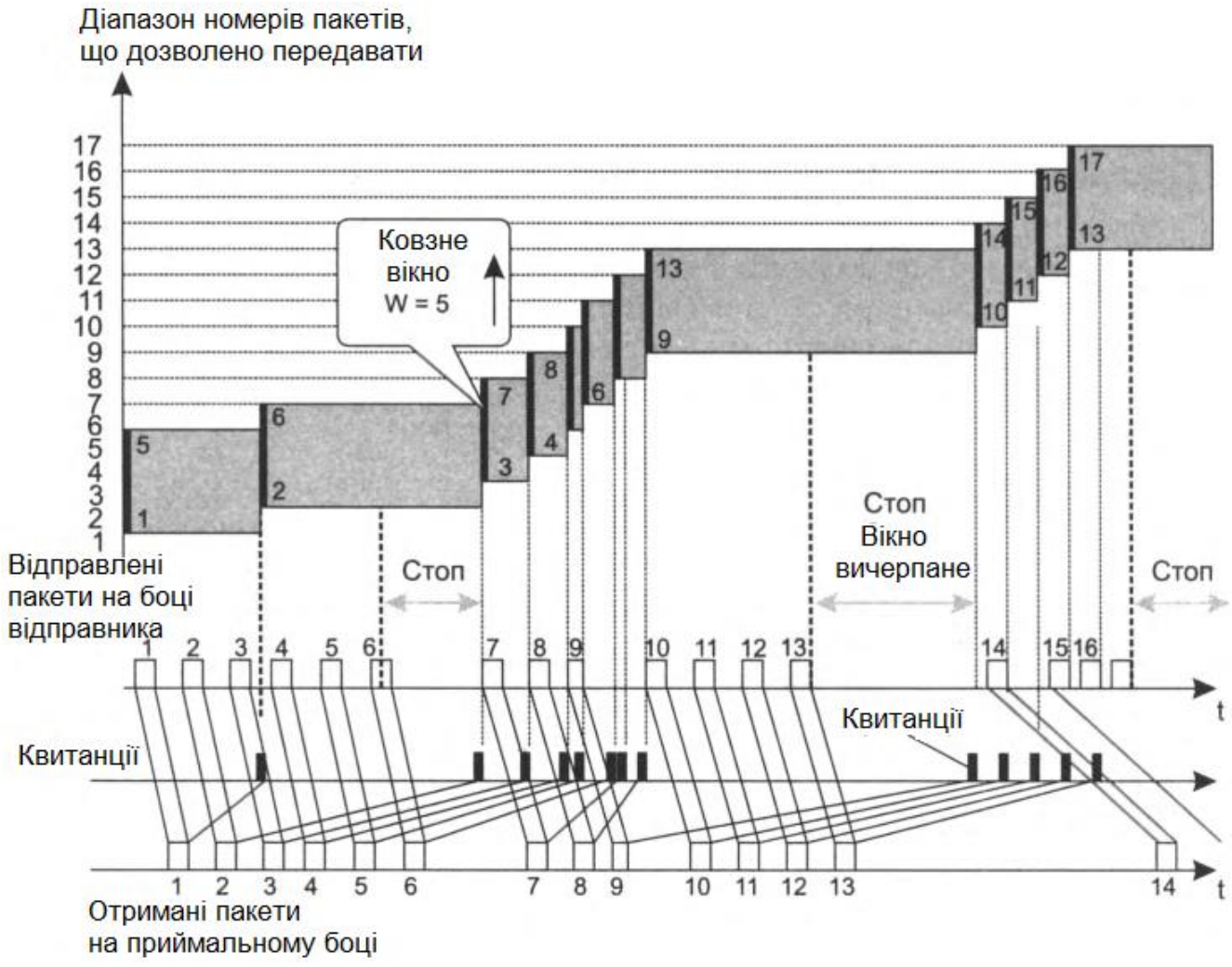

Рисунок 2.3 – Діаграма роботи механізму ковзного вікна

Тут робиться припущення, що пакети і підтвердження на них не губляться і поступають у тому ж самому порядку, у якому було здійснено їх відправлення. При цьому треба звернути увагу, що інтервали між пакетами і підтвердженнями є нерівномірними. Переміщення вікна визначається надходженням підтверджень, тобто повідомлення, що підтверджує успішне надходження чергового пакета надає змогу зробити переміщення вікна уперед. Також переміщення вікна визначається закінченням вікна, тобто вікно призупиниться, у тому випадку, коли вузол-відправник завершить передачу всіх пакетів з вікна, але при цьому не отримає підтвердження на жоден із них [6, 9].

На рисунку 2.3 номери пакетів зображені по вертикальній осі, тобто вікно рухається уздовж вертикалі. Розмір вікна складає п’ять пакетів. На початковому етапі, коли жодний пакет ще не надійшов, вікно охоплює діапазон номерів пакетів від 1-го до 5-го включно. Вузол-відправник починає передавати пакети, а згодом отримує підтвердження про їх доставку. Для спрощення зробимо припущення, що підтвердження отримуються у тій же самій послідовності, що й самі пакети на іншому боці. Коли надходить підтвердження, що був доставлений перший пакет, вікно рухається на одну позицію вгору, і дозволений до передачі діапазон пакетів теж зміщується, зокрема до пакетів із номерами від 2-го до 6-го [6].

Варто звернути увагу, що процеси передачі пакетів і приймання підтверджень про їх доставку здійснюються незалежно один від одного. У наведеному на рис. 2.3 прикладі вузол-відправник продовжує здійснювати передачу пакетів, але певний час він не отримує підтверджень про їх доставку. Після відправлення шостого пакета діапазон розміру вікна закінчується, і вузол-відправник робить зупинку передачі. Лише після надходження підтвердження щодо доставки 2-го пакета вікно рухається далі вгору на 1 пакет, тобто тепер дозволений для передачі діапазон пакетів буде з 3-го по 7-ий. Аналогічний рух вікна вгору здійснюється після отримання кожного нового повідомлення підтвердження. Причому треба звернути увагу, що вікно рухається вгору на один номер пакета, але розмір вікна при цьому залишається незмінним. Коли надходить підтвердження на доставку 8-го пакету, вікно переміститься до діапазону з 9-го по 13-й номери пакетів і буде в цьому положенні досить тривалий час, тому що з невизначених причин вузол-відправник перестає отримувати повідомлення про підтвердження доставки пакетів. Після передачі пакета з номером 13, що є останнім дозволеним пакетом у діапазоні розміру вікна, вузол-

відправник знову зупиняє надсилання пакетів. Але передача пакетів буде відновлення, коли надійде підтвердження на 9-й пакет [6].

Після кожного відправленого пакету у вузлі-відправнику встановлюється тайм-аут. Якщо за час тайм-ауту підтвердження не надійде, пакет (або підтвердження на нього) буде вважатися втраченим і він буде повторно передаватися. Якщо ж підтвердження будуть надходити регулярно в границях діапазону розміру вікна у 5 пакетів, то у цьому випадку швидкість обміну пакетами стане максимально можливою для наявного каналу і транспортного протоколу, що застосовується [10].

Загалом механізм ковзного вікна є складнішим за механізм простою джерела, оскільки потребує зберігання копій усіх непідтверджених пакетів у буфері вузла-відправника. До того ж, застосовуючи цей механізм, необхідно здійснювати контроль за декількома параметрами, зокрема такими як: розмір вікна, номер останнього пакета, що був підтверджений, а також номер пакета, який ще можна передати без отримання нових підтверджень [10].

Алгоритм механізму ковзного вікна – це основний механізм, що використовується протоколом TCP для здійснення управління обміну даними. Однак класичний варіант цього алгоритму, що був розглянутий вище, більш притаманний використанню у протоколах подібних до у протоколах подібних до старих X.25 і HDLC (High-Level Data Link Control – біт-орієнтований протокол управління канального рівня) – де на кожний блок даних має бути отримане підтвердження. Зокрема недолік такого класичного підходу полягає у тому, що тільки один пакет має бути переданий за один сеанс, тобто пакет має бути переданий і на нього має бути отримане підтвердження. Така реалізація механізму ковзного вікна не дозволяє забезпечити максимальну продуктивність обміну даними у мережі. Значно підвищити ефективність застосування цього механізму можливо, якщо дозволити передачу деякої множини пакетів за один сеанс з можливістю групування усіх підтверджень для них до одного. Далі розглянемо варіанти механізму ковзного вікна, що втілюють вищезазначені принципи і реалізуються практично для управління потоками трафіку саме протоколом TCP. Зокрема розглянемо наступні алгоритми роботи механізму ковзного вікна:

- управління передачею з поверненням на *N* пакетів;
- управління передачею з вибірковим повторенням.

Зауважимо, що у класичній схемі ковзного вікна одиницею даних, що передаються є пакти (або кадри), у той час, як в протоколі TCP – це сегменти, про що йшла мова у першому розділі. У контексті проведення викладення, що застосовується у цій кваліфікаційній роботі будемо вважати, що сегмент – це є аналог пакету, в аспекті того, що в ньому (сегменті) теж переносяться дані. Тому далі у цьому розділі під потоком даних будемо мати на увазі потік пакетів, для можливості проведення зв'язку з класичною схемою.

2.3 Механізм управління передачею з поверненням на *N* пакетів

У разі застосування цього варіанту механізму ковзного вікна основним завданням є потреба у підвищення ефективності використання каналу зв'язку у порівнянні з механізмом з простоєм джерела. Для цього вузлу-відправнику можна відправляти пакети (у кількості, яка обмежується розміром вікна) без підтвердження вузлом-одержувачем їх успішного прийому. Тому що вузол-відправник повинен мати можливість у разі потреби зробити повторне відправлення будь-якого з пакетів, що був переданий, то їх потрібно буферизувати. Крім того, він повинен слідкувати за статусом пакетів (відправлений чи не відправлений, був підтверджений чи ні). На рис. 2.4 наведене вікно передачі у певний часовий момент. Тут базою вікна є пакет з номером 7-м, а верхньою границею вікна – є пакет, що має номер 17-ть. Всі пакети з номерами, які знаходяться в цьому діапазоні, що обмежує розмір вікна, вузол-відправник може передавати без отримання підтверджень. З лівого боку вікна можна бачити більш «старі» пакети (з номерами від 1-го до 6-го). Вони вже були передані і на них надійшло підтвердження. Пакети в діапазоні від 18-го до 22-го (що можна бачити праворуч від вікна) на цей момент часу передавати недозволено [6].

У межах вікна є пакети, які вже були відправлені, але ще не були підтверджені (з 7-го по 12-й). Також є і ще не відправлені пакети, але які дозволено передати (з 13-го по 17-й). Відразу після того, як на найстаріший пакет у вікні (пакет з номером 7-м) надійде підтвердження про його успішну доставку, вікно зміститься вправо на одну позицію.

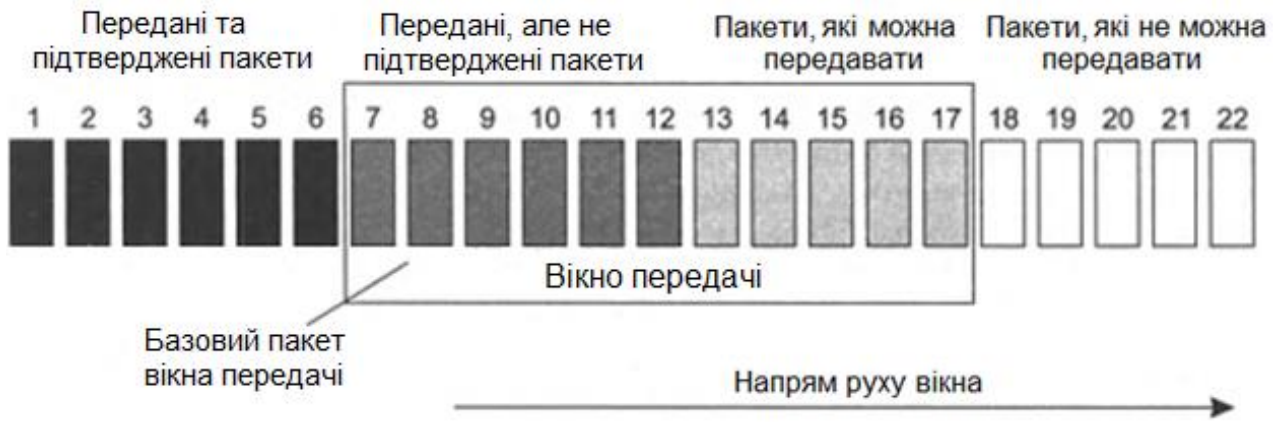

Рисунок 2.4 – Механізм віконного управління передачею з поверненням на N пакетів

Розглянемо алгоритм роботи вузла-одержувача. У разі надходження нового пакету вузол-одержувач завжди робить одні та ті ж самі дії, здійснюючи перевірку [6, 9]:

- чи є пакет неспотвореним;

- чи є пакет наступним за порядком у послідовності пакетів, які вже були отримані.

Якщо ці дві зазначені вище дії виконуються, то вузол-одержувач надсилає підтвердження з номером пакета, що був успішно прийнятий. Якщо хоча б одна з цих дій не буде виконана, то пакет буде відкинуто, а вузлу-відправнику буде надіслане підтвердження з номером останнього пакета, що був успішно прийнятий до цих дій. Зазначимо, що підтвердження у цьому механізмі управління передачею пакетів є накопичувальними. Тобто, якщо вузол-відправник отримав підтвердження з номером *n*-го пакету, то і всі пакети до нього також вважаються такими, що були прийняті правильно, тому що вузол-одержувач приймає неспотворені пакети тільки тоді, коли вони прибувають у точній відповідності з їх порядковими номерами в послідовності пакетів (можна сказати, що у вузла-одержувача є вікно, що має розмір в один пакет). Тобто, не треба здійснювати дублювання втраченого підтвердження, тому що воно буде компенсоване наступним підтвердженням, що надійде, яке теж буде накопичуваним [6, 9].

Тепер проаналізуємо алгоритм роботи на боці вузла-відправника. Вузол-відправник відправляє пакети, що не виходять за межі діапазону вікна, і встановлює таймер, значення тайм-ауту якого є рівним максимальному часу очікування підтвердження на базовий пакет. Після закінчення цього тайм-ауту вузол-відправник робить висновок, що пакет або підтвердження на нього були

втрачені, і відправляє його та всі інші пакети, що були відправлені, ще раз, тому що, якщо базовий пакет не був успішно прийнятий, то й решта тим більше не є прийнятими (саме цьому цей механізм і отримав назву «з поверненням на N пакетів») [6, 9].

Може також трапитися випадок, коли базовий пакет і кілька наступних були успішно доставлені, але підтвердження на них були втрачені. У такій ситуації вузол-відправник зробить повторне відправлення декількох пакетів. Вузол-одержувач, коли отримає дублікати пакетів, зрозуміє, в чому справа, і здійснить відправку квитанції останнього пакету, що був успішно доставлений (спрацює те, що підтвердження є накопичувальними, як про це вже згадувалося вище) [9].

Таким чином зауважимо, що алгоритм роботи вузла-одержувача в цьому механізмі є значно простішим, ніж алгоритм роботи вузла-відправника. Вузол-одержувач не застосовує вікно, а отже, не має потреби у здійсненні буферизації пакетів та моніторингу їх статусу. Від вузла-одержувача очікується лише розпізнавання помилкових пакетів та моніторинг послідовності їх номерів. Ефективність даного механізму є вищою у порівнянні з механізмом простою джерела за рахунок передачі у канал передачі відразу декількох пакетів. Але для нього властива надмірність, тому що вузол-одержувач здійснює видалення не тільки спотвореного, але і пакета, що був прийнятий коректно, якщо його номер порушує загальну послідовність пакетів. Крім того, вузол-відправник здійснює повторну передачу не тільки втраченого або спотвореного пакету, але також і тих пакетів, що були передані після нього [6].

2.4 Механізм управління передачею з вибірковим повторенням

Цей ще один механізм управління передачею потоку трафіку, що є варіантом реалізації механізму «ковзного вікна» та який підтримується протоколом TCP. Тут вузол-одержувач може зробити вибіркові запити на здійснення повторної передачі окремого пакета, а не всієї послідовності пакетів, що були передані, як це робиться у разі застосування механізму управління передачею із поверненням на N пакетів [6].

Щоб уникнути надмірних повторних передач пакетів, вузлу-одержувачу не дозволяється скидати пакет, що був коректно прийнятий, тільки через те, що його порядковий номер не відповідає очікуваній послідовності. Це означає, що такі

пакети, що «порушують» послідовність, повинні тимчасово зберігатися вузлом-одержувачем. Для цього використовується буфер, в якому вхідні пакети сортуються за їх номерами. Згідно із основною звдачею цього механізму, що полягає у здійсненні вибіркового повторення помилкових пакетів, то вибірковими (тобто індивідуальними) також мають бути і підтвердження, що дозволяють здійснювати підтвердження достовірності прийому кожного окремого пакета, а також і таймери, що роблять вимірювання тайм-аутів для кожного переданого пакету. У цьому механізмі концепція «ковзного вікна» застосовується як для здійснення передачі пакетів, так і для їх прийому. Тобто тут буде два вікна з однаковими розмірами і обидва будуть визначатися для однієї і тієї ж послідовності номерів пакетів. Однак рух цих вікон не обов'язково буде синхронізованим - у разі виникненні помилок вікно прийому може випереджати вікно передачі. Вузол-відправник і вузол-одержувач здійснюють обробку тільки тих пакетів, які знаходяться в межах відповідних вікон. На рисунку 2.5 показано приблизне положення вікон передачі та прийому в певний момент часу [6, 9].

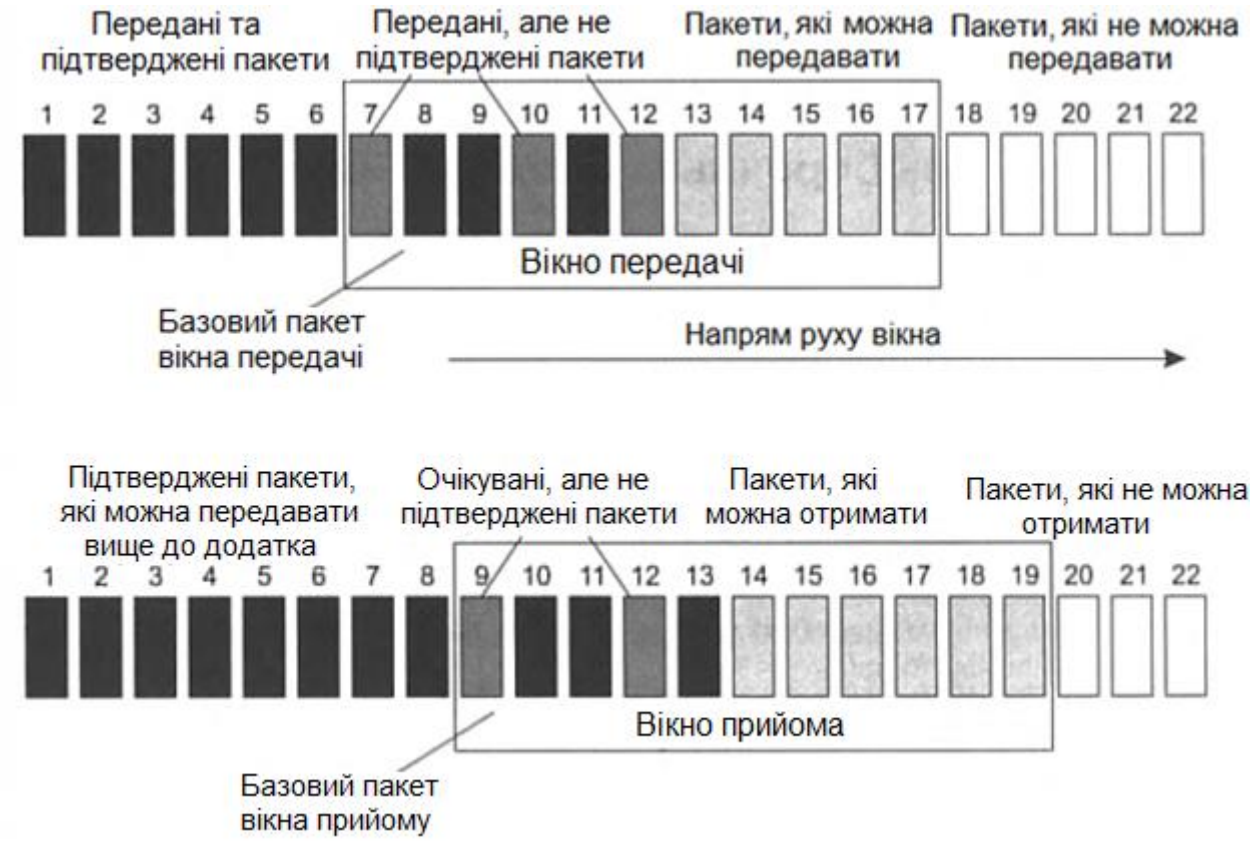

Рисунок 2.5 – Механізм віконного управління передачею і прийомом пакетів у разі вибіркового повторення

Вузол-відправник вже відправив пакети з номерами 1 - 6, що були прийняті від верхнього рівня, отримав на них підтвердження і знаходиться в очікуванні підтвердження на пакет з номером 7-м, що також був переданий. У свою чергу вузол-одержувач зібрав і відправив до свого верхнього рівня послідовність пакетів, що мають номери від 1-го до 8-го, але він все ще не одержував пакет за номером 9-ть. Вікно передачі в даний момент знаходиться між базовим пакетом із номером 7-м і граничним пакетом із номером 17-ть, а вікно прийому знаходиться в діапазоні пакетів із номерами від 9-го до 19-го. Всі пакети з номерами, що потрапляють у діапазони розмірів вікон, вузлу-відправнику дозволено передавати, а вузлу-одержувачу приймати без надходження підтверджень.

Оскільки в цьому механізмі підтвердження є індивідуальним, а не накопичувальним, як у попередньому механізмі, управління передачею пакетів, що був розглянутий вище, то у вікні передачі серед відправлених пакетів можуть бути як ті, що підтверджені (пакети з номерами 8-м, 9-ть, 11-ть), так і ті, що непідтверджені (пакети з номерами 7-м, 10-ть, 12-ть). Відповідно номери з вікна прийому в загальному випадку можуть відповідати [6]:

- прийнятим і підтвердженим пакетам (це пакети 10-ть, 11-ть та 13-ть), передачу яких не можливо здійснити з буфера на верхній рівен, тому що до їх послідовності додалися номери деяких пакетів, що є відсутніми (це пакети 9-ть і 12-ть);

- підтвердженням пакетів, що очікуються, але ще отриманими (пакети 9-ть і 12-ть);

- пакетами, що вузлу-одержувачу дозволено приймати (пакети від 14-го до 19-го), оскільки їх номери попадають у діапазон, що визначається розміром вікна прийому.

Вузол-одержувач здійснює прийом, розміщення в буфері і підтвердження будь-якого пакету, що був прийнятий, за умови, що він не є спотвореним та його номер потрапляє у діапазон, що обумовлюється розмірами вікна прийому. Якщо був прийнятий базовий пакет, то ліва границя вікна прийому зсувається до першого пакету, що очікується, але який ще не отриманий. На рис. 2.5 базовим у вікні прийому є пакет з номером 9-ть. Коли його буде успішно прийнято, то порожнеча з відсутнім пакетом заповниться, вікно зміститься, і новим базовим пакетом стане пакет з номером 12-ть. Неперервна послідовність пакетів з номерами 9-ть, 10-ть та 11-ть, що вийде за межі вікна прийому, тоді може бути передана до верхнього рівня [6].

У ряді випадків пакет може бути успішно прийнятий вузлом-одержувачем, але відповідне підтвердження на нього втрачається в процесі його передачі до вузла-відправника. В результаті вузол-відправник, не отримавши підтвердження у встановлений час, повторно передає той же самий пакет. Вузол-одержувач, отримавши дублікат, не повинен його ігнорувати – в даному випадку він зобов'язаний повторно відправити підтвердження, оскільки в іншому випадку вузол-відправник буде вважати, що пакет не доставлений, і продовжить нескінченно retransmit-ить його, блокуючи просування вікна передачі [6, 9].

Незважаючи на підвищені вимоги до буферизації та обробки таймерів, механізм вибіркового підтвердження дозволяє істотно скоротити кількість зайвих повторних передач і підвищити загальну ефективність каналу. На цей час більшість реалізацій транспортного рівня (наприклад, протокол TCP) використовують саме цей механізм управління передачею трафіку з підтвердженням доставки [9].

2.5 Механізм управління потоком трафіку по схемі з виділенням ліміту

2.5.1 Поняття сегменту і потоку байтів за протоколом TCP

Аналізуючи попередні варіанти механізмів управління потоками трафіку, що реалізуються протоколом TCP, для уніфікації викладення їх загальних особливостей було введено допущення, що одиницею даних, що передаються, є пакет, а сегмент, як одиниця даних протоколу TCP, є аналогом пакету. На загальному рівні опису роботи алгоритмів це є прийнятним, але надалі будемо застосовувати термін сегмент, для більш точного опису принципів його функціонування щодо здійснення управляння потоками трафіку. Зокрема стосовно роботи розглянутих механізмів треба розуміти, що розмір вікна визначається на множині нумерованих байтів потоку даних, що є неструктурованими, та які надходять з верхнього рівня на транспортний рівень і буферизуються там протоколом TCP.

У процесі встановлення з'єднання модулі TCP вузла-відправника і вузла-одержувача, які збираються взяти участь в обміні даними, погоджують параметри, за якими буде відбуватися процедура обміну ними. Деякі з цих параметрів залишаються незмінними протягом всієї сесії, тоді як інші, наприклад, що

залежать від рівня мережного завантаження або розмірів буферної пам'яті, можуть динамічно змінюватися. Одним з таких параметрів є початковий номер байта (ПНБ), від якого буде вестися відлік протягом усього часу функціонування даного з'єднання. На передаючій і приймаючій стороні будуть свої ННБ. Нумерація байтів всередині TCP-сегмента ведеться від його заголовка, як це показано на рисунку 2.6 [6, 10].

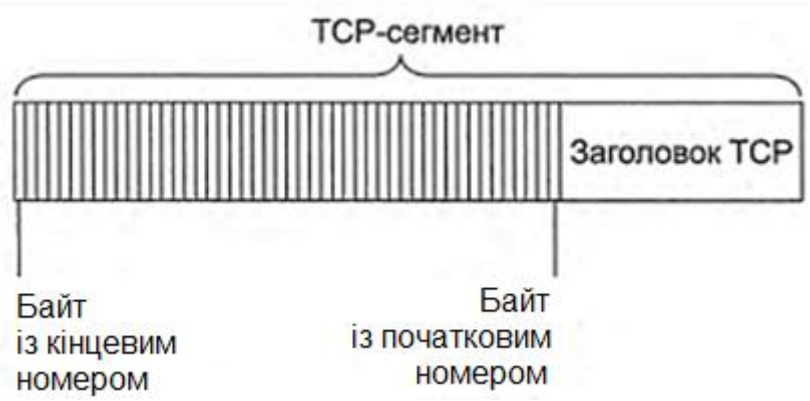

Рисунок 2.6 – Нумерація байтів у межах сегменту TCP

Коли вузол-відправник надсилає сегмент TCP, він розміщує у полі «Порядковий номер» (див. рис. 1.2) номер першого байту цього сегмента, який також буде його ідентифікатором (ID). На рисунку 2.7 показані чотири TCP-сегменти, які мають розмір по 1460 байтів і один сегмент розміром в 870 байтів. ID цих TCP-сегментів будуть номери 32600, 34060, 35520, тощо. На підставі цих номерів вузол-одержувач сегмента не тільки відрізняє його від інших сегментів, але і може виокремити отриманий фрагмент у загальному потоці байтів. Крім того, вузол-одержувач може на основі отриманих сегментів зробити певні висновки, наприклад, що він є дублікатом або що між двома отриманими TCP-сегментами є пропущені дані, тощо [6, 10].

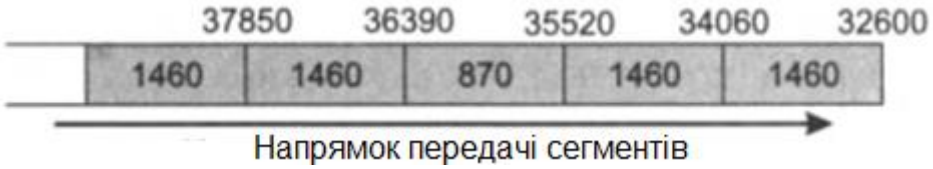

Рисунок 2.7 – Нумерація TCP-сегментів за їх розмірами

В якості підтвердження вузол-одержувач сегмента передає відповідний сегмент, де в полі «Номер підтвердження» якого (див. рис. 1.2) він розміщує число, яке на одиницю перевищує максимальний номер байту, що є в отриманому сегменті. Так, для першого відправленого сегмента (рис. 2.7) підтвердженням про одержування (підтвердженим номером) буде номер 34060, для другого – номер 35520 і т. д. Підтверджений номер часто інтерпретують не тільки як повідомлення про успішну доставку, але і як номер наступного байту даних, що очікується. Ще раз зазначимо, що підтвердження в протоколі TCP відправляється тільки в разі правильного прийому даних. Іншими словами, відсутність повідомлення про підтвердження буде означати або втрату TCP-сегмента, або втрату підтвердження, або те, що був прийнятий спотворений сегмент. Також слід мати на увазі, що один і той же TCP-сегмент може нести в собі як дані користувача (в полі даних), так і підтвердження (в заголовку), яке підтверджує успішне отримання даних з протилежного боку [6, 10].

2.5.2 Схема управління потоком трафіка із виділенням ліміту на передачу даних

Механізм управління потоком трафіку за механізмом із виділенням ліміту на здійснення передачі даних є більш гнучким алгоритмом, ніж раніше розглянуті варіанти реалізації механізму ковзного вікна.

За схемою цього механізму кожен байт, що передається має свій власний номер у послідовності (Sequence Number, SN). Коли протокол TCP надсилає сегмент, він виставляє в полі «Порядковий номер» номер першого байта в послідовності байтів, що розміщуються в полі даних цього сегмента. На стороні вузла-одержувача сегмент, що надійшов, підтверджується повідомленням, в якому вказується ($A=i,\ W=j$). Такий запис має таке трактування: здійснюється підтвердження отримання всіх байтів, аж до номера (i – 1); наступні байти, що очікуються, мають якийсь номер у послідовності i. Крім того, надається дозвіл на відправлення додаткового вікна W (Window), що має якусь кількість j даних, та де байти даних мають номери у послідовності від і до $i + j$ – 1. На рис. 2.8 показаний принцип функціонування даного механізму, де для більшої зрозумілості показано потік даних, що рухається тільки в одному напрямку. Також зроблено припущення, що в кожному TCP-сегменті здійснюється передача 200 байтів даних [1, 11].

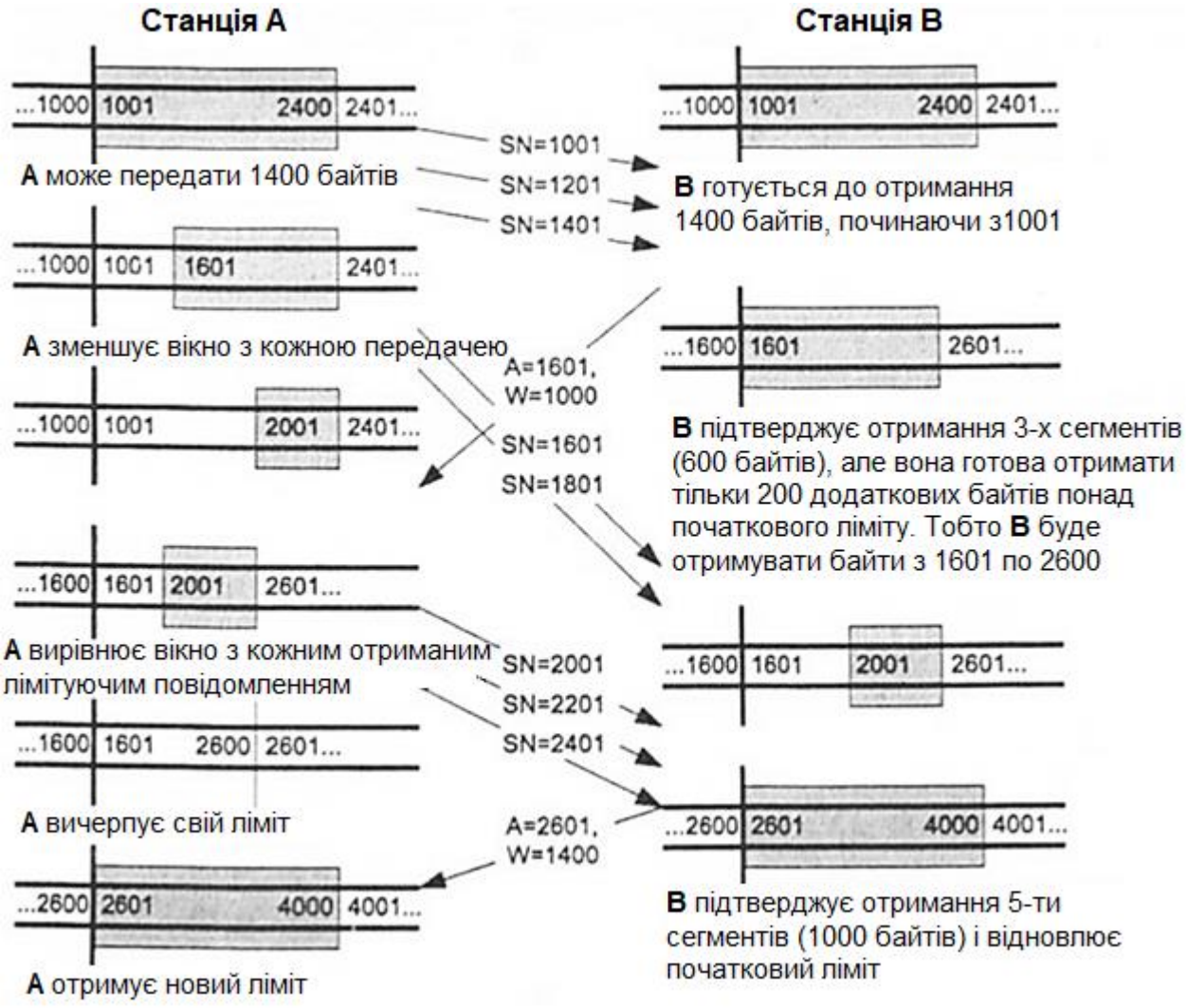

Рисунок 2.8 – Схема функціонування управління потоком із виділенням ліміту на передачу даних

Під час встановлення з'єднання номери в послідовностях вузла-відправника та вузла-одержувача є синхронізовані, і станція «А» має початковий ліміт на передачу даних 1400 байт, починаючи з байту, що має номер 1001. Після здійснення передачі 600-от байт у трьох TCP-сегментах станція «А» зменшує своє вікно передачі до 800-от байтів (вони мають номери від 1601 до 2400). Після надходження цього сегмента станція «В» робить підтвердження щодо отримання всіх байтів, аж до 1601, і створює своє вікно прийому на 1000 байт. Це говорить про те, що станція «А» може передавати байти, починаючи з номера 1601 і закінчуючи номером 2600, тобто п'ять TCP-сегментів. Однак до того моменту, коли повідомлення від станції «В» надходить до станції «А», вона вже встигає надіслати два TCP-сегменти, що містять байти з номерами від 1601 до 2000, що

було дозволено за наданим початковим лімітом. Отже, залишок ліміту станції «А» на цей момент буде всього 400 байт або за значених вище умов - це два TCP-сегменти. У процесі здійснення обміну станція «А» зміщує кінцеву (ліву) границю свого вікна кожного разу, коли робить передачу сегмента. Зсув початкової (правої) границі вікна стає можливим лише тоді, коли станція «А» дістає новий ліміт [1, 11].

На практиці з обох сторін одночасно застосовуються режими передачі та прийому, тому що потік трафіку передбачає передачу даних в обох напрямках (тобто реалізується режим повнодуплексної передачі). Механізм надання ліміту є досить гнучким. Наприклад, візьмемо ситуацію, у якій останнє повідомлення, що було надіслане станцією «В», мало формат: $A=i$, $W=j$. Останнім байтом даних, що був отриманий станцією «В», був байт з номером i – 1. В результаті для підвищення ліміту до значення k, якщо виконується умова $k > j$, та ще не надійшли додаткові дані, станція «В» формує повідомлення ($A=i$, $W=k$). Для підтвердження вхідного TCP-сегмента, що містить m байтів даних ($m < j$) без надання додаткового ліміту, станція «В» формує повідомлення ($A=i+m$, $W=j-m$) [1, 11].

Треба зазначити, що від вузла-одержувача не потрібне негайне підтвердження сегментів, що надходять. Він може деякий час чекати, а потім згенерувати підтвердження відразу на кілька сегментів. Вузол-одержувач мусить дотримуватися певної політики, що регулює кількість даних, яку він сам дозволяє відправляти вузлу-відправнику. Можна звернути увагу на дві політики вузла-одержувача: консервативну і оптимістичну. Консервативна політика управління потоком трафіку ґрунтується на тому, що ліміт відводиться відповідно до фактично наявного буферного простору. Якщо цю політику застосувати до ситуації, що розглядається на рис. 2.8, то перше лімітуюче повідомлення говорить про те, що станція «В» може розмістити 1000 байтів у своєму буфері, а друге лімітуюче повідомлення говорить про те, що станція «В» може розмістити 1400 байтів. Консервативна політика управління потоком може обмежити пропускну здатність логічного з'єднання у випадку, коли в мережі з'являються великі затримки [1, 11].

Вузол-одержувач може більш ефективно використовувати пропускну здатність каналу і навіть її збільшити за допомогою застосування оптимістичної політики для надання ліміту на вільний буферний простір, якого він на даний момент по факту поки ще не має. Наприклад, якщо буфер вузла-одержувача заповнений, але він очікує, що зможе звільнити 1000 байтів свого буферного

об'єму за час обміну інформацією між кінцевими вузлами, що беруть участь у з'єднанні, то він може відразу виділити ліміт у 1000 байтів. Якщо вузол-одержувач може підтримувати швидкість, що задана вузлом-відправником, то така політика дозволяє і підвищити пропускну здатність каналу, і не завдасть при цьому шкоди. Якщо ж вузол-відправник буде функціонувати швидше за вузол-одержувач, то в цьому випадку деякі TCP-сегменти будуть відкидатися через завантаженість буферу, і, як наслідок, це спричинить за собою повторну передачу. У такому випадку оптимістична політика управління потоком трафіку може підвищити імовірність виникнення перевантаження в мережі [1, 11].

3 ЗАГАЛЬНА МЕТОДИКА ОЦІНКИ ЧАСОВИХ ХАРАКТЕРИСТИК У ПРОЦЕСІ ПЕРЕДАЧІ ПІДТВЕРДЖЕНЬ НА ДОСТАВКУ TCP-СЕГМЕНТІВ

Як уже було згадано в попередньому розділі, протокол TCP не генерує і не відправляє негативні підтвердження, тобто такі підтвердження, які б явно вказували на виниклі збої та помилки. Натомість протокол спирається на позитивні підтвердження та механізм повторного відправлення, що активується у разі відсутності підтвердження у встановленому проміжку часу. Повторна відправка TCP-сегмента може бути ініційована двома причинами [1]:

- сегмент може бути пошкоджений в процесі передачі, однак, незважаючи на це, він буде доставлений до вузла-одержувача. Вбудована в TCP-сегмент контрольна сума дає можливість вузлу-одержувачу виявити в ньому помилку і відхилити;

- сегмент може не доходити до вузла-одержувача.

В обох випадках вузол-відправник не отримує негайного повідомлення про невдалу доставку сегмента. Воно приходить з деякою затримкою.

Якщо сегмент на стороні вузла-одержувача був прийнятий з помилками або зовсім не прийнятий, то в такій ситуації не формується і не відправляється підтвердження про його отримання. Це вимагає повторної відправки даного TCP-сегмента. Для визначення моменту повторної відправки використовується таймер, який запускається для кожного відправленого сегмента. Якщо тайм-аут таймера закінчується до отримання підтвердження для даного TCP-сегмента, вузол-відправник повинен відправити його повторно. Однією з ключових характеристик протоколу TCP є можливість регулювання тайм-ауту таймера. Це вкрай важливо, оскільки занадто короткий тайм-аут може призвести до частих непотрібних повторних відправлень TCP-сегментів, зменшуючи ефективність використання пропускної здатності каналу зв'язку або навіть мережі в цілому. Занадто великий тайм-аут не дозволить TCP швидко і правильно реагувати на втрату того чи іншого сегмента. Час тайм-ауту таймера повторної передачі слід встановлювати трохи більше, ніж час проходження туди і назад (час обігу) (Round Trip Time, RTT). Даний час обігу є складовою загальної затримки в мережі і залежить від безлічі факторів, які можуть змінюватися навіть у разі її стабільного завантаження [1].

Існує два підходи до налаштування тайм-ауту таймера повторної передачі.

Перший підхід орієнтований на використання фіксованого значення тайм-ауту таймера, який встановлюється на основі статистичних даних про

«нормальну» поведінку мережі. Тобто, визначається середнє значення RTT і тайм-аут встановлюється з невеликим запасом. Даний підхід ґрунтується на припущенні, що мережа працює в стабільному режимі, але при цьому не відбувається у разі необхідності гнучкої адаптації до різких змін мережних умов. Як вже було сказано, занадто тривалий тайм-аут призведе до уповільнення реакції протоколу, а занадто короткий тайм-аут може призвести до виникнення перевантаження в мережі через здійснення частих повторних відправлень TCP-сегментів [1].

Другий підхід орієнтований на використання адаптивного задавання тайм-ауту таймера повторної передачі. У цьому підході також є свої переваги і недоліки. Припустимо, що протокол TCP працює в режимі безперервного моніторингу часу отримання підтверджень на відправлені TCP-сегменти і коригує значення свого тайм-ауту на підставі затримки, що визначається під час моніторингу. Однак встановлене таким чином значення тайм-ауту не може вважатися ідеальним у всіх можливих ситуаціях з наступних причин [1]:

- підтвердження відправлення TCP-сегмента може відбуватися не відразу (нагадаємо, що зазвичай застосовуються сукупні підтвердження для декількох TCP-сегментів);

- якщо TCP-сегмент був відправлений повторно, то вузол-відправник не завжди може визначити, до якого сегменту відноситься отримане підтвердження – до початкового або до повторно відправленого;

- умови обміну даними в мережі можуть несподівано змінитися.

Варто підкреслити, що, з одного боку, ця проблема не має прийнятного універсального рішення, яке було б ідеальним у всіх випадках. Завжди буде присутній елемент невизначеності у виборі оптимального тайм-ауту для таймера повторної передачі. Однак, з іншого боку, всі реалізації протоколу TCP орієнтовані на проведення оцінки поточного часу передачі відповідно до підходу адаптивного задавання тайм-ауту таймера. Механізм проведення цієї оцінки в протоколі TCP ґрунтується, як зазначалося, на постійному моніторингу змін затримки надходження підтверджень про доставку на останні TCP-сегменти, які були відправлені. Після чого тайм-аут буде налаштований на значення, яке буде трохи більшим, ніж оцінений час передачі підтверджень TCP-сегментів [1, 12].

Розглянемо типову методику проведення такої оцінки, що базується на визначенні середнього часу затримки отримання підтверджень для певної кількості сегментів TCP, що були передані. Якщо це середнє значення з

достатньою точністю відображає очікувані затримки в майбутньому, то на його основі можна встановити значення тайм-ауту таймера повторної передачі, яке забезпечить високу ефективність і продуктивність. Для обчислення середнього часу використовується наступний вираз [1, 12]:

$$ARRT(K+1) = \frac{1}{K+1} \sum_{i=1}^{K+1} RTT(i), \tag{3.1}$$

де *RTT(i)* – час зворотного обігу, що спостерігається у процесі моніторингу для *i*-го переданого TCP-сегмента;

ARTT(K) – середній час зворотного обігу для визначених *K* сегментів TCP, що були відправлені першими.

Цей вираз може бути поданий у формі рекурентного співвідношення [1, 12]:

$$ARTT(K+1) = \frac{K}{K+1} ARTT(K) + \frac{1}{K+1} RTT(i). \tag{3.2}$$

Тут необхідно мати на увазі, що параметри RTT і ARTT в цьому виразі мають однакову вагу і кожний з них підвищується на константу 1/(*K*+1) (оскільки в знаменники обох параметрів входить один і той же вираз К+1). Як правило, більше значення ваги (більший рівень довіри) присвоюється більш новим, останнім результатам моніторингу, оскільки вони найбільш точно дозволяють передбачити подальшу поведінку мережі. Суть методу передбачення наступного середнього значення часу звернення на основі попередніх даних моніторингу часу зводиться до узагальнення формули (3.2), а саме [1, 12]:

$$SRTT\ (K + 1) = \alpha\ SRTT(K) + (1 - \alpha)\ RTT(K + 1), \tag{3.3}$$

де *SRTT*(*K*) (Smoothed Round Trip Time) – це параметр, що описує усереднену оцінку часу звернення.

Зробимо порівняння виразу (3.3) з попереднім виразом (3.2). Для цього скористаємося коефіцієнтом α, який приймає значення $0 < \alpha < 1$) та не залежить від кількості останніх даних у процесі моніторингу. Він дозволяє врахувати всі дані, що були отримані у процесі моніторингу, але за умови, що більш ранні дані

моніторингу за часом отримання будуть мати менші вагові коефіцієнти. Таким чином, коефіцієнт α характеризує так званий «фактор усереднення» за часом надходження підтверджень [1, 12].

Так, для значення α = 0,5 найбільш вагомими стають чотири або п'ять останніх моніторингових значень даних, тоді як для значення коефіцієнта α = 0,875 середня вага розподіляється для десяти і більше останніх отриманих значень даних моніторингу. Щодо переваги вибору невеликого значення коефіцієнта α є те, що при цьому відображаються швидкі зміни в значеннях величин даних, які отримані в процесі моніторингу. Недолік полягає в тому, що у випадку якщо має місце коротка хвилеподібна зміна в отриманих значеннях з подальшим переходом до усередненого, то використання невеликої величини коефіцієнта α призведе до різких змін розрахункового часу обігу [1, 12].

Звернемо увагу, що вираз (3.3) використовується в [12] для здійснення оцінки поточного часу обігу.

Як уже було сказано вище, тайм-аут повторної передачі необхідно встановлювати у значення, яке має бути трохи вищим, ніж час обігу, що отриманий в процесі проведеної оцінки. Взагалі тайм-аут повторної передач – це деякий проміжок часу, протягом якого вузол-відправник очікує підтвердження про успішну доставку відправленого TCP-сегменту. Якщо протягом цього часу підтвердження не надходить, то вузол-відправник вважає, що пакет було втрачено або пошкоджено, і здійснює повторну передачу сегменту. Треба звернути увагу, що у механізмі управління потоком трафіку на основі вибіркового повторення, для кожного TCP-сегменту використовується окремий таймер, що дозволяє більш точно та ефективно здійснювати повторні передачі лише коли у цьому є потреба. Для розрахунку цієї величини перевищення (тайм-ауту повторної передачі) можна застосувати наступний вираз [1, 12]:

$$RTO(K + 1) = SRTT(K + 1) + \Delta, \qquad (3.4)$$

де *RTO* (Retransmission Time Out) – тайм-аут повторної передачі;
Δ – коефіцієнт.

Недолік наведеного виразу (3.4) полягає в тому, що значення коефіцієнта Δ не є пропорційним значенню SRTT (тобто жодним чином не враховує цю

усереднену оцінку часу звернення). Для великих значень усередненої оцінки SRTT будь-який коефіцієнт Δ може виявитися занадто великим і спричинить непотрібні затримки при здійсненні повторних передач TCP-сегментів, що були втрачені. У зв'язку з цим в [12] визначається тайм-аут таймера пропорційно до значення SRTT з такими обмеженнями:

$$RTO(K+1) = \min(UBOUND, \max(LBOUVD, \beta \times SRTT(K+1))), \qquad (3.5)$$

де *UBOUND* та *LBOUND* – граничні фіксовані значення інтервалу очікування повторної передачі;

β – коефіцієнт, що описує так званий фактор зміни затримки.

У документі [12] не наполягається на введені якихось фіксованих значень інтервалу, але наводиться діапазон змін цих коефіцієнтів, зокрема: коефіцієнт α (вираз (3.3)) рекомендується встановлювати між 0.8 та 0.9, а коефіцієнт β (вираз (3.5)) – відповідно між 1.3 та 2.0.

ВИСНОВКИ

У цій кваліфікаційній роботі було зроблено аналіз механізмів управління потоками трафіку, що підтримуються протоколом TCP у процесі передачі даних. За напрям проведення аналізу був взятий найбільш поширений серед протоколів передачі метод передачі з підтвердженням на основі алгоритму ковзного вікна, робота якого направлена на запит повторної передачі у разі виявлення втрати або помилки при передачі TCP-сегменту. При цьому також приділялася значна увага підходам щодо налаштування тайм-ауту таймера повторної передачі. Також надана загальна методика проведення оцінки середнього часу затримки отримання підтверджень для певної кількості сегментів TCP, що були передані, і визначення тайм-ауту повторної передачі.

У першому розділі кваліфікаційної роботи були розглянуті загальні особливості та принципи роботи протоколу TCP. Протокол TCP призначений для здійснення передачі даних між додатками. Він має засоби управління потоком і корекції помилок і є орієнтованим на встановлення логічного з'єднання, що дає йому змогу забезпечувати надійність і достовірність обміну даними між процесами на комп'ютерах, які входять у загальну мережу. Показано, що інформаційною одиницею TCP є сегмент, який має свій формат і забезпечується заголовком. Структура заголовка TCP-сегмента і його елементи описані у роботі [4, 6].

Крім того у цьому розділі особлива увага приділена розгляду особливостей встановлення логічних з’єднань за протоколом TCP у процесі передачі даних. Робота у режимі встановлення логічного з'єднання дозволяє стежити за обміном даними, зокрема щоб вони не були загублені, отримані з помилками або продубльовані, а також, щоб дані вчасно надійшли до вузла-одержувача у тій же послідовності, в якій їх було відправлено [6].

У другому розділі кваліфікаційної роботи був проведений аналіз механізмів управління потоками трафіку, що реалізуються протоколом TCP у процесі передачі даних. Цей аналіз проводився для механізмів, що входять у склад найвідомішого методу, що використовує для організації надійного обміну даними та управління передачу з підтвердженням, та за своїми підходами застосування розділяються дві групи: простою джерела і ковзного вікна. Були розглянуті загальні класичні принципи застосування цих підходів у відповідних механізмах

управління потоками трафіку, та безпосередньо ті механізми, що підтримуються протоколом TCP при передачі даних.

Зокрема показано, що недоліком класичного підходу застосування ковзного вікна є те, що тільки один пакет може передаватися за один сеанс, тобто має здійснитися передача пакету і на нього має надійте підтвердження. Це не дозволяє класичному механізму ковзного вікна забезпечити максимальну продуктивність обміну даними у мережі ПД. Підвищити ефективність його застосування можливо, якщо дозволити передачу деякої множини пакетів за один сеанс з можливістю групування усіх підтверджень для них до одного. У цьому аспекті були розглянуті алгоритми роботи таких механізмів, як:

- управління передачею з поверненням на N пакеті;
- управління передачею з вибірковим повторенням;
- управління потоком трафіку по схемі з виділенням ліміту.

Зазначено, що механізм управління потоком трафіку за схемою із виділенням ліміту на здійснення передачі даних є більш гнучким алгоритмом, ніж інші варіанти реалізації механізмів, що відносяться до методу ковзного вікна.

У третьому розділі кваліфікаційної роботи запропонована і обґрунтована загальна методика оцінки часових характеристик у процесі передачі підтверджень на доставку TCP-сегментів. Показано, що якщо TCP-сегмент на боці вузла-одержувача був прийнятий за якихось причин не правильно чи зовсім був втрачений, то підтвердження за принципом роботи протоколу TCP не буде створене і відправлене. Це у свою чергу потребує здійснення повторної передачі даного TCP-сегмента. Для визначення моменту повторної відправки кожного відправленого сегмента треба застосовувати таймер. Зокрема, якщо тайм-аут таймера вичерпується до того, як надійде підтвердження на TCP-сегмент, то вузол-відправник має надіслати його повторно. Однією з ключових характеристик протоколу TCP є можливість регулювання тайм-ауту таймера.

У роботі були визначені підходи щодо здійснення налаштування тайм-ауту таймера повторної передачі та запропонована типова методика проведення вибору його оптимальних значень, що ґрунтується на оцінці середнього часу затримки отримання підтверджень для певної кількості сегментів TCP, що були передані. Показано, що тайм-аут повторної передачі необхідно встановлювати у значення, яке має бути трохи вищим, ніж час обігу, що отриманий в процесі проведеної оцінки. Наведений вираз для розрахунку величини тайм-ауту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Кульгин М. Компьютерные сети. Практика построения. Для профессионалов / Максим Кульгин – СПб: Питер, 2004. – 462 с.:

2. Что такое протокол TCP [Електронний ресурс] // Skypro wiki. – Доступ здійснено 02.05.2025. – Режим доступу до ресурсу: https://sky.pro/wiki/sql/chto-delaet-protokol-tcp/.

3. Аноприенко А.Я. Комбинированные подходы к управлению перегрузками в сетях TCP/IP [Електронний ресурс] / А.Я. Аноприенко, С.В. Рычка // Кафедра ЭВМ ДонНТУ. – Доступ здійснено 02.05.2025. – Режим доступу до ресурсу: https://pmap.donntu.edu.ua/sites/upload/articles/art132.pdf.

4. Сети и телекоммуникации: учебник и практикум для вузов [Електронний ресурс] / К.Е. Самуйлова, И. А. Шалимова, Д.С. Кулябова. – М: Издательство Юрайт, 2025. – 464 с. – Режим доступу до ресурсу: https://studme.org/189009/informatika/protokol#gads_btm.

5. Золотов С. Протоколы Internet / С. Золотов. – СПб: BHV – Санкт-Петербург, 2002 – 304 с.

6. Олифер В. Коипьютерные сети. Принципы, технологии, протоколы: юбилейное издание / Виктор Олифер, Наталья Олифер. – СПб.: Питер, 2020. – 1008 с.

7. Семенов Ю.А. Протоколы Интернет / Ю.А. Семенов. – М.: Горячая линия – Телеком, 2001. – 1100 с.

8. Трофимчук О.М. Аналіз систем автоматичного запиту на повторну передачу / О.М. Трофимчук, В.М. Василенко, С.В. Зайцев // Математичне моделювання в економіці. – №3. – 2018. – С. 41 - 51.

9. Artem Dudich Устройство TCP/Реализация SYN-flood атаки [Електронний ресурс] / Dudich Artem // Habr. – 2023. – Режим доступу до ресурсу: https://habr.com/ru/articles/782728/.

10. Олифер В. Коипьютерные сети. Принципы, технологии, протоколы: 4-е изд. / В. Олифер, Н. Олифер. – СПб.: Питер, 2010. – 944 с.

11. Кульгин М. Оптимизация работы протокола TCP в распределенных сетях [Електронний ресурс] / Максим Кульгин // Сети / Network world. – № 10. – 2009. – Режим доступу до ресурсу: https://www.osp.ru/nets/1999/10/144283.

12. RFC 9293. Wesley E. Transmission Control Protocol (TCP) [Електронний ресурс] / Eddy Wesley // Internet Engineering Task Force (IETF). – 2022. – Режим доступу: https://datatracker.ietf.org/doc/rfc9293/.