6 МЕРЕЖЕВИЙ РІВЕНЬ

У стандартній моделі взаємодії відкритих систем ISO / OSI до мережевого рівня віднесені функції забезпечення передачі даних через складні мережі, які об'єднані довільними зв'язками та побудовані на основі різних локальних і глобальних мережевих технологій.

Протоколи мережевого рівня реалізуються, як правило, у вигляді програмних модулів і виконуються па кінцевих вузлах-комп'ютерах, а також на проміжних вузлах-маршрутизаторах.

6.1 Обмеження мостів та комутаторів

Створення складної, структурованої мережі, яка інтегрує різні базові технології, може здійснюватися й засобами канального рівня: для цього можуть бути використані деякі типи мостів та комутаторів.

Міст або комутатор розділяє мережу на сегменти, локалізує трафік усередині сегмента, що робить лінії зв'язку такими, що розподілюються переважно між станціями даного сегмента. Тим самим мережа розпадається на окремі підмережі, з яких можуть бути побудовані складові мережі досить великих розмірів.

Однак, побудова складних мереж тільки на основі повторювачів, мостів і комутаторів має суттєві обмеження та недоліки.

- По-перше, топологія побудованої мережі *не повинна містити петлі*. Дійсно, міст / комутатор може вирішувати завдання доставки пакета адресату тільки тоді, коли між відправником та отримувачем існує єдиний шлях. У той же час надлишкові зв'язки, які й утворюють петлі, часто є необхідними для кращого балансування навантаження, а також для підвищення надійності мережі за рахунок утворення резервних шляхів.
- По-друге, логічні сегменти мережі, які розташовані між мостами або комутаторами, *слабко ізольовані один від одного*, а саме не захищені від так званих *широкомовних штормів*. Якщо будь-яка станція надсилає широкомовне повідомлення, то це повідомлення передається усім станціям усіх логічних сегментів мережі. Захист від широкомовних штормів в мережах, побудованих на основі мостів і комутаторів, має кількісний, а не якісний характер: адміністратор просто обмежує кількість широкомовних пакетів, яку дозволяється генерувати

деякому вузлу в одиницю часу. Використання механізму віртуальних мереж, реалізованого в багатьох комутаторах, хоча й дозволяє досить гнучко створювати ізольовані по трафіку групи станцій, але при цьому ізолює їх повністю, тобто так, що вузли однієї віртуальної мережі не можуть взаємодіяти з вузлами іншої віртуальної мережі.

- По-третє, в мережах, побудованих на основі мостів і комутаторів, досить *складно вирішується завдання управління трафіком* на основі значення даних, що містяться в пакеті. В таких мережах це можливо тільки за допомогою призначених для користувача фільтрів, для створення яких адміністратору доводиться мати справу з двійковим поданням вмісту пакетів.
- По-четверте, реалізація транспортної підсистеми тільки засобами фізичного та канального рівнів, до яких відносяться мости та комутатори, призводить до *недостатньо гнучкої, однорівневої системи адресації*: в якості адреси призначення використовується *MAC-адреса*, що жорстко пов'язана з мережевим адаптером.
- Нарешті, можливість трансляції протоколів канального рівня є далеко не в усіх типах мостів і комутаторів. До того ж ці можливості є обмеженими. Зокрема, в мережах, що об'єднуються, *повинні збігатися максимально допустимі розміри полів даних в кадрах*, так як мости та комутатори *не підтримують функцію фрагментації кадрів*.

Наявність серйозних обмежень в протоколах канального рівня показує, що побудова на основі засобів цього рівня великих неоднорідних мереж є досить проблематичною. Природне рішення в цих випадках - це залучення можливостей вищого, мережевого рівня.

6.2 Архітектура складеної мережі

Основна ідея введення мережевого рівня полягає в наступному. Мережа у загальному випадку розглядається як сукупність декількох мереж і називається складеною мережею, або інтермережею (internetwork, або internet). Мережі, що входять в складену мережу, називаються підмережами (subnet) або просто мережами (рис. 6.1).

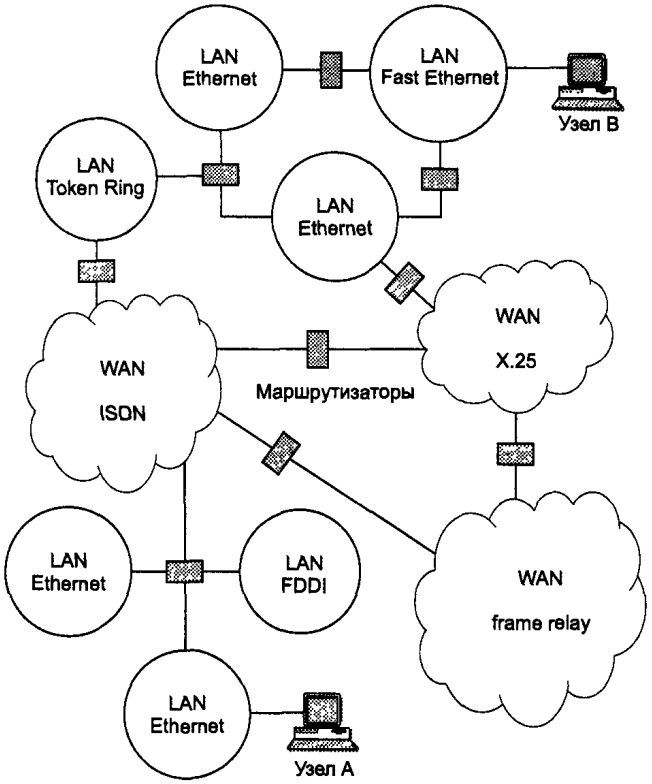

Рис. 6.1. Архітектура складеної мережі

Підмережі з'єднуються між собою маршрутизаторами. Компонентами складеної мережі можуть бути як локальні, так й глобальні мережі. Внутрішня структура кожної мережі на рисунку не показана, так як вона не має значення при розгляді мережевого протоколу. Усі вузли в межах однієї підмережі взаємодіють, використовуючи єдину для них технологію. Так, в складену мережу, що наведена на рисунку, входить кілька мереж різних технологій: локальні мережі Ethernet, Fast Ethernet, Token Ring, FDDI та глобальні мережі frame relay, X.25, ISDN. Кожна з цих технологій здатна забезпечити взаємодію усіх вузлів в своїй підмережі, але не здатна побудувати інформаційний зв'язок між довільно обраними вузлами, які належать різним підмережам, наприклад між вузлом А та вузлом В на рис. 6.1. Отже, для організації взаємодії між будь-якою довільною парою вузлів цієї «великої» складеної мережі потрібні додаткові можливості. Такі можливості й надає мережевий рівень.

Мережевий рівень виступає в якості координатора, що організує роботу всіх підмереж, що розташовані на шляху просування пакету по складеній мережі. Для переміщення даних в межах підмереж мережевий рівень звертається до використовуваних в цих підмережах технологій.

Хоча багато технологій локальних мереж (Ethernet, Token Ring, FDDT, Fast Ethernet та ін.) використовують одну й ту ж систему адресації вузлів на основі MAC-адрес, існує чимало технологій (X.25, ATM, frame relay), в яких застосовуються інші схеми адресації.

Адреси, що привласнені вузлам відповідно до локальних технологій підмереж, називають *локальними*. Щоб мережевий рівень міг виконати своє завдання, йому необхідна власна глобальна система адресації, яка не залежить від способів адресації вузлів в окремих підмережах, яка дозволила б на мережевому рівні універсальним та однозначним способом ідентифікувати будь-який вузол складеної мережі. (Під час обговорення складеної мережі в таких термінах, як «локальна технологія», «локальна адреса» й т.п., слово «локальна» показує, що об'єкт, який характеризується цим словом, відноситься до частини великої складеної (глобальної) мережі). Природним способом формування мережевої адреси є унікальна нумерація усіх підмереж складеної мережі та нумерація усіх вузлів в межах кожної підмережі. Таким чином, мережевою адресою є пара: номер мережі (підмережі) та номер вузла.

Номером вузла може бути або локальна адреса цього вузла (така схема прийнята в стеці IPX / SPX), або деяке число, ніяк не пов'язане з локальною технологією, яка однозначно ідентифікує вузол в межах даної підмережі.

У першому випадку мережева адреса стає залежною від локальних технологій, що обмежує її застосування. Наприклад, мережеві адреси IPX / SPX розраховані на роботу в складених мережах, які об'єднують мережі, в яких використовуються тільки MAC-адреси або адреси аналогічного формату.

Другий підхід є більш універсальним, він є характерним для стека TCP / IP. В обох випадках кожен вузол складеної мережі має поряд зі своєю локальною адресою ще одну - універсальну мережеву адресу. До даних, які надходять на мережевий рівень і які необхідно передати через складену мережу, додається заголовок мережевого рівня. Дані разом із заголовком утворюють пакет. Заголовок пакета мережного рівня має уніфікований формат, який не залежить від форматів кадрів канального рівня тих мереж, які можуть входити в складену мережу, і несе разом з іншою службовою інформацією дані про номер мережі призначення цього пакету. Мережевий рівень визначає маршрут і переміщує пакет між підмережами. Кожен раз, коли пакет мережевого рівня передається з однієї мережі в іншу, він витягується з кадру першої підмережі (звільняється від канального заголовка цієї мережі) та упаковується в кадр (забезпечується новим заголовком) канального рівня наступної підмережі. Інформацію, на основі якої

робиться ця заміна, надають службові поля пакета мережевого рівня. В полі адреси призначення нового кадру вказується локальна адреса наступного маршрутизатора.

ПРИМІТКА

Якщо в підмережі доставка даних здійснюється засобами канального та фізичного рівнів (як, наприклад, в стандартних локальних мережах), то пакети мережевого рівня упаковуються у кадри канального рівня. Якщо у будь-якій підмережі для транспортування повідомлень використовується технологія, що заснована на стеках з великою кількістю рівнів, то пакети мережевого рівня упаковуються в блоки переданих даних найвищого рівня підмережі.

Якщо проводити аналогію між взаємодією різнорідних мереж і листуванням людей з різних країн, то мережева інформація - це загальноприйнятий індекс країни, доданий до адреси листа, написаного на одному із ста мов земної кулі, наприклад на санскриті. І навіть якщо цей лист має пройти через безліч країн, поштові працівники яких не знають санскриту, зрозумілий їм індекс країни-адресата підкаже, через які проміжні країни краще передати лист, щоб він найкоротшим шляхом потрапив у Індію. А вже там працівники місцевих поштових відділень зможуть прочитати точну адресу, яка ідентифікує місто, вулицю, будинок і людину, та доставити лист адресату, так як адреса написана мовою й у формі, прийнятої в даній країні.

Основним полем заголовка мережевого рівня є номер мережі-адресата. У розглянутих раніше протоколах локальних мереж такого поля в кадрах передбачено не було - передбачалося, що усі вузли належать одній мережі. Явна нумерація мереж дозволяє протоколам мережевого рівня складати точну карту міжмережевих зв'язків та обирати раціональні маршрути при будь-якій їх топології, в тому числі альтернативні маршрути, якщо вони є. Цього не вміють робити мости та комутатори.

Крім номера мережі заголовок мережевого рівня повинен містити й іншу інформацію, яка необхідна для успішної передачі пакета з мережі одного типу в мережу іншого типу. Такою інформацією може бути, наприклад:

- номер фрагменту пакета, необхідний для успішного проведення операцій збирання-розбирання фрагментів при з'єднанні мереж з різними максимальними розмірами пакетів;
- час життя пакету, яке вказує, як довго він подорожує по інтермережі. Цей час може використовуватися для знищення пакетів, що «заблукали»;

- якість послуги - критерій вибору маршруту при міжмережевих передачах- наприклад, вузол-відправник може вимагати передати пакет з максимальною надійністю, можливо, за рахунок збільшення часу доставки.

Коли дві або більше мережі організують спільну транспортну службу, то такий режим взаємодії зазвичай називають *міжмережевою взаємодією (internetworking)*.

6.3 Принципи маршрутизації

Найважливішим завданням мережевого рівня є маршрутизація - передача пакетів між двома кінцевими вузлами у складеній мережі.

Розглянемо механізм маршрутизації на прикладі складеної мережі, наведеної на рис. 6.2.

У цій мережі 20 маршрутизаторів (зображених у вигляді пронумерованих квадрантних блоків) об'єднують 18 мереж в загальну мережу; N1, N 2, ..., N18 - це номери мереж. В кожному маршрутизаторі та на кінцевих вузлах A та B функціонують протоколи IP.

До інтерфейсів (портів) маршрутизаторів приєднуються мережі. Кожен інтерфейс маршрутизатора можна розглядати як окремий вузол мережі: він має мережеву адресу та локальну адресу в тій підмережі, яка до нього підключена. Наприклад, маршрутизатор під номером 1 має три інтерфейси, до яких підключені мережі N1, N2, N3. На рисунку мережеві адреси цих портів позначені IP_{11}, IP_{12} та IP_{13}. Інтерфейс IP_{11} є вузлом мережі N1, й отже, у полі номера мережі порту IP_{11} міститься номер N1. Аналогічно інтерфейс IP_{12} - це вузол у мережі N2, а порт IP_{13} - вузол у мережі N3.

Таким чином, маршрутизатор можна розглядати як сукупність декількох вузлів, кожен з яких входить в свою мережу. *Як єдиний пристрій маршрутизатор не має виділеної адреси, ні мережевої, ні локальної.*

У складних складених мережах майже завжди існують кілька альтернативних маршрутів для передачі пакетів між двома кінцевими вузлами. Так, пакет, відправлений від вузла A до вузла B, може пройти через маршрутизатори 17, 12, 5, 4 та 1, або через маршрутизатори 17, 13, 7, 6 та 3. Можна знайти ще кілька маршрутів між вузлами A та B.

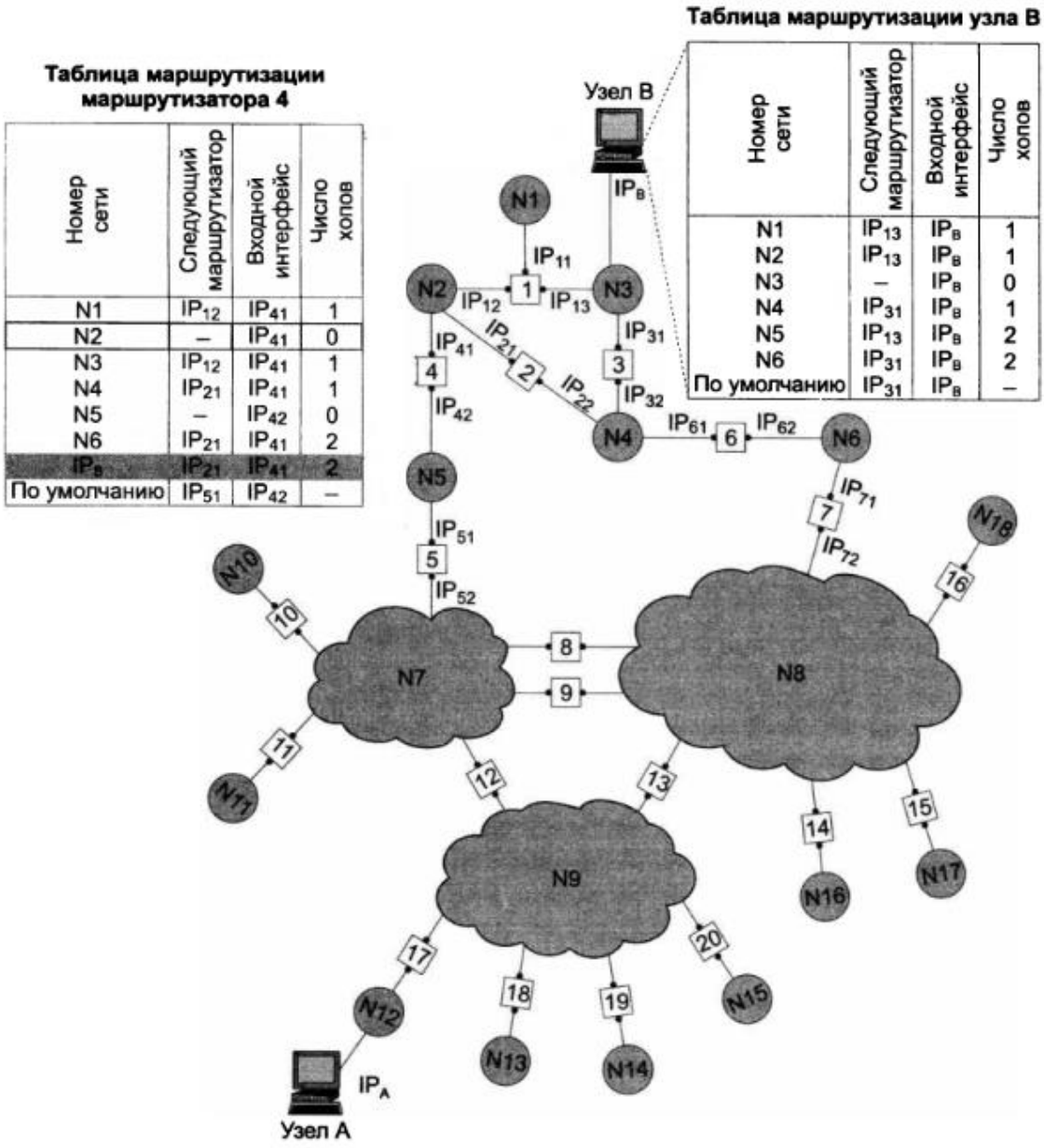

Рис. 6.2. Принципи маршрутизації в складеній мережі

Завдання вибору маршруту з декількох можливих вирішують маршрутизатори, а також кінцеві вузли. Маршрут обирається на підставі інформації про поточну конфігурацію мережі, яка є у наявності в цих пристроях, а також на підставі *критерію вибору маршруту*. Зазвичай в якості критерію береться затримка проходження маршруту окремим пакетом або середня пропускна здатність маршруту для послідовності пакетів. Часто також застосовується найбільш простий критерій, який враховує тільки кількість проміжних маршрутизаторів на маршруті (ретрансляційних ділянок або хопів). Отримана в результаті аналізу інформація про маршрути подальшого просування пакетів поміщається в таблицю маршрутизації.

6.4 Спрощена таблиця маршрутизації

Використовуючи умовні позначення для мережевих адрес маршрутизаторів та номерів мереж, що показані на рис. 6.2, подивимося, як могла б виглядати таблиця маршрутизації, наприклад, в маршрутизаторі 4 (табл. 6.1).

Таблиця 6.1. Таблиця маршрутизації маршрутизатора 4

Адрес назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
N1	IP_{12} (R1)	IP41	1
N2	–	IP41	0 (подсоединена)
N3	IP_{12} (R1)	IP41	1
N4	IP_{21} (R2)	IP41	1
N5	–	IP42	0 (подсоединена)
N6	IP_{21} (R2)	IP21	2
IP_B	IP_{21} (R2)	IP41	2
Маршрут по умолчанию	IP_{51} (R5)	IP42	–

Перший стовпець таблиці містить *адреси призначення пакетів*.

У кожному рядку таблиці поруч з адресою призначення вказується мережева адреса наступного маршрутизатора (точніше, мережева адреса інтерфейса наступного маршрутизатора), на який треба відправити пакет, щоб той пересувався до заданої адреси раціональним маршрутом. Перед тим як відправити пакет до наступного маршрутизатора, поточний маршрутизатор повинен визначити, на який з кількох власних портів (IP_{41} або IP_{42}) він повинен передати даний пакет. Для цього формується третій стовпець таблиці маршрутизації, який містить *мережеві адреси вихідних інтерфейсів*.

Деякі реалізації мережевих протоколів допускають наявність у таблиці маршрутизації відразу *декількох рядків*, що відповідають одній адресі призначення. В цьому випадку при виборі маршруту приймається до уваги стовпець, в який записується відстань до мережі призначення. При цьому відстань вимірюється будь-якою метрикою, що використовується відповідно до заданого в мережевому пакеті критерію. Відстань може вимірюватися часом проходження пакету по лініях зв'язку, різними характеристиками надійності ліній зв'язку на даному маршруті, пропускною здатністю або іншою величиною, що відображає якість даного маршруту по відношенню до заданого критерію. У таблиці 6.1

відстань між мережами вимірюється хопамі. Відстань для мереж, безпосередньо підключених до портів маршрутизатора, приймається рівною 0, проте в деяких реалізаціях відлік відстаней починається з 1.

Коли пакет надходить до маршрутизатора, модуль IP витягує з його заголовка номер мережі призначення і послідовно порівнює його з номерами мереж з кожного рядка таблиці. Рядок з відповідним номером мережі показує найближчий маршрутизатор, на який слід передати пакет. Наприклад, якщо на якийсь порт маршрутизатора 4 надходить пакет, адресований до мережі N6, то згідно таблиці маршрутизації адреса наступного маршрутизатора - IP_{21}, тобто черговим етапом руху даного пакета буде рух до порта 1 маршрутизатора 2.

Найчастіше в якості адреси призначення в таблиці вказується не повна IP-адреса, а тільки номер мережі призначення. Таким чином, для усіх пакетів, які направляються в одну мережу, протокол IP буде пропонувати один маршрут (ми поки не беремо до уваги можливі зміни стану мережі, такі як відмови маршрутизаторів або обриви кабелів). Однак, в деяких випадках виникає необхідність для одного з вузлів мережі визначити *специфічний маршрут*, який буде відрізняєтись від маршруту, що є заданим для усіх інших вузлів мережі. Для цього в таблицю маршрутизації для даного вузла додають окремий рядок, який містить повну IP-адресу цього вузла та відповідну маршрутну інформацію. Такий запис є в таблиці 6.1 для вузла B.

Нехай, наприклад, адміністратор маршрутизатора 4, керуючись міркуваннями безпеки, вирішив, що пакети, які прямують до вузла B (повна адреса IP_5), повинні пересуватися через маршрутизатор 2 (інтерфейс IP_{21}), а не через маршрутизатор 1 (інтерфейс IP_{12}), через який передаються пакети усім іншим вузлам мережі N3. Якщо в таблиці є записи про маршрути як до мережі в цілому, так й до її окремого вузла, то при надходженні пакета, адресованого до даного вузла, маршрутизатор віддасть перевагу специфічному маршруту.

Оскільки пакет може бути адресований до *будь-якої мережі* складеної мережі, може здатися, що кожна таблиця маршрутизації повинна мати записи про усі мережі, що входять в складену мережу. Однак, при такому підході у разі великої мережі обсяг таблиць маршрутизації може виявитися дуже великим, що вплине на час її перегляду, буде вимагати багато місця для зберігання й т.п. Тому на практиці широко відомий шлях зменшення кількості записів в таблиці маршрутизації, заснований на введенні *маршруту за замовчуванням (default route),* який враховує особливості топології мережі. Розглянемо, наприклад, маршрутизатори, що знаходяться на периферії складеної мережі. В таблицях

таких маршрутизаторів достатньо записати номери тільки тих мереж, які безпосередньо приєднані до даного маршрутизатору або розташовані поблизу на тупикових маршрутах. Про усі інші мережі можна в таблиці зробити єдиний запис, що вказує на маршрутизатор, через який пролягає шлях до усіх цих мереж. Такий маршрутизатор називається *маршрутизатором за замовчуванням (default router)*. У цьому прикладі на маршрутизаторі 4 є специфічні маршрути тільки для пакетів, які прямують до мереж N1 - N6. Для усіх інших пакетів, адресованих до мереж N7 - N18, маршрутизатор пропонує продовжити шлях через один й той же порт IP51 маршрутизатора 5, який в даному випадку й є маршрутизатором за замовчуванням.

6.5 Таблиці маршрутизації кінцевих вузлів

Завдання маршрутизації вирішують не тільки проміжні вузли (маршрутизатори), а й кінцеві вузли - комп'ютери. Вирішення цього завдання починається з того, що засобами протоколу IP на кінцевому вузлі визначається куди направлений пакет - до іншої мережі або до вузла даної мережі. Якщо номер мережі призначення збігається з номером даної мережі, це означає, що виконувати маршрутизацію пакета не потрібно. В іншому випадку маршрутизація потрібна.

Структури таблиць маршрутизації кінцевих вузлів та транзитних маршрутизаторів аналогічні. Звернемося знову до мережі, зображеної на рисунку 6.2.

Таблиця 6.2. Таблиця маршрутизації кінцевого вузла B

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
N1	IP_{13} (R1)	IP_B	1
N2	IP_{13} (R1)	IP_B	1
N3	–	IP_B	0
N4	IP_{31} (R3)	IP_B	1
N5	IP_{13} (R1)	IP_B	2
N6	IP_{31} (R3)	IP_B	2
Маршрут по умолчанию	IP_{31} (R3)	IP_B	–

Таблиця маршрутизації кінцевого вузла В, що належить мережі N3, може виглядати як таблиця 6.2. В таблиці IP_B - мережева адреса інтерфейсу комп'ютера В. На підставі цієї таблиці кінцевий вузол В вибирає, на який з двох наявних в локальній мережі N3 маршрутизаторів (R1 або R3) слід надсилати той чи інший пакет.

Кінцеві вузли в ще більшому ступені, ніж маршрутизатори, користуються прийомом маршрутизації за замовчуванням. Хоча вони також в загальному випадку мають в своєму розпорядженні таблицю маршрутизації, її обсяг зазвичай незначний, що пояснюється периферійним розташуванням усіх кінцевих вузлів. Кінцевий вузол часто взагалі працює без таблиці маршрутизації, маючи тільки відомості про адресу маршрутизатора за замовчуванням. При наявності одного маршрутизатора в локальній мережі цей варіант - єдиний можливий для усіх кінцевих вузлів. Але навіть при наявності декількох маршрутизаторів в локальній мережі, коли перед кінцевим вузлом стоїть проблема їх вибору, призначення маршруту за замовчуванням часто застосовується в комп'ютерах для зменшення обсягу його таблиці маршрутизації.

Розглянемо таблицю маршрутизації іншого кінцевого вузла складеної мережі - вузла А (табл. 6.3). Компактний вид таблиці маршрутизації вузла А відображає той факт, що усі пакети, що направляються від вузла А, або не виходять за межі мережі N12, або неодмінно проходять через порт 1 маршрутизатора 17. Цей маршрутизатор і визначений в таблиці маршрутизації у якості маршрутизатора за замовчуванням.

Таблиця 6.3. Таблиця маршрутизації кінцевого вузла А

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
N12	–	IPA	0
Маршрут по умолчанию	$IP_{17.1}$ (R17)	IPA	–

Ще однією відмінністю роботи маршрутизатора від кінцевого вузла є спосіб побудови таблиці маршрутизації. Якщо маршрутизатори, як правило, автоматично створюють таблиці маршрутизації, обмінюючись службовою інформацією, то для кінцевих вузлів таблиці маршрутизації часто створюються вручну адміністраторами та зберігаються у вигляді постійних файлів на дисках.

6.6 Перегляд таблиць маршрутизації

Розглянемо алгоритм перегляду таблиці маршрутизації, що реалізовується у маршрутизаторі протоколом IP. Для цього будемо використовувати таблицю 6.1 та рисунок 6.2.

1. Нехай на один з інтерфейсів маршрутизатора надходить пакет. Протокол IP витягує з пакету IP-адресу призначення (припустимо, адреса призначення IP_B).

2. Виконується перша фаза перегляду таблиці - пошук конкретного маршруту до вузла. IP-адреса (цілком) послідовно рядок за рядком порівнюється з вмістом поля адреси призначення таблиці маршрутизації. Якщо відбувся збіг (як в табл. 6.1), то з відповідного рядку витягуються адреса наступного маршрутизатора (IP_{21}) та ідентифікатор вихідного інтерфейсу (IP_{41}). На цьому перегляд таблиці закінчується.

3. Припустимо тепер, що в таблиці відсутній рядок з адресою призначення IP_B, а відповідно, збіг адрес не відбудеться. У цьому випадку протокол IP переходить до другої фази перегляду - пошуку маршруту до мережі призначення. Із IP-адреси виділяється номер мережі (в нашому прикладі з адреси IP_B виділяється номер мережі N3), і таблиця знову аналізується на предмет збігу номера мережі в будь-якому рядку з номером мережі із пакету. При виявленні збігу адрес (в нашому прикладі збуг відбувся) з відповідного рядку таблиці витягується адреса наступного маршрутизатора (IP_{12}) та ідентифікатор вихідного інтерфейсу (IP_{41}). Перегляд таблиці на цьому завершується.

4. Нарешті, припустимо, що адреса призначення в пакеті була такою, що збіг не відбувся ні на першій, ні на другій фазі перегляду. У такому випадку засобами протоколу IP або вибирається маршрут за замовчуванням (й пакет надсилається на адресу IP_{51}), або, якщо маршрут за замовчуванням відсутній, пакет відкидається. Перегляд таблиці на цьому закінчується.

6.7 Реалізація міжмережевої взаємодії засобами TCP / IP

В даний час стек TCP / IP є найпопулярнішим засобом організації складених мереж. На рис. 6.3 показана частка, яку становить той чи інший стек протоколів у загальносвітовій мережевій базі. До 1996 року безперечним лідером був стек IPX / SPX компанії Novell, але потім картина різко змінилася - стек TCP / IP за темпами зростання кількості встановлення почав набагато випереджати інші

стеки, а з 1998 року вийшов в лідери. Саме тому подальше вивчення функцій мережевого рівня буде проводитися на прикладі стека TCP / IP.

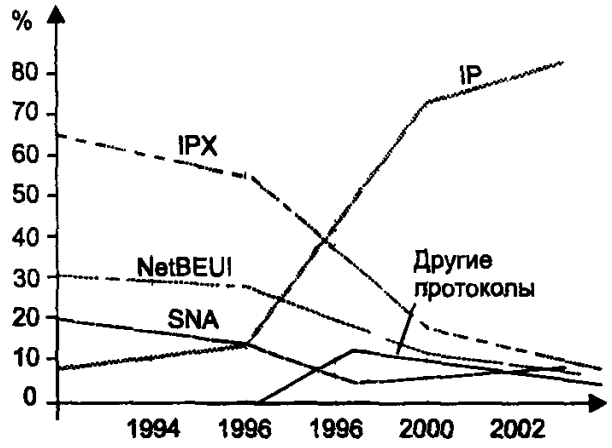

Рис. 6.3. Стек TCP / IP стає основним засобом побудови складних мереж

Структура стека протоколів TCP / IP

Сьогодні стек TCP / IP широко використовується як у глобальних, так й у локальних мережах. Цей стек має ієрархічну структуру, в якій визначено чотири рівні (рис. 6.4).

Прикладной уровень	FTP, Telnet, HTTP, SMTP, SNMP, TFTP
Транспортный уровень	TCP, UDP
Сетевой уровень	IP, ICMP, RIP, OSPF
Уровень сетевых интерфейсов	Не регламентируется

Рис. 6.4. Ієрархічна структура стека TCP/IP

Прикладний рівень стека TCP / IP відповідає трьом верхнім рівням моделі OSI: *прикладному, представницькому та сеансовому*. Він об'єднує послуги, що надаються системою додаткам користувача. За довгі роки застосування в мережах

різних країн та організацій у стеці TCP / IP з’явилась велика кількість протоколів та служб прикладного рівня. До них відносяться такі поширені протоколи, як протокол передачі файлів (File Transfer Protocol, FTP), протокол емуляції терміналу telnet, простий протокол передачі пошти (Simple Mail Transfer Protocol, SMTP), протокол передачі гіпертексту (Hypertext Transfer Protocol, HTTP) та багато інших. Протоколи прикладного рівня розгортаються на хостах.

Транспортний рівень стека TCP / IP може надавати вищому рівню два типи сервісу:

- гарантовану доставку забезпечує протокол управління передачею (Transmission Control Protocol, TCP);
- доставку по можливості, або з максимальними зусиллями, забезпечує протокол дейтаграм, призначених для користувача (User Datagram Protocol, UDP).

Для того щоб забезпечити надійне доставлення даних, протокол TCP передбачає встановлення логічного з'єднання, що дозволяє йому нумерувати пакети, підтверджувати їх прийом квитанціями, у разі втрати організовувати повторні передачі, розпізнавати та знищувати дублікати, доставляти прикладному рівню пакети в тому порядку, в якому вони були відправлені. Завдяки цьому протоколу об'єкти на хості-відправнику та хості-одержувачу можуть підтримувати обмін даними в дуплексному режимі. TCP дає можливість без помилок доставити сформований на одному з комп'ютерів потік байтів на будь-який інший комп'ютер, який входить в складену мережу.

Другий протокол цього рівня, UDP, є найпростішим дейтаграмним протоколом, який використовується тоді, коли завдання надійного обміну даними або взагалі не ставиться, або вирішується засобами більш високого рівня - прикладним рівнем або додатками користувача.

До функцій протоколів TCP та UDP відноситься також виконання ролі сполучної ланки між прилеглими до транспортного рівня прикладним та мережевим рівнями. Транспортний рівень приймає від прикладного протоколу завдання на передачу даних з тією чи іншою якістю прикладному рівню-одержувачу. Протоколи TCP та UDP розглядають нижче розташований мережевий рівень як інструмент, який не дуже надійний, але здатний переміщати пакет у вільної та ризикованої подорожі по складеній мережі. Програмні модулі, що реалізують протоколи TCP та UDP, подібно модулям протоколів прикладного рівня встановлюються на хостах.

Мережевий рівень, що називають також ***рівнем інтернету***, є стрижнем всієї архітектури TCP / IP. Саме цей рівень, функції якого відповідають мережевому

рівню моделі OSI, забезпечує переміщення пакетів в межах складеної мережі, утвореної об'єднанням декількох підмереж. Протоколи мережевого рівня підтримують інтерфейс з вищерозміщеним транспортним рівнем, отримуючи від нього запити на передачу даних по складеній мережі, а також з нижчим рівнем мережевих інтерфейсів, функції якого ми розглянемо далі.

Основним протоколом мережевого рівня є міжмережевий протокол (Internet Protocol IP). До його завдань входить просування пакету між мережами - від одного маршрутизатора до іншого до тих пір, поки пакет не потрапить до мережі призначення. На відміну від протоколів прикладного та транспортного рівнів протокол IP розгортається не тільки на хостах, але й на усіх маршрутизаторах (шлюзах). *Протокол IP* - це дейтаграммний протокол, який працює без встановлення з'єднань за принципом доставки з максимальними зусиллями. Такий тип мережевого сервісу називають також «ненадійним».

До мережевого рівня TCP / IP часто відносять протоколи, які виконують допоміжні функції по відношенню до IP. Це перш за все протоколи маршрутизації RIP та OSPF, які призначені для вивчення топології мережі, визначення маршрутів та складання таблиць маршрутизації, на підставі яких протокол IP переміщує пакети у потрібному напрямку. З цієї ж причини до мережевого рівня можуть бути віднесені протокол міжмережевих керуючих повідомлень (Internet Control Message Protocol, ICMP), призначений для передачі маршрутизатором до джерела відомостей про помилки, що виникли при передачі пакета, та деякі інші протоколи.

Ідеологічною відмінністю архітектури стека TCP / IP від багаторівневої архітектури інших стеків є інтерпретація функцій самого нижнього рівня - ***рівня мережевих інтерфейсів***.

Нагадаємо, що нижні рівні моделі OSI (*канальний та фізичний*) реалізують безліч функцій: доступ до середовища передачі, формування кадрів, узгодження величин електричних сигналів, кодування та синхронізації, а також деякі інші. Усі ці функції складають сутність таких протоколів обміну даними, як Ethernet, PPP та багатьох інших.

У нижнього рівня стека TCP / IP завдання істотно простіше - він відповідає тільки за організацію взаємодії з підмережами різних технологій, що входять в складену мережу.

TCP / IP розглядає будь-яку підмережу, що входить в складену мережу, як засіб транспортування пакетів між двома сусідніми маршрутизаторами.

Завдання організації інтерфейсу між технологією TCP / IP та будь-якою іншою технологією проміжної мережі спрощено можна звести до двох завдань:

- упаковка (інкапсуляція) IP-пакета в одиницю переданих даних проміжної мережі;
- перетворення мережевих адрес в адреси технології даної проміжної мережі.

Такий гнучкий підхід спрощує вирішення проблеми розширення набору підтримуваних технологій. З появою нової популярної технології вона швидко включається в стек TCP / IP шляхом розробки відповідного стандарту, що визначає метод інкапсуляції IP-пакетів в її кадри (наприклад, специфікація RFC 1577, яка визначає роботу протоколу IP через мережі ATM, з'явилася в 1994 році незабаром після прийняття основних стандартів ATM). Так як для кожної технології, яка знову з'являється, розробляються власні інтерфейсні засоби, функції цього рівня не можна визначити раз й назавжди, саме тому нижній рівень стека TCP / IP не регламентується.

Кожен комунікаційний протокол оперує деякою одиницею переданих даних. Назви цих одиниць іноді закріплюються стандартом, а частіше просто визначаються традицією. У стеці TCP / IP за багато років його існування утворилася термінологія в цій області (рис. 6.5).

Рис. 6.5. Назви одиниць даних, що використовуються в TCP / IP

Потоком даних, інформаційним потоком, або просто потоком, називають дані, що надходять від додатків на вхід протоколів транспортного рівня - TCP та UDP.

Протокол TCP «нарізає» з потоку даних ***сегменти***.

Одиницю даних протоколу UDP часто називають ***дейтаграмою***, або датаграммой. Дейтаграмма - це загальна назва для одиниць даних, якими оперують протоколи без встановлення з'єднань. До таких протоколів відноситься й протокол IP, тому його одиницю даних іноді теж називають дейтаграмою, хоча досить часто використовується й інший термін - ***пакет***.

У стеці TCP / IP одиниці даних будь-яких технологій, в які упаковуються IP-пакети для їх подальшої передачі через мережі складеної мережі, прийнято називати також ***кадрами***, або ***фреймами***. При цьому не має значення, яка назва використовується для цієї одиниці даних в технології складеної мережі. Для TCP / IP фреймом є й кадр Ethernet, й комірка ATM, й пакет X.25 у тих випадках, коли вони використовуються у якості контейнера, в якому IP-пакет передається через складену мережу.

Література:

1. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы_2001.
2. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы_2010.

6.8 Адресація в IP-мережах

6.8.1 Типи адрес стеку TCP/IP

У стеці TCP/IP використовуються три типи адрес:

- локальні (апаратні) адреси;
- IP-адреси;
- символьні доменні імена.

В термінології TCP/IP під ***локальною адресою*** розуміється такий тип адреси, який використовується засобами базової технології для доставки даних в межах підмережі, яка є елементом складеної інтермережі. У різних підмережах допустимі різні мережеві технології, різні стеки комунікаційних протоколів. Тому при створенні стека TCP/IP передбачалася наявність різних типів локальних адрес. Якщо підмережею інтермережі є локальна мережа, то *локальна адреса – це MAC-адреса*. MAC-адреса призначається мережевим адаптерам і мережевим інтерфейсам маршрутизаторів. MAC-адреси призначаються виробниками устаткування і є унікальними, так як управляються централізовано. Для усіх існуючих технологій локальних мереж MAC-адреса має формат 6 байт, наприклад 11-A0-17-3D-BC-01. Однак, протокол IP може працювати й над протоколами більш високого рівня, наприклад над протоколом IPX або X.25. В цьому випадку локальними адресами для протоколу IP відповідно будуть адреси IPX та X.25. Слід враховувати, що комп’ютер в локальній мережі може мати кілька локальних адрес навіть при одному мережевому адаптері. Деякі мережеві пристрої взагалі не мають локальних адрес. Наприклад, до таких пристроїв відносяться глобальні порти маршрутизаторів, які призначені для з’єднань типу «точка-точка».

IP-адреси являють собою основний тип адрес, на підставі яких мережевий рівень передає пакети між мережами. Ці адреси складаються з 4 байт, наприклад, у десятковій формі - 109.26.17.100. IP-адреса призначається адміністратором мережі під час конфігурування комп'ютерів та маршрутизаторів. IP-адреса складається з двох частин: номера мережі та номера вузла. Номер мережі може бути обраний адміністратором довільно, або призначений за рекомендацією спеціального підрозділу Internet (Internet Network Information Center, InterNIC), якщо мережа повинна працювати як складова частина Internet. Зазвичай постачальники послуг Internet отримують діапазони адрес у підрозділів InterNIC, а потім розподіляють їх між абонентами. Номер вузла в протоколі IP призначається незалежно від

локальної адреси вузла. Маршрутизатор за визначенням входить одразу в кілька мереж. Тому кожен порт маршрутизатора має власну IP-адресу. Кінцевий вузол (комп'ютер) також може входити в кілька IP-мереж. У цьому випадку комп'ютер повинен мати кілька IP-адрес, по кількості мережевих зв'язків. Таким чином, *IP-адреса характеризує не окремий комп'ютер або маршрутизатор, а одне мережеве з'єднання*.

Символьні доменні імена. Символьні імена в IP-мережах називаються доменними і будуються за ієрархічною ознакою. Складові повного символьного імені в IP-мережах розділяються крапкою і перелічуються в наступному порядку: спочатку просте ім'я кінцевого вузла, потім ім'я групи вузлів (наприклад, ім'я організації), потім ім'я більшої групи (поддомена) і так до імені домена найвищого рівня (наприклад, домену, який об'єднує організації за географічним принципом: UA − Україна, RU − Росія, UK − Великобританія, SU − США). Прикладом доменного імені може бути ім'я ***ekt.sumdu.edu.ua.*** Між доменним ім'ям та IP-адресою вузла немає ніякої алгоритмічної відповідності, тому необхідно використовувати якісь додаткові таблиці або служби, щоб вузол мережі однозначно визначався як по доменному імені, так й по IP-адресі. У мережах TCP/IP використовується спеціальна розподілена служба Domain Name System (DNS), яка встановлює цю відповідність на підставі створюваних адміністраторами мережі таблиць відповідності. Тому *доменні імена* називають також *DNS-іменами*.

6.8.2 Класи IP-адрес

IP-адреса має довжину 4 байти і звичайно записується у вигляді чотирьох чисел, що представляють значення кожного байту в десятковій формі і розділяються точками. Наприклад, 128.10.2.30 − традиційна десяткова форма представлення адреси, а 10000000.00001010.00000010.00011110 − двійкова форма представлення цієї адреси.

Адреса складається з двох логічних частин − номера мережі та номера вузла в мережі. Яка частина адреси відноситься до номеру мережі, а яка − до номеру вузла, визначається значеннями перших бітів адреси. Значення цих бітів є також ознаками того, до якого класу належить та чи інша IP-адреса.

На рис. 6.6 показана структура IP-адрес різних класів.

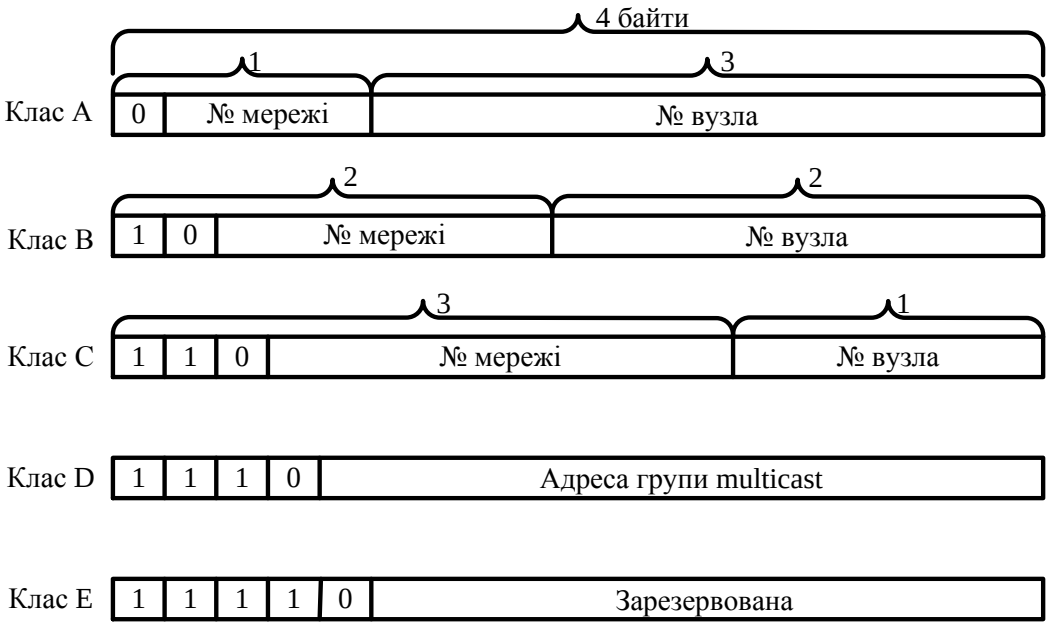

Рис. 6.6. Структура IP-адрес

Якщо адреса починається з 0, то мережу відносять до класу A і номер мережі займає один байт, інші 3 байти інтерпретуються як номер вузла в мережі. Мережі класу A мають номери в діапазоні від 1 до 126. (Номер 0 не використовується, а номер 127 зарезервований для спеціальних цілей, про що буде сказано нижче). Мереж класу A небагато, зате кількість вузлів в них може досягати 2^{24}, тобто 16 777 216 вузлів.

Якщо перші два біти адреси дорівнюють 10, то мережа відноситься до класу B. У мережах класу B під номер мережі та під номер вузла відводиться по 16 біт, тобто по 2 байти. Таким чином, мережа класу B є мережею середніх розмірів з максимальною кількістю вузлів 2^{16}, що складає 65 536 вузлів.

Якщо адреса починається з послідовності 110, то це мережа класу C. У цьому випадку під номер мережі відводиться 24 біти, а під номер вузла − 8 біт. Мережі цього класу найбільш поширені, кількість вузлів в них обмежене 2^8, тобто 256 вузлами.

Якщо адреса починається з послідовності 1110, то вона є адресою класу D і позначає особливу, групову адресу − multicast. Якщо в пакеті в якості адреси призначення вказана адреса класу D, то такий пакет повинні отримати усі вузли, яким призначена ця адреса.

Якщо адреса починається з послідовності 11110, то це означає, що дана адреса відноситься до класу Е. Адреси цього класу зарезервовані для майбутніх застосувань.

У табл. 6.4 наведені діапазони номерів мереж і максимальна кількість вузлів, які відповідають кожному класу мереж.

Таблиця 6.4. Характеристики адрес різного класу

Клас	Перші біти	Найменший номер мережі	Найбільший номер мережі	Максимальна кількість вузлів в мережі
A	0	1.0.0.0	126.0.0.0	2^{24}
B	10	128.0.0.0	191.255.0.0	2^{16}
C	110	192.0.1.0	223.255.255.0	2^{8}
D	1110	224.0.0.0	239.255.255.255	Multicast
E	11110	240.0.0.0	247.255.255.255	Зарезервований

Великі мережі отримують адреси класу А, середні − класу В, а маленькі − класу С.

6.8.3 Особливі IP-адреси

В протоколі IP існує кілька угод про ***особливу інтерпретацію IP − адрес***.

- Якщо уся IP − адреса складається тільки з двійкових нулів, то вона позначає адресу того вузла, який згенерував цей пакет. Цей режим використовується тільки в деяких повідомленнях протоколу ICMP.
- Якщо в полі номера мережі є тільки нулі, то за замовчуванням вважається, що вузол призначення належить тій самій мережі, що й вузол, який відправив пакет.
- Якщо усі двійкові розряди IP-адреси дорівнюють 1, то пакет з такою адресою призначення повинен розсилатися усім вузлам, що знаходяться в тій же мережі, що й джерело цього пакета. Така розсилка називається *обмеженим широкомовним повідомленням (limited broadcast)*.
- Якщо в полі номера вузла призначення є тільки одиниці, то пакет, що має таку адресу, розсилається усім вузлам мережі із заданим номером мережі. Наприклад, пакет з адресою 192.190.21.255 надсилається усім вузлам мережі 192.190.21.0. Така розсилка називається *широкомовним повідомленням (broadcast)*.

При адресації необхідно враховувати ті обмеження, які вносяться особливим призначенням деяких IP-адрес. Так, ні номер мережі, ні номер вузла не може складатися тільки з двійкових одиниць або тільки з двійкових нулів. Звідси випливає, що максимальна кількість вузлів, яка наведена в таблиці для мереж кожного класу, на практиці має бути зменшено на 2. Наприклад, в мережах класу С під номер вузла відводиться 8 біт, які дозволяють задавати 256 номерів: від 0 до 255. Однак, на практиці максимальна кількість вузлів в мережі класу С не може перевищувати 254, так як адреси 0 та 255 мають спеціальне призначення. З цих же міркувань випливає, що кінцевий вузол не може мати адресу 98.255.255.255, оскільки номер вузла в цієї адресі класу А складається лише з двійкових одиниць.

- Особливий сенс має IP-адреса, перший октет якої дорівнює 127. Вона використовується для тестування програм та взаємодії процесів в межах однієї машини. Коли програма надсилає дані за IP-адресою 127.0.0.1, то утворюється так звана «петля». Дані не передаються по мережі, а повертаються модулям верхнього рівня в якості таких, що щойно прийняті. Тому в IP-мережі забороняється назначати машинам IP-адреси, що починаються з числа 127. Ця адреса має назву *loopback*. Можна віднести адресу 127.0.0.0 до внутрішньої мережі модуля маршрутизації вузла, а адресу 127.0.0.1 − до адреси цього модуля у внутрішній мережі. Насправді будь-яка адреса мережі 127.0.0.0 служить для позначення свого модуля маршрутизації, а не тільки 127.0.0.1, наприклад 127.0.0.3.

У протоколі IP немає поняття широкомовного в тому сенсі, в якому воно використовується в протоколах канального рівня локальних мереж, коли дані повинні бути доставлені абсолютно усім вузлам. Як обмежена широкомовна IP-адреса, так й широкомовна IP-адреса мають межі поширення в інтермережі. Вони обмежені або мережею, до якої належить вузол-джерело пакета, або мережею, номер якої зазначений в адресі призначення. Тому розподіл мережі за допомогою маршрутизаторів на частини локалізує широкомовний шторм межами однієї зі складових частин загальної мережі через те, що немає способу адресувати пакет одночасно усім вузлам усіх мереж складеної мережі.

Форма запису групової IP-адреси (*multicast*) означає, що даний пакет повинен бути доставлений одразу декільком вузлам, які утворюють групу з номером, вказаним у полі адреси. Вузли самі ідентифікують себе, тобто визначають, до якої з груп вони відносяться. Один й той же вузол може входити в кілька груп. Члени будь-якої групи *multicast* не обов’язково повинні належати одній мережі. У загальному випадку вони можуть розподілятися по абсолютно різних мережах, які знаходяться одна від одної на довільній кількості хопів. Групова адреса не ділиться

на поля номера мережі та вузла й обробляється маршрутизатором особливим чином.

Основне призначення multicast-адрес − поширення інформації за схемою «один-до-багатьох». Хост, який хоче передавати одну й ту ж інформацію багатьом абонентам, за допомогою спеціального протоколу IGMP (Internet Group Management Protocol) повідомляє про створення в мережі нової широкомовної групи з певною адресою. Машрутизатори, які підтримують широкомовну групу, поширюють інформацію про створення нової групи в мережах, підключених до портів цього маршрутизатора. Хости, які хочуть приєднатися до новостворюваної багатомовної групи, повідомляють про це своїм локальним маршрутизаторам і ті передають цю інформацію хосту, який є ініціатором створення нової групи.

Щоб маршрутизатори могли автоматично поширювати пакети з адресою multicast по складеній мережі, необхідно використовувати в кінцевих маршрутизаторах модифіковані протоколи обміну маршрутною інформацією, такі як, наприклад, MOSPF (Multicast OSPF, аналог OSPF).

Групова адресація призначена для економічного поширення в Internet або великої корпоративної мережі аудіо- або відеопрограм, призначених одразу великій аудиторії слухачів або глядачів. Якщо такі засоби знайдуть широке застосування (зараз вони утворюють в основному невеликі експериментальні острівці у мережі Internet), то Internet зможе створити серйозну конкуренцію радіо та телебаченню.

6.8.4 Використання масок в IP-адресації

Традиційна схема ділення IP-адреси на номер мережі та номер вузла заснована на понятті класу, який визначається значеннями кількох перших бітів адреси. Саме тому, що перший байт адреси 185.23.44.206 потрапляє в діапазон 128 − 191, ми можемо сказати, що ця адреса відноситься до класу B, а значить, номером мережі є перші два байти, доповнені двома нульовими байтами − 185.23.0.0, а номером вузла − 0.0.44.206.

А якщо використовувати будь-який іншу ознаку, за допомогою якої можна було б більш гнучко встановлювати межу між номером мережі та номером вузла? В якості такої ознаки зараз набули широкого поширення маски.

Маска − це число, яке використовується в парі з IP-адресою, причому двійковий запис маски містить одиниці в тих розрядах, які повинні в IP-адресі інтерпретуватися як номер мережі.

Оскільки номер мережі є цілісною частиною адреси, одиниці в масці також повинні представляти безперервну послідовність.

Для стандартних класів мереж маски мають таке значення:

- клас A - 11111111.00000000.00000000.00000000 (255.0.0.0);
- клас B - 11111111.11111111.00000000.00000000 (255.255.0.0);
- клас C - 11111111.11111111.11111111.00000000 (255.255.255.0).

ПРИМІТКА

Для запису масок використовуються й інші формати, наприклад, зручно інтерпретувати значення маски, записаної в шістнадцятковому коді: FF.FF.00.00 − маска для адрес класу B. Часто зустрічається й таке позначення 185.23.44.206/16 − цей запис говорить про те, що маска для цієї адреси містить 16 одиниць або що в зазначеній IP-адресі під номер мережі відведено 16 двійкових розрядів.

Шляхом додавання до кожної IP-адреси маски, можна відмовитися від понять класів адрес та зробити більш гнучкою систему адресації.

Наприклад, якщо розглянуту вище адресу 185.23.44.206 асоціювати з маскою 255.255.255.0, то номером мережі буде 185.23.44.0, а не 185.23.0.0, як це визначено системою класів. Номером вузла у такому випадку буде не 0.0.44.206, а 0.0.0.206.

У масках кількість одиниць в послідовності, яка визначає границю номера мережі, не обов'язково має бути кратною 8, щоб повторювати поділ адреси на байти. Нехай, наприклад, для IP − адреси 129.64.134.5 вказана маска 255.255.128.0, тобто в двійковому виді:

- IP-адреса − 129.64.134.5 − 10000001.01000000.10000110.00000101.
- Маска − 255.255.128.0 − 11111111.11111111.10000000.00000000.

Якщо ігнорувати маску, то відповідно до системи класів адреса 129.64.134.5 відноситься до класу B, а значить, номером мережі є перші 2 байти − 129.64.0.0, а номером вузла − 0.0.134.5.

Якщо використовувати для визначення границі номера мережі маску, то 17 послідовних одиниць в масці, «накладені» на IP-адресу, визначають:

- номер мережі:
  10000001.01000000.10000000.00000000 (129.64.128.0);
- номер вузла:
  00000000.00000000.00000110.00000101 (0.0.6.5).

Механізм масок широко поширений в IP − маршрутизації, причому маски можуть використовуватися для різних цілей.

- За їх допомогою адміністратор може структурувати свою мережу, не вимагаючи від постачальника послуг додаткових номерів мереж.
- На основі цього механізму постачальники послуг можуть об'єднувати адресні простори кількох мереж шляхом введення так званих «префіксів» з метою зменшення обсягу таблиць маршрутизації та підвищення за рахунок цього продуктивності маршрутизаторів.

6.8.5 Використання масок для структуризації мережі

Алгоритм маршрутизації ускладнюється, коли в систему адресації вузлів вносяться додаткові елементи − маски. Якою є причина відмови від методу адресації, заснованого на класах, який добре себе зарекомендував протягом багатьох років? Таких причин декілька, й одна з них − потреба в структуризації мереж.

Часто адміністратори мереж відчувають незручності через те, що кількість централізовано виділених їм номерів мереж є недостатньою для того, щоб структурувати мережу належним чином, наприклад розмістити усі комп'ютери, що рідко взаємодіють між собою, по різним мережам. У такій ситуації можливі два шляхи. Перший з них пов'язаний з отриманням від InterNIC або постачальника послуг Internet додаткових номерів мереж. Другий спосіб, який застосовується частіше, пов'язаний з використанням технології масок, яка дозволяє розділяти одну мережу на кілька підмереж.

Припустимо, адміністратор отримав в своє розпорядження адресу класу B: 129.44.0.0. Він може організувати мережу з великою кількістю вузлів, номера яких він може брати з діапазону 0.0.0.1 − 0.0.255.254 (з урахуванням того, що адреси лише з нулів та лише з одиниць мають спеціальне призначення та не можуть використовуватись для адресації вузлів). Однак, адміністратору не потрібна одна велика неструктурована мережа. Виробнича необхідність диктує адміністраторові інше рішення, відповідно до якого мережа повинна бути розділена на три окремих підмережі, при цьому трафік в кожної підмережі повинен бути надійно локалізованим. Це дозволить легше діагностувати мережу та проводити в кожній з підмереж особливу політику безпеки.

Подивимося, як вирішується ця проблема шляхом використання механізму масок.

Отже, номер мережі, який адміністратор отримав від постачальника послуг, − 129.44.0.0 (10000001.00101100.00000000.00000000).

В якості маски було обрано значення

255.255.192.0 (11111111.11111111.11000000.00000000).

Після накладення маски на цю адресу кількість розрядів, які інтерпретуються як номер мережі, збільшилася з 16 (стандартна довжина поля номера мережі для класу B) до 18 (кількість одиниць в масці), тобто адміністратор отримав можливість використовувати для нумерації підмереж два додаткових біти. Це дозволяє йому зробити з одного, централізовано заданого йому номера мережі, чотири:

- 129.44.0.0 (10000001.00101100.00000000.00000000);
- 129.44.64.0 (10000001.00101100.01000000.00000000);
- 129.44.128.0 (10000001.00101100.10000000.00000000);
- 129.44.192.0 (10000001.00101100.11000000.00000000).

Два додаткових останніх біти у номері мережі часто інтерпретуються як номери підмереж (subnet). Тоді чотири перелічені вище підмережі мають номера 0 (00), 1 (01), 2 (10) та 3 (11) відповідно.

ПРИМІТКА

Деякі програмні та апаратні маршрутизатори не підтримують номера підмереж, які складаються або тільки з одних нулів, або тільки з одних одиниць. Наприклад, для деяких типів обладнання номер мережі 129.44.0.0 з маскою 255.255.192.0, що використаний в розглянутому прикладі, виявиться неприпустимим, оскільки в цьому випадку розряди в полі номера підмережі мають значення 00. З аналогічних міркувань неприпустимим може виявитися й номер мережі 129.44.192.0 з тим же значенням маски. Тут номер підмережі складається тільки з одиниць. Однак, більш сучасні маршрутизатори не мають цих обмежень. Тому, приймаючи рішення про використання механізму масок, необхідно з'ясувати характеристики того обладнання, яке ви маєте в своєму розпорядженні, щоб відповідним чином налаштувати маршрутизатори та комп'ютери мережі.

У результаті використання масок була запропонована наступна схема розподілу адресного простору (рис. 6.7).

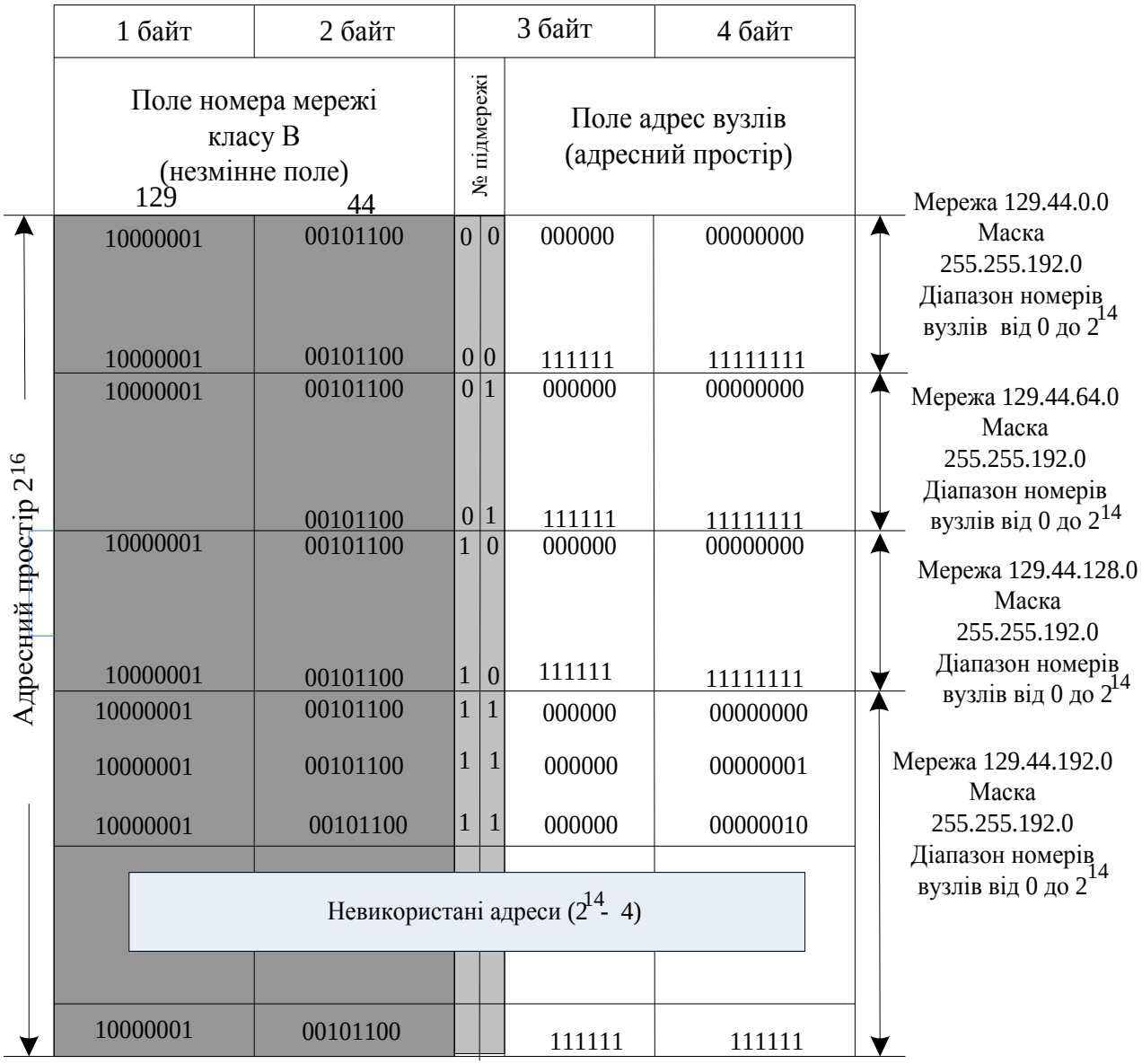

Рис. 6.7. Розподіл адресного простору мережі класу В 129.44.0.0 на чотири рівні частини шляхом використання масок однакової довжини 255.255.192.0

Мережа, яка отримана у результаті проведеної структуризації, показана на рисунку 6.8. Весь трафік у внутрішню мережу 129.44.0.0, що направляється із зовнішньої мережі, надходить через маршрутизатор М1. З метою структуризації інформаційних потоків у внутрішній мережі встановлений додатковий маршрутизатор М2.

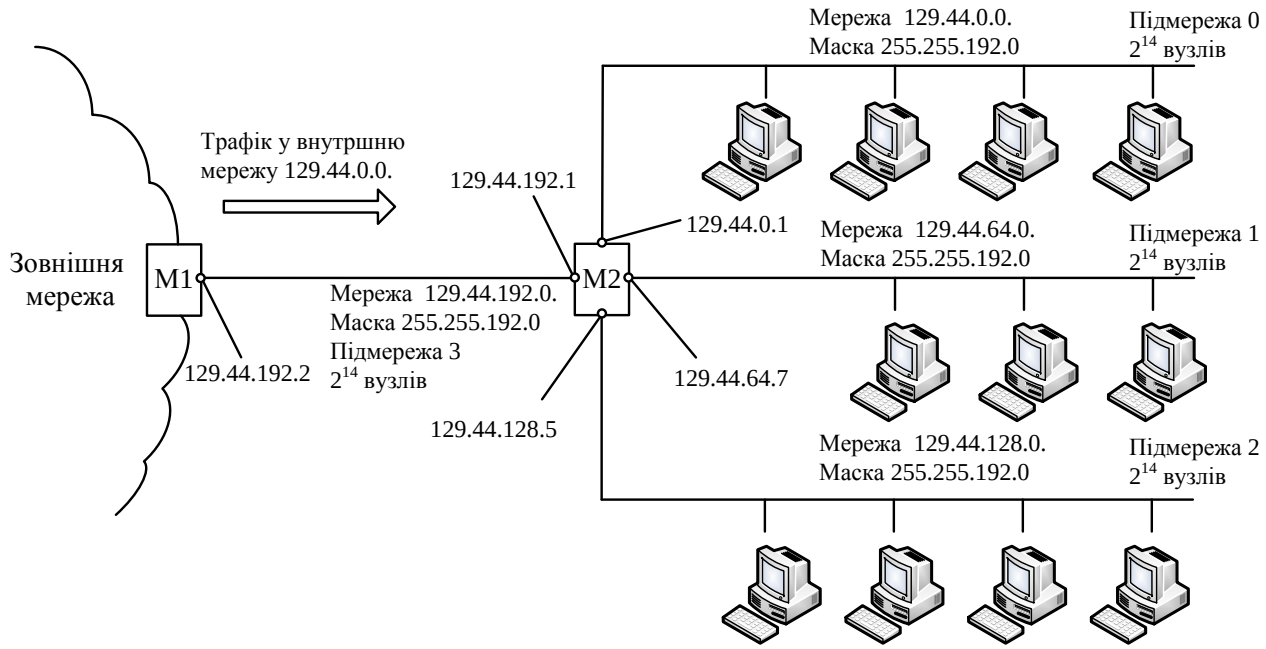

Рис. 6.8. Маршрутизація з використанням масок однакової довжини

Усі вузли були розділені на три різні мережі, яким були призначені номери 129.44.0.0, 129.44.64.0 та 129.44.128.0 і маски однакової довжини − 255.255.192.0. Кожна з новостворених мереж була підключена до відповідно сконфігурованих портів внутрішнього маршрутизатора M2. Крім того, ще одна мережа (номер 129.44.192.0, маска 255.255.192.0) була виділена для створення з'єднання між зовнішнім та внутрішнім маршрутизаторами.

Особливо відзначимо, що в цій мережі для адресації вузлів були зайняті всього дві адреси 129.44.192.1 (порт маршрутизатора M2) та 129.44.192.2 (порт маршрутизатора M1). Ще дві адреси 129.44.192.0 та 129.44.192.255 є особливими адресами. Отже, величезна кількість вузлів (2^{14} - 4) в цій підмережі ніяк не використовуються.

Ззовні мережа, як й раніше, виглядає як єдина мережа класу B, а на місцевому рівні − це повноцінна складена мережа, яка містить три окремі мережі. Загальний трафік розподілюється місцевим маршрутизатором M2 між цими мережами відповідно до таблиці маршрутизації. (Зауважимо, що поділ великої мережі, яка має одну адресу старшого класу, наприклад A або B, за допомогою масок має ще одну перевагу в порівнянні з використанням кількох адрес стандартних класів для мереж меншого розміру, наприклад мереж класу C. Він дозволяє приховати внутрішню структуру мережі підприємства від зовнішнього спостереження й тим самим підвищити її безпеку).

Розглянемо, як змінюється робота модуля IP, коли стає необхідним враховувати наявність масок.

По-перше, в кожному записі таблиці маршрутизації з'являється нове поле − поле маски.

По-друге, змінюється алгоритм визначення маршруту по таблиці маршрутизації. Після того як IP-адреса зчитується з отриманого IP-пакета, необхідно визначити адресу наступного маршрутизатора, на який треба передати пакет з цією адресою. Модуль IP послідовно переглядає усі записи таблиці маршрутизації. З кожним записом виконуються наступні дії.

- Маска M, що міститься в даному записі, накладається на IP-адресу вузла призначення, яка зчитана з пакету.
- Отримане в результаті число є номером мережі призначення пакету, що оброблюється. Воно порівнюється з номером мережі, який міститься в даному записі таблиці маршрутизації.
- Якщо номери мереж збігаються, то пакет передається маршрутизатору, адреса якого міститься у відповідному полі даного запису.

Тепер розглянемо цей алгоритм на прикладі маршрутизації пакетів в мережі, зображеної на рис. 6.8. Усі маршрутизатори зовнішньої мережі, зустрівши пакети з адресами, що починаються з 129.44, інтерпретують їх як адреси класу B і направляють по маршрутам, які ведуть до маршрутизатора M1. Маршрутизатор M1 в свою чергу направляє весь вхідний трафік мережі 129.44.0.0 на маршрутизатор M2, а саме на його порт 129.44.192.1.

Маршрутизатор M2 обробляє усі пакети, що надійшли до нього, відповідно до таблиці маршрутизації (табл. 6.4).

Таблиця 6.4. Таблиця маршрутизатора M2 в мережі з масками однакової довжини

Номер мережі	**Маска**	**Адреса наступного маршрутизатора**	**Адрес порта**	**Відстань**
129.44.0.0	255.255.192.0	129.44.0.1	129.44.0.1	Підключена
129.44.64.0	255.255.192.0	129.44.64.7	129.44.64.7	Підключена
129.44.128.0	255.255.192.0	129.44.128.5	129.44.128.5	Підключена
129.44.192.0	255.255.192.0	129.44.192.1	129.44.192.1	Підключена
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	Підключена

Перші чотири записи в таблиці відповідають внутрішнім підмережам, безпосередньо підключеним до портів маршрутизатора М2.

Запис 0.0.0.0 з маскою 0.0.0.0 відповідає маршруту за замовчуванням. Дійсно, будь-яка адреса в отриманому пакеті після накладення на неї маски 0.0.0.0 утворює адресу мережі 0.0.0.0, що збігається з адресою, зазначеною в записі. Маршрутизатор виконує порівняння з адресою 0.0.0.0 в останню чергу, в тому випадку коли отримана адреса не співпала з жодною адресою в таблиці маршрутизації, яка б відрізнялася від 0.0.0.0. Записів з адресою 0.0.0.0 у таблиці маршрутизації може бути кілька. У цьому випадку маршрутизатор передає пакет по усіх вищевказаних маршрутах.

Нехай, наприклад, з маршрутизатора М1 на порт 129.44.192.1 маршрутизатора М2 надходить пакет з адресою призначення 29.44.78.200. Модуль IP починає послідовно переглядати усі рядки таблиці, до тих пір поки не знайде збіги номера мережі в адресі призначення й в рядку таблиці. Маска з першого рядка 255.255.192.0 накладається на адресу 129.44.78.200, в результаті чого визначається номер мережі 129.44.64.0.

У двійковому вигляді ця операція виглядає наступним чином:

10000001.00101100.01001110.11001000

11111111.11111111.11000000.00000000

- -

10000001.00101100.01000000.00000000

Отриманий номер 129.44.64.0 порівнюється з номером мережі в першому рядку таблиці 129.44.0.0. Оскільки вони не збігаються, то відбувається перехід до наступного рядка. Тепер зчитується маска з другого рядка (в даному випадку вона має таке ж значення, але в загальному випадку це зовсім не обов’язково) і накладається на адресу призначення пакета 129.44.78.200. Зрозуміло, що через збіг довжини масок буде отриманий той же номер мережі 129.44.64.0. Цей номер збігається з номером мережі в другому рядку таблиці, й, відповідно, маршрут для даного пакета знайдений. Пакет повинен бути відправлений на порт маршрутизатора 129.44.64.7 в мережу, яка безпосередньо підключена до даного маршрутизатора.

Розглянемо ще один приклад. IP − адреса 129.44.141.15 (10000001.00101100.10001101.00001111), яка при використанні класів поділяється на номер мережі 129.44.0.0 та номер вузла 0.0.141.15, тепер, при використанні маски 255.255.192.0, буде інтерпретуватися як: 129.44.128.0 − номер мережі, 0.0.13.15 − номер вузла.

6.9 Формування IP-пакета. Фрагментація дейтаграм

Основу транспортних засобів стека протоколів TCP/IP становить ***протокол міжмережевої взаємодії (Internet Protocol, IP)***. Він забезпечує передачу дейтаграм від відправника до одержувачів через об'єднану систему комп'ютерних мереж.

Назва даного протоколу − Intrenet Protocol − відображає його сутність: він повинен передавати пакети між мережами. У кожній черговій мережі, що знаходиться на шляху переміщення пакета, протокол IP викликає засоби транспортування, що прийняті в цій мережі. За їх допомогою цей пакет надходить до маршрутизатора, який буде пересилати пакет до наступної мережі, або безпосередньо до вузла-одержувача.

Протокол IP відноситься до *протоколів без встановлення з'єднань*. Перед протоколом IP не ставиться завдання надійної доставки повідомлень від відправника до одержувача. Протокол IP обробляє кожен IP-пакет як незалежну одиницю, що не має зв'язку з будь-якими іншими IP - пакетами. У протоколі IP немає механізмів, які зазвичай застосовуються для збільшення достовірності кінцевих даних: відсутнє квитування − обмін підтвердженнями між відправником та отримувачем, немає процедури упорядкування, повторних передач або інших подібних функцій. Якщо під час просування пакета відбулася будь-яка помилка, то протокол IP за своєю ініціативою нічого не робить для виправлення цієї помилки. Наприклад, якщо на проміжному маршрутизаторі пакет був відкинутий через закінчення часу життя або через помилку в контрольній сумі, то модуль IP не намагається повторно надіслати зіпсований або втрачений пакет. Усі питання забезпечення надійності доставки даних по складеній мережі в стеці TCP/IP вирішує протокол TCP, який працює безпосередньо над протоколом IP. Саме TCP - протокол організовує повторну передачу пакетів, коли в цьому виникає необхідність.

Важливою особливістю протоколу IP, що відрізняє його від інших мережевих протоколів (наприклад, від мережевого протоколу IPX), є його здатність виконувати динамічну фрагментацію пакетів при передачі їх між мережами з різними максимально допустимими значеннями полів даних кадрів MTU. Властивість фрагментації багато в чому сприяла тому, що протокол IP зміг зайняти домінуючі позиції в складних складених мережах.

Є прямий зв'язок між функціональною складністю протоколу і складністю заголовка пакетів, які цей протокол використовує. Це пояснюється тим, що основні службові дані, на підставі яких протокол виконує ту чи іншу дію,

переносяться між двома модулями, які реалізують цей протокол на різних машинах, саме в полях заголовків пакетів. Знання призначення кожного поля заголовка IP-пакета пояснюють усі основні режими роботи протоколу IP.

6.9.1 Структура пакета IPv4

IP-пакет складається із заголовка та поля даних. Заголовок, як правило, має довжину 20 байт, і має наступну структуру (рис. 6.9).

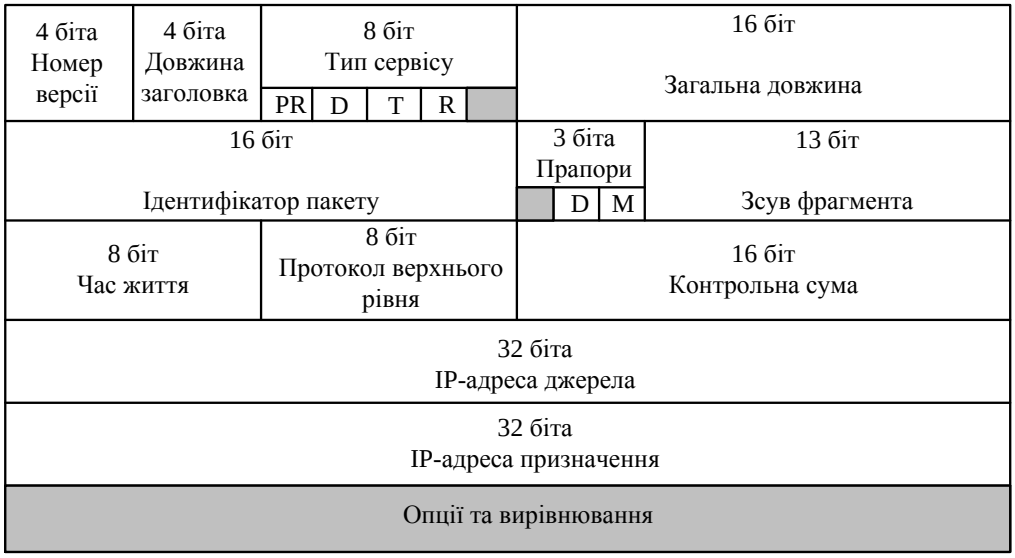

Рис. 6.9 Структура заголовка IP-пакета

Поле *Номер версії (Version),* що містить 4 біта, вказує версію протоколу IP. Зараз повсюдно використовується версія 4 (IPv4), і почався перехід на версію 6 (IPv6).

Поле *Довжина заголовка (IHL)* IP-пакета містить 4 біта й вказує на значення довжини заголовка, що вимірюється 32-бітовими словами. Зазвичай заголовок має довжину в 20 байт (п'ять 32-бітових слів). Але при збільшенні обсягу службової інформації довжина заголовка може бути збільшена за рахунок використання додаткових байтів у полі Опції (*IP Options*). Найбільший заголовок містить 60 октетів. (Але не менше ніж 5 слів по 32 біти кожне слово),

Поле *Тип сервису (Type of Service)* містить один байт і задає пріоритетність пакету та критерій вибору маршрута. Перші три біта цього поля утворюють підполе *пріоритету* пакета *(Precedence).* Пріоритет може мати значення від найнижчого − 0 (звичайний пакет) до найвищого − 7 (пакет керуючої інформації). Маршрутизатор

і комп’ютери можуть брати до уваги пріоритет пакета та обробляти більш важливі пакети в першу чергу.

Поле *Тип сервісу* містить також три біта, що визначають критерій вибору маршруту. Реально вибір здійснюється між трьома альтернативами: не значною затримкою, високою достовірністю та високою пропускною здатністю. Встановлений біт D (delay) говорить про те, що маршрут повинен вибиратися для мінімізації затримки доставки даного пакету, біт T − для максимізації пропускної здатності, а біт R − для максимізації надійності доставки. У багатьох мережах поліпшення одного з цих параметрів пов’язане з погіршенням іншого, крім того, обробка кожного з них вимагає додаткових обчислювальних витрат. Тому рідко, коли має сенс встановлювати одночасно хоча б два з цих трьох критеріїв вибору маршруту. Зарезервовані біти (6 та 7) мають нульове значення.

Поле *Загальна довжина (Total Length)* займає 2 байта і означає загальну довжину пакета з урахуванням заголовка й поля даних. Максимальна довжина пакета обмежена розрядністю поля, яке визначає цю величину, й становить 65 535 байт. Проте в більшості хост-комп’ютерів і мереж такі великі пакети не використовуються. При передачі по мережах різного типу довжина пакета вибирається з урахуванням максимальної довжини пакету протокола нижнього рівня, який пересуває IP-пакети по мережі. Якщо це кадри Ethernet, то вибираються пакети з максимальною довжиною в 1500 байт, які розміщуються в поле даних кадру Ethernet. У стандарті передбачається, що усі хости повинні бути готові приймати пакети довжиною до 576 байт. Й не важливо надходять пакети цілком або по фрагментах. Хостам рекомендується відправляти пакети розміром більш ніж 576 байт тільки якщо вони впевнені, що приймаючий пристрій або проміжна мережа готові обслуговувати пакети такого розміру.

Поле *Ідентифікатор пакету (Identification)* займає 2 байта і використовується для розпізнавання пакетів, які утворилися шляхом фрагментації вихідного пакета. Усі фрагменти повинні мати однакове значення цього поля. Призначається ідентифікатор пакету вузлом, що відправляє пакет, для коректного збирання фрагментів пакету приймачем.

Поле *Прапори (Flags)* займає 3 біта і містить ознаки, які пов’язані з фрагментацією. Встановлений біт DF (Do not Fragment) забороняє маршрутизатору фрагментувати даний пакет, а встановлений біт MF (More Fragments) говорить про те, що даний пакет є проміжним (не останнім) фрагментом. Ще один біт є зарезервованим і повинен мати нульове значення.

Поле *Зсув фрагмента (Fragment Offset)* займає 13 біт і задає зсув в байтах поля даних цього пакета від початку загального поля даних вихідного пакета, що фрагментується. По суті це місцезнаходження фрагмента у вихідному пакеті. Перший фрагмент має нульовий зсув. Використовується при складанні / розбиранні фрагментів пакетів при передачі їх між мережами з різними величинами *MTU (Maximum Transmission Unit - максимальний розмір поля даних)*. Зсув повинен бути кратним 8 байтам.

Поле *Час життя (TTL - Time to Live)* займає один байт та означає граничний термін, протягом якого пакет може пересуватися по мережі. Час життя даного пакета вимірюється в секундах і задається джерелом передачі. На маршрутизаторах та в інших вузлах мережі після закінчення кожної секунди з поточного часу життя віднімається одиниця. Одиниця віднімається й у тому випадку, коли час затримки буде менше секунди. Оскільки сучасні маршрутизатори рідко оброблюють пакет довше, ніж за одну секунду, то час життя можна вважати рівним максимальної кількості вузлів, які дозволено пройти даному пакету перед тим, як він досягне місця призначення. Тому параметр TTL можна використовувати для підрахування кількості хопів (проміжних вузлів). Якщо параметр часу життя стане нульовим раніше, ніж одержувач отримає пакет, то цей пакет буде знищений. При TTL = 1 пакет не може вийти за межі мережі відправника. Час життя можна розглядати як годинниковий механізм самознищення. Значення цього поля змінюється при обробці заголовка IP-пакета.

Ідентифікатор *Протокол верхнього рівня (Protocol)* займає один байт і вказує, якому протоколу верхнього рівня належить інформація, що розміщена в поле даних пакета. Наприклад, це можуть бути сегменти протоколу TCP, дейтаграми UDP, пакети ICMP або OSPF. Значення ідентифікаторів для різних протоколів наводяться в документі RFC «Assigned Numbers».

Контрольна сума (Header Checksum) займає 2 байта і розраховується тільки по заголовку. Оскільки деякі поля заголовка змінюють своє значення в процесі передачі пакета по мережі (наприклад, час життя), контрольна сума перевіряється та повторно розраховується при кожній обробці IP-заголовка. Контрольна сума − 16 біт − розраховується як доповнення до суми усіх 16 - бітових слів заголовка. При обчисленні контрольної суми значення самого поля «контрольна сума» встановлюється в нуль. Якщо контрольна сума буде невірною, то пакет буде відкинутий, як тільки помилка буде виявлена.

Поля *IP-адреса джерела (Source IP Address)* та *IP-адреса призначення (Destination IP Address)* мають однакову довжину − 32 біта − та однакову структуру.

Поле *Опції (IP Options)* є необов'язковим і використовується зазвичай тільки при налагодженні мережі. Механізм опцій надає функції управління, які є необхідними в певних ситуаціях, проте він не потрібен при звичайних комунікаціях. Це поле складається з декількох підполів, кожне з яких може бути одним з восьми визначених типів. У цих підполях можна вказувати точний маршрут проходження маршрутизаторів, реєструвати маршрутизатори, через які проходить пакет, розміщувати дані системи безпеки, а також часові позначки. Так як кількість підполів може бути довільною, то в кінці поля Опції повинно бути додано кілька байт для вирівнювання заголовка пакета по 32-бітної границі.

Поле *Вирівнювання (Padding)* використовується для того, щоб переконатися в тому, що IP-заголовок закінчується на 32-бітної границі. Вирівнювання здійснюється нулями.

6.9.2 Фрагментація дейтаграм

Протокол IP дозволяє виконувати фрагментацію пакетів, що надходять на вхідні порти маршрутизаторів.

Слід розрізняти фрагментацію повідомлень у вузлі - відправнику та динамічну фрагментацію повідомлень в транзитних вузлах мережі − маршрутизаторах. Практично у всіх стеках протоколів є протоколи, які відповідають за фрагментацію повідомлень прикладного рівня на такі частини, які укладаються в кадри канального рівня. У стеці TCP/IP це завдання вирішує протокол TCP, який розбиває потік байтів, що передається йому з прикладного рівня, на повідомлення потрібного розміру (наприклад, на 1460 байт для протоколу Ethernet). Тому вузол-відправник не використовує можливості протоколу IP по фрагментації пакетів.

А при необхідності передати пакет в наступну мережу, для якої розмір пакета є занадто великим, IP-фрагментація стає необхідною.

Значення MTU для різних технологій перелічені в табл. 6.5.

З таблиці 6.5 видно, що значення MTU для найбільш популярних технологій істотно відрізняються. А це значить, що в сучасній мережі, якій властива гетерогенність, фрагментація не є рідкісним явищем.

В функції рівня IP входить розбиття занадто довгого для конкретного типу складеної мережі повідомлення на більш короткі пакети зі створенням відповідних службових полів, потрібних для подальшого складання фрагментів у вихідне повідомлення.

Таблиця 6.5 - Значення MTU для різних технологій

Технологія	**Максимальний розмір поля даних (MTU)**
DIX Ethernet	1500 байт
Ethernet 802.3	1492 байт
Token Ring (IBM, 16 Мбит/с)	17914 байт
Token Ring (802.5, 4 Мбит/с)	4464 байт
FDDI	4352 байт
X.25	576 байт

У більшості типів локальних та глобальних мереж значення MTU, тобто максимальний розмір поля даних, в яке повинен інкапсулювати свій пакет протокол IP, значно відрізняється. Мережі Ethernet мають значення MTU, що дорівнюють 1500 байт, мережі FDDI - 4096 байт, а мережі X.25 найчастіше працюють з MTU в 128 байт.

IP-пакет може бути позначений як такий, що не можна фрагментувати (DF = 1). Будь-який пакет, що помічений таким чином, не може бути фрагментований модулем IP ні за яких умов. Якщо такий пакет не може досягти одержувача без фрагментації, то цей пакет просто знищується, а вузлу-відправнику надсилається відповідне ICMP-повідомлення.

Протокол IP допускає можливість використання в межах окремої підмережі її власних засобів фрагментування, невидимих для протоколу IP. Наприклад, технологія ATM ділить IP-пакети, що надходять на комірки з полем даних в 48 байт за допомогою свого рівня сегментування, а потім збирає комірки у вихідні пакети на виході з мережі. Але такі технології, як ATM, є скоріше винятком, ніж правилом.

Процедури фрагментації та збирання протоколу IP розраховані на те, щоб пакет міг бути розбитий на практично будь-яку кількість частин, які згодом могли б бути знову зібрані. Одержувач фрагмента використовує поле ідентифікації для того, щоб не переплутати фрагменти різних пакетів. Модуль IP, що відправляє пакет, встановлює в поле ідентифікації значення, яке має бути унікальним для даної пари відправник - одержувач, а також час, протягом якого пакет може бути активним в мережі.

Поле зсуву фрагмента повідомляє одержувачеві положення фрагмента у вихідному пакеті. Зсув фрагменту та його довжина визначають частину вихідного пакету, яка міститься у цьому фрагменті. Прапор «More fragments» показує появу

останнього фрагмента. Модуль протоколу IP, що відправляє нерозбитий на фрагменти пакет, встановлює в нуль прапор «More fragments» і зсув у фрагменті.

Ці поля дають достатню кількість інформації для збирання пакета.

Щоб розділити на фрагменти великий пакет, модуль протоколу IP, встановлений, наприклад, на маршрутизаторі, створює кілька нових пакетів і копіює вміст полів IP - заголовка з великого пакета в IP - заголовки усіх нових пакетів. Дані зі старого пакета діляться на відповідну кількість частин, розмір кожної з яких, окрім останньої, обов’язково повинен бути кратним 8 байт. Розмір останньої частини даних дорівнює отриманому залишку.

Кожна з отриманих частин даних розміщується в новому пакеті. Коли відбувається фрагментація, то деякі параметри IP-заголовка копіюються в заголовки усіх фрагментів, а інші залишаються лише у заголовку першого фрагмента. Процес фрагментації може змінити значення деяких полів. Наприклад, значення контрольної суми заголовка, прапора «More fragments», зміщення фрагмента, довжини IP - заголовка та загальної довжини пакета. В заголовок кожного пакета заносяться відповідні значення в поле зсуву «Fragment offset», а в поле загальної довжини пакету розміщується довжина кожного пакета. Перший фрагмент матиме в полі «Fragment offset» нульове значення. В усіх пакетах, окрім останнього, прапор «More fragments» встановлюється в одиницю, а в останньому фрагменті − в нуль.

Щоб зібрати фрагменти пакета, модуль протоколу IP (наприклад, модуль на хост-комп'ютері) об’єднує IP-пакети, які мають однакові значення в полях ідентифікатора, відправника, одержувача та протоколу. Таким чином, відправник повинен вибрати ідентифікатор таким чином, щоб він був унікальним для даної пари відправник-одержувач, для даного протоколу і протягом того часу, поки даний пакет (або будь-який його фрагмент) може пересуватися по складеній IP-мережі.

Очевидно, що модуль протоколу IP, що відправляє пакети, повинен мати таблицю ідентифікаторів, де кожний запис співвідноситься з кожним окремим одержувачем, з яким здійснювався зв’язок, і вказує останнє значення максимального часу життя пакета в IP-мережі. Однак, оскільки поле ідентифікатора допускає 65536 різних значень, деякі хости можуть використовувати просто унікальні ідентифікатори, які не залежать від адреси одержувача.

У деяких випадках доцільно, щоб ідентифікатори IP-пакетів вибиралися протоколами більш високого рівня, ніж IP-рівень. Наприклад, в протоколі TCP

передбачена повторна передача TCP-сегментів, які з будь-яких причин не дійшли до адресату. Імовірність правильного прийому збільшувалася б, якби при повторній передачі ідентифікатор для IP-пакета був би тим, що й у початковому IP-пакеті, оскільки його фрагменти могли б використовуватися для складання правильного TCP-сегменту.

Процедура об'єднання полягає в поєднанні даних з кожного фрагмента в позицію, зазначену в заголовку пакета в поле «fragment offset».

Кожен модуль IP повинен бути здатний передати пакет з 68 байтів без подальшої фрагментації. Це пов'язано з тим, що IP - заголовок може включати до 60 байтів, а мінімальний фрагмент даних − 8 байтів. Кожен одержувач повинен вміти приймати пакет з 576 байтів в якості єдиного пакету або у вигляді фрагментів, що підлягають складанню.

Якщо біт прапора заборони фрагментації (Do not Fragment, DF) встановлено, то фрагментація даного пакета заборонена, навіть якщо в цьому випадку він буде втрачений. Даний засіб може використовуватися для запобігання фрагментації у тих випадках, коли хост-одержувач не має достатніх ресурсів для складання фрагментів.

Робота протоколу IP по фрагментації пакетів в хостах і маршрутизаторах ілюструється на рис. 6.10.

Нехай комп'ютер 1 пов'язаний з мережею, яка має значення MTU в 4096 байт, наприклад з мережею FDDI. При надходженні на IP-рівень комп'ютера 1 повідомлення від транспортного рівня розміром в 5600 байт протокол IP ділить його на два IP-пакета, встановлюючи в першому пакеті ознаку фрагментації та привласнюючи пакету унікальний ідентифікатор, наприклад 486. У першому пакеті величина поля зсуву дорівнює 0, а в другому - 2800. Ознака фрагментації у другому пакеті дорівнює нулю, що показує, що це останній фрагмент пакета. Загальна довжина IP-пакету становить 2800 байт плюс 20 (розмір IP-заголовка), то тобто 2820 байт, що вміщується в поле даних кадру FDDI. Далі модуль IP комп'ютера 1 передає ці пакети мережевому інтерфейсу (утвореному протоколами канального рівня К1 та фізичного рівня Ф1). Мережевий інтерфейс відправляє кадри наступному маршрутизатору.

Після того, як кадри будуть оброблені рівнем мережевого інтерфейсу маршрутизатора (К1 та Ф1) і звільняться від заголовків FDDI, модуль IP за мережевою адресою визначає, що два пакети, що надійшли, потрібно передати в мережу 2, яка є мережею Ethernet і має значення MTU 1500 байт.

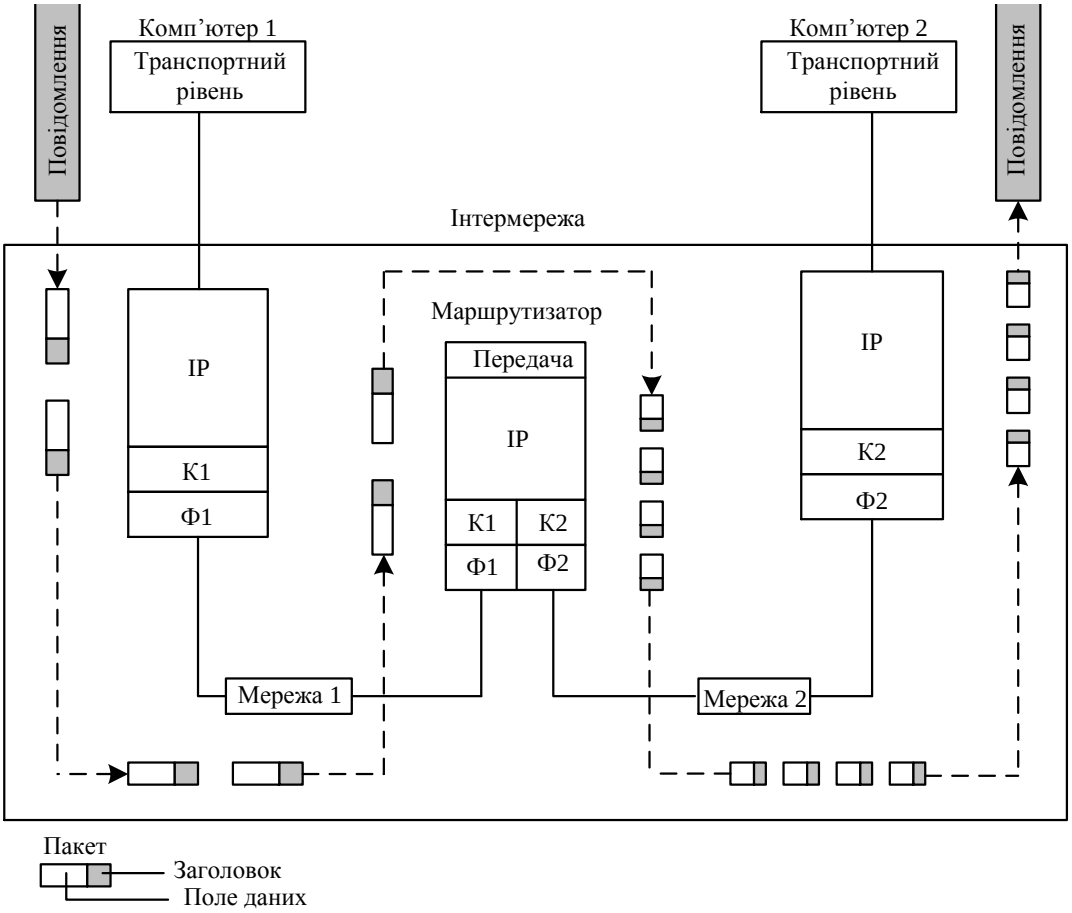

Рис. 6.10 Фрагментація IP-пакетів при передачі між мережами з різним максимальним розміром пакетів:
К1 та Ф1 - канальний та фізичний рівень мережі 1;
К2 та Ф2 - канальний та фізичний рівень мережі 2

Отже, IP-пакети, що надійшли, необхідно фрагментувати. Маршрутизатор витягує поле даних з кожного пакета й ділить його ще навпіл, щоб кожна частина вмістилася в полі даних кадру Ethernet. Потім він формує нові IP-пакети, кожен з яких має довжину 1400 + 20 = 1420 байт, що менше ніж 1500 байт, тому вони нормально розміщуються в полі даних кадрів Ethernet.

У результаті в комп'ютер 2 по мережі Ethernet надходять чотири IP-пакета із загальним ідентифікатором 486, що дозволяє протоколу IP, який працює у комп'ютері 2, правильно зібрати вихідне повідомлення. Якщо пакети надійшли не в тому порядку, в якому були надіслані, то зсув вкаже правильний порядок їх об'єднання.

Відзначимо, що *IP-маршрутизатори не збирають фрагменти пакетів* в більш великі пакети, навіть якщо на шляху зустрічається мережа, яка допускає таке укрупнення. Це пов'язано з тим, що окремі фрагменти повідомлення можуть

переміщатися по інтермережі по різних маршрутах, тому немає гарантії, що усі фрагменти проходять через будь-який проміжний маршрутизатор на їхньому шляху. При надходженні першого фрагменту пакета вузол призначення запускає таймер, який визначає максимально допустимий час очікування надходження інших фрагментів цього пакету. Таймер встановлюється на максимальне з двох значень: початковий час очікування та час життя, вказаний в прийнятому фрагменті. Таким чином, початкове встановлення таймера є нижньою границею для часу очікування при складанні пакету. Якщо часу очікування, встановленний на таймері, минає раніше ніж надходження останнього фрагменту, то усі ресурси збирання, що пов’язані з даними пакетом, звільняються, усі отримані до цього моменту фрагменти пакета відкидаються, а до вузлу, що надіслав вихідний пакет, відправляється повідомлення про помилку за допомогою протоколу ICMP.

6.10 Протокол IPv6

Протягом багатьох років протокол IP залишався надзвичайно популярним. Він працював дуже добре, і основне підтвердження тому - експоненціальне зростання мережі Інтернет. На жаль, протокол IP став жертвою власної популярності: почав закінчуватись адресний простір. Таку загрозу помітили майже 20 років тому, й питання про те, що з цим робити, викликав в інтернет-співтоваристві бурю дискусій та розбіжностей.

Розглянемо цю проблему та її рішення. У перспективі гарною ідеєю є перехід до більш довгих адрес. IPv6 (IP version 6 - IP версії 6) - нова розробка, яка має замінити попередню.

Протокол IPv6 використовує 128-бітові адреси. У доступному для огляду майбутньому додаткового збільшення довжини адреси, швидше за все, не буде потрібно. Однак, виявилося, що впровадити IPv6 не так вже й просто. Це принципово новий протокол мережевого рівня, несумісний з IPv4, незважаючи на подібності цих протоколів. Крім того, компанії та окремі користувачі не розуміють, чому їм треба перейти на IPv6. Тому зараз IPv6 займає лише крихітну частину Інтернету, не дивлячись на те що цей протокол визнаний стандартом ще в 1998 році. На кінець 2012 року частка IPv6 в мережевому трафіку становила близько 5%. До кінця 2013 року очікувалося зростання на 3%. Згідно статистики Google на січень 2020 року, частка IPv6 в мережевому трафіку становила близько 30%. Що буде далі стане зрозуміло в найближчі кілька років, коли будуть розподілені останні IPv4-адреси. Цікаво, чи стануть люди продавати свої адреси на чорному ринку? Невідомо.

Крім перелічених проблем з виділенням адрес, є й інші, що загрозливо виростають на горизонті. До недавніх пір Інтернетом користувалися в основному університети, високотехнологічні підприємства та урядові організації. З лавиноподібним зростанням інтересу до Інтернету, який розпочався в середині 90-х років, в третьому тисячолітті, швидше за все, ним буде користуватися набагато більша кількість користувачів з принципово різними вимогами.

По-перше, користувачі смартфонів можуть підключатися до Інтернету для доступу до домашніх баз даних.

По-друге, при неминучому зближенні комп'ютерної промисловості, засобів зв'язку та індустрії розваг, можливо, дуже скоро кожен телефон та телевізор планети стане вузлом Інтернету, що у результаті призведе до появи мільярдів машин, використовуваних для аудіо та відео на замовлення.

У таких обставинах стає очевидно, що протокол IP повинен еволюціонувати та стати більш гнучким.

Передбачаючи появу цих проблем, проблемна група проектування Інтернету IETF почала у 1990 р. роботу над новою версією протоколу IP, в якій ніколи не повинна виникнути проблема нестачі адрес, а також будуть вирішені багато інших проблем. Крім того, нова версія протоколу повинна була бути більш гнучкою та ефективною.

Були сформульовані наступні основні цілі.

1. Підтримка мільярдів хостів навіть при неефективному використанні адресного простору.

2. Зменшення розміру таблиць маршрутизації.

3. Спрощення протоколу для прискорення обробки пакетів маршрутизаторами.

4. Більш надійне забезпечення безпеки (аутентифікації та конфіденційності).

5. Необхідність звертати більшу увагу на тип сервісу, зокрема, при передачі даних реального часу.

6. Спрощення роботи багатоадресних розсилок за допомогою вказівки областей розсилки.

7. Можливість зміни положення хоста без необхідності змінювати його адресу.

8. Можливість подальшого розвитку протоколу в майбутньому.

9. Можливість співіснування старого та нового протоколів протягом декількох років.

Розробка IPv6 дала шанс поліпшити можливості IPv4, виходячи з потреб сучасного Інтернету. Щоб знайти протокол, що задовольняє усім цим вимогам, IETF запросив спеціалістів до дискусій та пропозицій. Було отримано двадцять одну відповідь. Далеко не усі варіанти містили пропозиції, які повністю задовольняють цим вимогам. У грудні 1992 р. були розглянуті сім серйозних пропозицій. Їх зміст змінювався від невеликих змін в протоколі IP до повної відмови від нього та заміни абсолютно іншим протоколом.

Одна з пропозицій полягала в використанні замість IP протоколу CLNP, який завдяки його 160-розрядної адреси забезпечував би достатній адресний простір на віки вічні. Цього простору вистачило б, якби кожна молекула води у світовому океані захотіла створити власну невелику мережу (близько 2^5 адрес). Крім того, це рішення об'єднало б два основних мережевих протоколів. Однак, все ж визнали, що при подібному виборі доведеться визнати, що дещо в світі OSI було зроблено

правильно, що було б політично некоректним в інтернет-колах. Протокол CLNP, насправді, дуже мало відрізняється від протоколу IP. Остаточний вибір був зроблений на користь протоколу, який відрізняється від IP значно сильніше, ніж CLNP. Ще одним аргументом проти CLNP була його слабка підтримка типу сервісу, який був необхідним для ефективної передачі мультимедіа.

Три кращих пропозиції були опубліковані в журналі IEEE Network Magazine (Deerin, 1993; Francis, 1993; Katz і Ford, 1993). Після довгих обговорень, переробок та боротьби за перше місце була обрана модифікована комбінована версія *Дірінга (Deerin) і Френсіса (Francis)*, яка зараз має назву протокола ***SIPP (Simple Internet Protocol Plus - Простий інтернет-протокол Плюс)***. Новому протоколу було дано позначення ***IPv6***.

Протокол IPv6 гарно виконує поставлені завдання. Він має достоїнства протоколу IP і позбавлений деяких його недоліків, до того ж наділений деякими новими особливостями. У загальному випадку, протокол IPv6 несумісний з протоколом IPv4, але зате сумісний з усіма іншими протоколами Інтернету, включаючи TCP, UDP, ICMP, IGMP, OSPF, BGP і DNS. Основні особливості протоколу IPv6 розглянемо далі.

1. Перш за все, *у протоколі IPv6 поля адрес довше, ніж у IPv4*. Вони мають довжину 128 біт, що вирішує основну проблему, поставлену при розробці протоколу, - забезпечити практично необмежений запас інтернет-адрес. Ми ще коротко згадаємо про адреси трохи пізніше.

2. Друге помітне поліпшення протоколу IPv6 в порівнянні з IPv4 складається в *більш простому заголовку пакета*. Він складається усього з 7 полів (замість 13 у протоколі IPv4). Таким чином, маршрутизатори можуть швидше обробляти пакети, що підвищує їх продуктивність. Короткий опис заголовків буде приведено нижче.

3. Третє удосконалення полягає в *покращеній підтримці необов'язкових параметрів.* Така зміна дійсно була суттєвою, тому що у новому заголовку поля, що раніше були необхідними, стали необов'язковими (бо вони й так використовувалися не часто). Крім того, змінився спосіб представлення необов'язкових параметрів, що прискорило обробку пакетів маршрутизаторами.

4. По-четверте, протокол IPv6 демонструє великий крок вперед в області *безпеки*. У проблемної групи проектування Інтернету IETF була повна папка вирізок з газет із повідомленнями про те, як 12-річні хлопці за допомогою свого персонального комп'ютера через Інтернет вломилися у банк або у військову базу. Було ясно, що треба якось поліпшити систему безпеки. *Аутентифікація та*

конфіденційність є ключовими рисами нового IP-протоколу. Після модифікації IPv4 різниця з точки зору безпеки стала не такою вже й великою.

Нарешті, в новому протоколі було приділено більше уваги якості обслуговування. Різні нерішучі спроби з реалізації якості обслуговування робилися й у минулому, але при зростанні мультимедійного трафіку в Інтернеті відчуття їх актуальності зростає.

6.10.1 Основний заголовок IPv6

Заголовок IPv6 показаний на рис. 6.11.

Зміщення в байтах		0								1								2								3							
	Відступ в бітах	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
0	0	Version				Traffic Class								Flow Label																			
4	32	Payload Length																Next Header								Hop Limit							
8	64	Source Address																															
C	96																																
10	128																																
14	160																																
18	192	Destination Address																															
1C	224																																
20	256																																
24	288																																

Рисунок 6.11 - Фіксований заголовок IPv6 (обов'язкові поля)

Поле ***Версія (Version)*** *(4 біти)* містить число 6 (0110) для IPv6 і число 4 (0100) для IPv4. На період переходу з IPv4 на IPv6, який, ймовірно, займе ще немало років, маршрутизатори за значенням цього поля зможуть відрізняти пакети нового стандарту від старого. Подібна перевірка вимагає кількох тактів процесора. Це може виявитися небажаним в деяких ситуаціях, тим більше, що у заголовку з інформацією про канал передачі даних звичайно зазначається протокол для демультиплексування. Тому цю перевірку можна пропустити. Наприклад, в Ethernet у поля *Тип* існує кілька різних значень, які визначають корисні дані IPv4- або IPv6-пакета.

Дискусія між групами, які керуються принципами «Роби правильно» і «Роби швидко», безсумнівно, буде довгою та енергійною.

Поле ***Приоритет (Traffic class)*** *(8 бітів)* використовується для того, щоб розрізняти пакети з різними вимогами до доставлення в реальному часі. Воно використовується для забезпечення якості обслуговування разом з архітектурою диференційованого обслуговування (так як й однойменне поле IPv4). Це поле складається з двох значень. Старші 6 бітів використовуються для класифікації пакетів. Молодші 2 біти, так як й в IPv4, використовуються для повідомлень про перевантаження.

Поле ***Мітка потоку (Flow label)*** *(20 біт)* застосовується для встановлення між відправником та отримувачем псевдоз'єднання з окремими властивостями та вимогами. Це поле дозволяє значно спростити процедуру маршрутизації однорідного потоку пакетів. *Потік – це послідовність пакетів, що надсилаються відправником окремому адресату.* При цьому передбачається, що усі пакети даного потоку мають бути однаково оброблені. Характер даної обробки задається додатковими заголовками.

Допускається існування декількох потоків між відправником та отримувачем. Мітка потоку присвоюється вузлом-відправником шляхом генерування псевдовипадкового 20-бітного числа. Усі пакети одного потоку повинні мати однакові заголовки, які оброблюються маршрутизатором.

При отриманні першого пакету з міткою потоку маршрутизатор аналізує додаткові заголовки, виконує призначені цими заголовками функції та запам'ятовує результати оброблення (адресу наступного вузла, опції, переміщення адрес у заголовку маршрутизації й т.п.). Ключом для такого запису є комбінація адреси джерела та мітки потоку. Наступні пакети з такою комбінацією адреси джерела та мітки потоку оброблюються без детального аналізу усіх полів заголовка.

Час життя запису складає не більше ніж 6 секунд, навіть якщо пакети цього потоку продовжують надходити. При обнулінні запису та отриманні наступного пакету потока пакет оброблюється у звичайному режимі, й для нього формується новий запис. Час життя потоку може бути визначено вузлом-відправником за допомогою протоколу управління або за допомогою опцій заголовка переходів й може перевищувати 6 секунд.

Поле ***Довжина корисного навантаження (Payload Length)*** повідомляє, скільки байтів слід за 40-байтовим заголовком, показаним на рис. 6.11. У заголовку IPv4 аналогічне поле називалося *Повна довжина* й визначало весь розмір пакета. На відміну від поля *Повна довжина* протоколу IPv4 дане поле не вміщує заголовок пакета. Максимальний розмір, визначений розміром поля, - 64 Кбайта.

Поле ***Наступний заголовок (Next header)*** розкриває секрет можливості використання спрощеного заголовка. Справа в тому, що після звичайного 40-байтового заголовка можуть йти додаткові (необов'язкові) розширені заголовки. Це поле повідомляє, який з шести додаткових заголовків (на даний момент) слід за основним. В останньому IP-заголовку поле *Наступний заголовок* повідомляє, якій програмі обробленя протоколу транспортного рівня (тобто TCP або UDP) передати цей пакет.

Поле ***Максимальне число транзитних ділянок (Hop limit)*** *(20 біт)* не дає пакетам вічно блукати по мережі. Це поле - аналог поля *Час життя (Time to live)* в протоколі IPv4. Це поле зменшується на одиницю на кожної транзитної ділянці. Теоретично, в протоколі IPv4 це поле повинно було містити секунди часу життя пакета. Проте жоден маршрутизатор не використав його подібним чином, тому ім'я поля було приведено у відповідність зі способом його застосування.

Далі ідуть поля ***Адреса відправника*** та ***Адреса одержувача (Source Address*** та ***Destination Address)*** *(по 128 біт)*. У вихідної пропозиції Дірінга (протоколі SIPP) використовувалися 8-байтові адреси. Але при розгляді проекту було вирішено, що 8-байтових адрес вистачить лише на кілька десятиліть, в той час як 16-байтових адрес повинно вистачити надовго. Інші розробники заперечували, що 16 байт для адрес занадто багато, тоді як треті наполягали на 20-байтних адресах для сумісності з дейтаграмним протоколом OSI. Ще одна фракція виступала за адреси змінної довжини. Після тривалих суперечок було вирішено, що найкращим компромісним рішенням є *16-байтові адреси фіксованої довжини*.

Для написання 16-байтових адрес була розроблена нова нотація. Адреси в IPv6 записуються у вигляді восьми груп по чотири шістнадцяткові цифри, розділених двокрапкою, наприклад:

8000: 0000: 0000: 0000: 0123: 4567: 89AB: CDEF

Оскільки багато адрес будуть містити велику кількість нулів, були дозволені три метода скороченого запису адрес.

По-перше, можуть бути пропущені старші нулі в кожній групі, наприклад 0123 можна записувати як 123.

По-друге, одна або більше груп, що повністю складаються з нулів, можуть замінюватися парою двокрапок.

Таким чином, наведена вище адреса буде мати такий вигляд:

8000 :: 123: 4567: 89AB: CDEF

Нарешті, адреси IPv4 можуть записуватися як пара двокрапок, після якої пишеться адреса в старому десятковому форматі, наприклад:

:: 192.31.20.46

Кількість усіх можливих 16-байтових адрес є дуже великою - 2^{128}, що приблизно дорівнює $3 \cdot 10^{38}$. Якщо покрити комп'ютерами усю планету, включаючи сушу та океани, то протокол IPv6 дозволить мати біля $7 \cdot 10^{23}$ IP-адрес на квадратний метр. Це число більше ніж число Авогадро. Хоча в плани розробників не входило надання власної IP-адреси кожній молекулі на поверхні Землі, вони виявилися не такими далекими від забезпечення такої послуги.

На практиці не весь адресний простір використовується ефективно. Наприклад, не використовуються абсолютно усі комбінації телефонних номерів. Наприклад, телефонні номери Манхеттена (код 212) майже повністю зайняті, тоді як в штаті Вайомінг (код 307) вони майже не використовуються. В RFC 3194 стверджується, що якщо орієнтуватися на використання телефонних номерів, то навіть при самому песимістичному сценарії все одно виходить більше ніж 1000 IP-адрес на квадратний метр поверхні Землі (включаючи як сушу, так й море). При будь-якому більш ймовірному сценарії забезпечуються трильйони адрес на квадратний метр. Таким чином, малоймовірно, що в доступному для огляду майбутньому виявиться брак адрес. Також слід відмітити, що на сьогодні тільки для 28% адресного простору передбачено використання. Інші 72% зарезервовані на майбутнє.

Корисно порівняти заголовок IPv4 (рис. 6.9) з заголовком IPv6 (рис. 6.11), щоб побачити, що залишилося від старого стандарту. Поле *Довжина заголовка* зникло, оскільки заголовок IPv6 має фіксовану довжину. Поле *Протокол* також було прибрано, оскільки поле *Наступний заголовок* повідомляє, що слідує за останнім IP-заголовком (тобто UDP або TCP-сегмент).

Були вилучені усі поля, що відносяться до *фрагментації*, так як в протоколі IPv6 використовується інший підхід до фрагментації. По-перше, усі хости, що підтримують протокол IPv6, повинні динамічно визначати потрібний розмір пакета. Для цього використовується технологія Path MTU discovery. Коли хост надсилає занадто великий IPv6-пакет, замість того щоб його фрагментувати, то маршрутизатор, нездатний переслати пакет далі, надсилає назад повідомлення про

помилку. Отримавши це повідомлення, хост повинен припинити передачу цьому адресату. Набагато правильніше буде навчити усі хости відправляти пакети необхідного розміру, ніж вчити маршрутизатори фрагментувати їх на льоту.

Крім того, *мінімальний розмір пакета* був збільшений з 576 до 1280, щоб можна було передавати 1024 байт даних, плюс кілька заголовків.

Нарешті, поле *Контрольна сума* було видалено, так як її підрахунок значно знижує продуктивність. В даний час усе більше використовуються надійні лінії зв'язку, а на канальному та транспортному рівнях підраховуються свої контрольні суми. Наявність ще однієї контрольної суми не варта була б тих витрат продуктивності, яких вимагав би її підрахунок. В результаті перелічених видалених полів заголовка пакета вийшов простий, швидкий та в той же час гнучкий протокол мережевого рівня з величезним адресним простором.

6.10.2 Додаткові заголовки

В видалених полях заголовка іноді виникає необхідність, тому в протоколі IPv6 була представлена нова концепція (необов'язкового) додаткового заголовка.

На сьогодні визначено шість типів додаткових заголовків, які перелічені у таблиці 6.6. Усі вони є необов'язковими, але в разі використання більш ніж одного додаткового заголовка вони повинні розташовуватися відразу за фіксованим заголовком, бажано в зазначеному порядку.

Таблиця 6.6 - Додаткові заголовки IPv6

Додатковий заголовок	**Опис**
Параметри маршрутизації	Різноманітна інформація для маршрутизаторів
Параметри одержувача	Додаткова інформація для одержувача
Маршрутизація	Частковий список транзитних маршрутизаторів на шляху пакету
Фрагментація	Управління фрагментами дейтаграм
Аутентифікація	Перевірка дійсності відправника
Шифровані дані	Інформація про зашифрованому вмісті

У деяких заголовків формат фіксований, інші містять змінну кількість полів змінної довжини. Для них кожен пункт кодується у вигляді трійки (тип, довжина, значення).

Тип є однобайтовим полем, що містить код параметра. Перші два біти цього поля повідомляють маршрутизаторам, які не знають як обробляти даний параметр, що треба робити з пакетом. Можливі чотири наступні варіанти: пропустити параметр, ігнорувати пакет, ігнорувати пакет та надіслати назад ICMP-пакет, а також те ж саме, що й попередній варіант, але не надсилати назад ICMP-пакет у разі багатоадресної розсилки (щоб один невірний багатоадресний пакет не породив мільйони ICMP-повідомлень).

Поле *Довжина* також має розмір 1 байт. Воно повідомляє, наскільки велике значення (від 0 до 255 байт).

Поле *Значення* містить необхідну інформацію розміром до 255 байт.

Заголовок параметрів маршрутизації містить інформацію, яку повинні досліджувати маршрутизатори протягом усього шляху просування пакету. Поки що було визначено один варіант використання цього параметра: підтримка дейтаграм, що перевищують 64 Кбайт. Формат заголовка наведена на рис. 6.12. При цьому полю *Довжина корисного навантаження* у фіксованому заголовку присвоюється значення 0.

Наступний заголовок	0	194	4
Довжина корисного навантаження			

Рисунок 6.12 - Додатковий заголовок для великих дейтаграм

Як й усі додаткові заголовки, він починається з байту, що означає тип наступного заголовка. Наступний байт містить довжину додаткового заголовка в байтах, не враховуючи перші 8 байт, що є обов'язковими. З цього починаються усі розширення.

Наступні два байти вказують, що даний параметр містить розмір дейтаграми (код 194) у вигляді 4-байтового числа. Розміри менше ніж 65 536 не допускаються, так як можуть призвести до того, що перший же маршрутизатор проігнорує даний пакет та відправить назад ICMP-повідомлення про помилку. Дейтаграми, що використовують подібні розширення заголовка, називаються *джамбограмами* (*jumbograms*, від слова «jumbo», що означає щось велике та незграбне). Використання джамбограм важливо для суперкомп'ютерних додатків, яким необхідно ефективно передавати по Інтернету гігабайти даних.

Заголовок розширюється на поля, які повинні інетерпретуватися тільки ***хостом-одержувачем***. У початковій версії IPv6 задавалися «нульові» настройки

для доповнення цього заголовка до кратності 8 байт, а сам заголовок не використовувався. Це передбачалось для того, щоб програмне забезпечення нових роутерів та хостів могло їх обробити, якщо хтось подумає про додаткові налаштування у майбутньому.

Маршрутний заголовок містить інформацію про один або про кілька маршрутизаторів, через які слід пройти по шляху до одержувача. Це дуже сильно нагадує вільну маршрутизацію стандарту IPv4 в тому, що зазначені в списку маршрутизатори повинні бути пройдені строго по порядку, тоді як через не вказані проходять між ними. Формат маршрутного заголовка показаний на рисунку 6.13.

Наступний заголовок	Довжина додаткового заголовка	Тип маршрутизації	Кількість сегментів, що залишились
Дані, що залежать від типу			

Рисунок 6.13 - Додатковий заголовок для маршрутизації

Перші чотири байти додаткового маршрутного заголовка містять чотири однобайтних цілих числа. Поля *Наступний заголовок* і *Довжина додаткового заголовка* були описані раніше. В поле *Тип маршрутизації* описується формат решти заголовка. Якщо він дорівнює 0, це означає, що далі йде зарезервоване 32-розрядне слово, а за ним - деяка кількість адрес IPv6. У майбутньому, можливо, будуть в міру необхідності винаходити якісь нові поля.

Нарешті, в поле *Кількість сегментів, що залишились*, вказується, скільки адрес зі списку ще залишилося відвідати. Його значення зменшується при проходженні кожної адреси. Коли воно досягає нуля ніяких вказівок щодо подальшого маршруту пакета не надається. Зазвичай в цей момент пакет вже знаходиться досить близько до місця призначення, й оптимальний маршрут очевидний.

Заголовок фрагментації визначає фрагментацію способом, схожим з протоколом IPv4.

Заголовок містить ідентифікатор дейтаграми, номер фрагмента й біт, що інформує про те, чи є цей фрагмент останнім. На відміну від IPv4 в протоколі IPv6

фрагментувати пакет може тільки хост-джерело. Маршрутизатор фрагментувати пакети, що надсилаються, не можуть. Хоча цю зміну можна вважати відмовою від оригінальної філософії IP, вона цілком в концепції сучасного застосування IPv4. До того ж вона спрощує та прискорює роботу маршрутизаторів. Як вже було сказано, маршрутизатор відкидає занадто великі пакети, посилаючи у відповідь ICMP-пакет, який вказує хосту-джерелу на необхідність повторно передати пакет, виконавши його фрагментацію на менші частини.

Заголовок аутентифікації надає механізм підтвердження автентичності відправника пакета.

Шифрування даних, що містяться в поле корисного навантаження, забезпечує конфіденційність: прочитати вміст пакету зможе тільки той, для кого призначений пакет. Для виконання цих завдань в заголовках використовуються криптографічні методи захисту даних.

6.10.3 Полеміка

При тій відкритості, з якої відбувався процес розроблення протоколу IPv6, й при переконаності розробників у власній правоті не дивно, що багато рішень приймалися в умовах вельми палких дискусій. Усі подробиці описані у відповідних RFC.

Про спори з приводу довжини поля адреси вже згадувалося. В результаті було прийнято компромісне рішення: 16-байтові адреси фіксованої довжини.

Інший бій розгорівся через розмір поля *Максимальна кількість транзитних ділянок*. Одна з груп вважала, що обмеження кількості транзитних ділянок числом 255 (це явно випливає з використання 8-бітного поля) є великою помилкою. Насправді, маршрути з 32 транзитних ділянок вже стали звичайними, а через 10 років можуть стати звичайними більш довгі маршрути. Прихильники цієї пропозиції заявляли, що використання полів адрес величезного розміру було рішенням далекоглядним, а крихітних лічильників транзитних ділянок - недалекоглядним. Найстрашніший гріх, який, на їхню думку, можуть зробити фахівці з обчислювальної техніки, - це виділити для чогось недостатню кількість розрядів.

У відповідь їм було заявлено, що подібні аргументи можна привести для збільшення будь-якого поля, що призведе до розбухання заголовка. Крім того, призначення поля *Максимальна кількість транзитних ділянок* полягає в тому, щоб

не допустити занадто довгого пересування пакетів, й 65 535 транзитних ділянок - це дуже багато. До того ж, у міру зростання Інтернету буде створюватися все більша кількість міжміських ліній, що дозволить передавати пакети з будь-якої країни в будь-яку країну максимум за шість транзитних пересилань. Якщо від одержувача або відправника до відповідного міжнародного шлюзу виявиться більше 125 транзитних ділянок, то, мабуть, щось не в порядку з магістралями цієї держави.

У підсумку цю битву виграли прихильники 8-бітового лічильника.

Ще одним предметом спору виявився максимальний розмір пакету. Володарі суперкомп'ютерів наполягали на розмірі пакетів, що перевищує 64 Кбайт. Коли суперкомп'ютер починає передачу, він займається серйозною справою й не хоче, щоб його переривали через кожні 64 Кбайт.

Аргумент проти великих пакетів полягає в тому, що якщо пакет розміром в 1 Мбайт буде передаватися по лінії T1 зі швидкістю 1,5 Мбіт / с, то він займе лінію на цілих 5 с, що буде викликати занадто велику затримку, помітну для інтерактивних користувачів. В даному питанні вдалося досягти компромісу: нормальні пакети обмежені розміром 64 Кбайт, але за допомогою додаткового заголовка можна пересилати дейтаграми величезного розміру.

Ще одним спірним питанням виявилося видалення контрольної суми IPv4. Дехто порівнював цей хід з видаленням гальм з автомобіля. При цьому автомобіль стає легше й може рухатися швидше, але якщо трапиться щось несподіване, то можуть бути проблеми.

Аргумент проти контрольних сум полягав у тому, що кожен додаток, який дійсно піклується про цілісність своїх даних, все одно перераховує контрольну суму на транспортному рівні. Тому наявність ще однієї контрольної суми на мережевому рівні є зайвим. Крім того, контрольна сума підраховується ще й на рівні передачі даних. Більш того, експерименти показали, що обчислення контрольної суми становило основні витрати протоколу IPv4. Ця битва була виграна табором супротивників контрольної суми, тому в протоколі IPv6 контрольної суми немає.

Навколо мобільних хостів також розгорілася суперечка. Якщо мобільний комп'ютер виявиться на іншому кінці Землі, чи зможе він продовжувати використовувати колишню IPv6-адресу, або він повинен буде використовувати схему з внутрішнім та зовнішнім агентами?

З'явилося багато бажаючих створити в протоколі IPv6 явну підтримку мобільних хостів. Ці спроби зазнали поразки, оскільки по жодній конкретній

пропозиції не вдалося досягти консенсусу. Ймовірно, найбільш жаркі баталії розгорілися навколо питання безпеки. Усі були згодні, що це необхідно. Спірним було те, де й як слід реалізовувати безпеку. По-перше - де. Аргументом за розміщення системи безпеки на мережевому рівні було те, що при цьому вона стає стандартною службою, якою можуть користуватися усі додатки без жодного попереднього планування.

Контраргумент полягав у тому, що по-справжньому захищеним додаткам підходить лише наскрізне шифрування, коли шифрування здійснюється самим джерелом, а дешифрування - безпосереднім одержувачем. У всіх інших випадках користувач опиняється в залежності від реалізації мережевого рівня, яка, можливо, містить помилки, та над якою у нього немає контролю. У відповідь на цей аргумент можна сказати, що додаток може просто відмовитися від використання вбудованих в IP функцій захисту та виконувати усю цю роботу самостійно. Заперечення на цей контраргумент полягає у тому, що користувачі, які не довіряють мережам, не хочуть платити за функцію, яка ними не використовується, і реалізація якої ускладнює та уповільнює роботу протоколу, навіть якщо сама функція відключена.

Інший аспект питання розташування системи безпеки стосується того факту, що в багатьох (але не в усіх) країнах прийняті суворі експортні закони, що стосуються криптографії. У деяких країнах, особливо у Франції та Іраку, суворо заборонено використання криптографії навіть всередині країни, щоб у населення не могло бути секретів від поліції. У результаті будь-яка реалізація протоколу IP, що використовує досить потужну криптографічну систему, не може бути експортована за межі Сполучених Штатів (й багатьох інших країн).

Таким чином, доводиться підтримувати два набори програмного забезпечення - один для внутрішнього використання, а інший для експорту, проти чого рішуче виступає більшість виробників комп'ютерів.

Єдине питання, по якому не було суперечок, полягав у тому, що ніхто не очікує, що IPv4-інтернет буде вимкнений в неділю, а в понеділок вранці буде включений вже IPv6-інтернет. Замість цього спочатку з'являться «островки» IPv6, які будуть спілкуватися по тунелях. У міру свого зростання, островки IPv6 будуть об'єднуватися в більші островки. Нарешті, усі островки об'єднаються, й Інтернет виявиться повністю трансформованим.

Так, принаймні, виглядав план. Впровадження IPv6 виявилося набагато складнішим. Цей протокол до сих пір дуже мало використовується, хоча більшість операційних систем повністю підтримують його. Як правило, IPv6 застосовується

у тих випадках, коли оператору будь-якої мережі (наприклад, мобільної) потрібні додаткові IP-адреси.

Щоб спростити перехід до нового протоколу, було придумано багато різних стратегій. Серед них методи автоматичної настройки тунелів, які забезпечують передачу IPv6-пакетів через мережу IPv4, й технології, що дозволяють хостам автоматично знаходити кінцеву точку тунелю. Двохстекові хости підтримують як IPv4, так й IPv6 і можуть вибирати протокол в залежності від адреси призначення пакета. Ці стратегії допоможуть прискорити процес переходу до IPv6, коли він буде неминучим.

6.11 Протоколи маршрутизації

Протоколи маршрутизації можуть бути побудовані на основі різних алгоритмів, що відрізняються способами побудови таблиць маршрутизації, способами вибору найкращого маршруту та іншими особливостями своєї роботи.

6.11.1 Класифікація протоколів маршрутизації

Як правило, при виборі оптимального маршруту в комунікаційному вузлі визначається тільки наступний (найближчий) маршрутизатор, а не вся послідовність маршрутизаторів від початкового до кінцевого вузла. Відповідно до цього підходу, маршрутизація виконується за розподіленою схемою − кожен маршрутизатор є відповідальним за вибір тільки одного кроку маршруту. А остаточний маршрут складається в результаті роботи усіх маршрутизаторів, через які проходить даний пакет. Такі алгоритми маршрутизації називаються ***однокроковими***.

Існує й прямо протилежний, ***багатокроковий*** підхід − ***маршрутизація від джерела (Source Routing)***. Відповідно до нього вузол-джерело задає у відправленому в мережу пакеті повний маршрут його прямування через усі проміжні маршрутизатори. При використанні багатокрокової маршрутизації немає необхідності будувати та аналізувати таблиці маршрутизації. Це прискорює проходження пакету по мережі, розвантажує маршрутизатори, але при цьому виявляється велике навантаження у кінцевих вузлах. Ця схема в обчислювальних мережах застосовується сьогодні набагато рідше, ніж схема розподіленої однокрокової маршрутизації. Однак, у новій версії протоколу IP поряд з класичною однокроковою маршрутизацією дозволяється й маршрутизація від джерела.

Однокрокові алгоритми *в залежності від способу формування таблиць маршрутизації* поділяються на три класи:

- алгоритми фіксованої (або статичної) маршрутизації;
- алгоритми простої маршрутизації;
- алгоритми адаптивної (або динамічної) маршрутизації.

В ***алгоритмах фіксованої маршрутизації*** усі записи в таблиці маршрутизації є статичними. Адміністратор мережі сам вирішує, на які

маршрутизатори треба передавати пакети з тими чи іншими адресами, й вручну (наприклад, за допомогою утиліти route ОС Unix або Windows NT) заносить відповідні записи в таблицю маршрутизації. Таблиця, як правило, створюється в процесі завантаження. В подальшому вона використовується без змін до тих пір, поки її вміст не буде змінено вручну. Такі виправлення можуть знадобитися, наприклад, якщо в мережі відмовляє будь-який маршрутизатор та його функції покладаються на інший маршрутизатор. Розрізняють ***одномаршрутні таблиці***, в яких для кожного адресата заданий один шлях, та ***багатомаршрутні таблиці***, які визначають кілька альтернативних шляхів для кожного адресата. У багатомаршрутних таблицях повинно бути задане правило вибору одного з маршрутів. Найчастіше один шлях є основним, а решта − резервними. Зрозуміло, що алгоритм фіксованої маршрутизації, з його ручним способом формування таблиць маршрутизації, прийнятний тільки в невеликих мережах з простою топологією. Однак, цей алгоритм може бути ефективно використаний також для роботи на магістралях великих мереж, так як сама магістраль може мати просту структуру з очевидними найкращими шляхами проходження пакетів в підмережі, що приєднана до магістралі.

В ***алгоритмах простої маршрутизації*** таблиця маршрутизації або зовсім не використовується, або будується без участі протоколів маршрутизації. Виділяють три типи простої маршрутизації:

- випадкова маршрутизація, коли отриманий пакет надсилається по першому випадковому напрямку, крім вхідного;
- лавинна маршрутизація, коли пакет широкомовно надсилається по усіх можливих напрямах, крім вхідного (аналогічно обробленню мостами або комутаторами кадрів з невідомою адресою);
- маршрутизація з попереднього досвіду, коли вибір маршруту здійснюється по таблиці, але таблиця будується за принципом моста або комутатора шляхом аналізу адресних полів пакетів, що з'являються на вхідних портах.

Найпоширенішими є ***алгоритми адаптивної (або динамічної) маршрутизації***. Ці алгоритми забезпечують автоматичне оновлення таблиць маршрутизації після зміни конфігурації мережі. Протоколи, які побудовані на основі адаптивних алгоритмів, дозволяють усім маршрутизаторам збирати інформацію про топологію зв'язків в мережі, оперативно відпрацьовуючи усі зміни

конфігурації зв'язків. У таблицях маршрутизації при адаптивної маршрутизації зазвичай є інформація про інтервал часу, протягом якого даний маршрут буде залишатися дійсним. Це час називають часом життя маршруту (Time To Live , TTL).

Адаптивні алгоритми зазвичай мають розподілений характер, який виражається в тому, що в мережі відсутні будь-які виділені маршрутизатори, які збирали би та узагальнювали топологічну інформацію. Ця робота розподілена між усіма маршрутизаторами.

Останнім часом намітилася тенденція використовувати так звані сервери маршрутів. Сервер маршрутів збирає маршрутну інформацію, а потім роздає її за запитами маршрутизаторів, які у такому випадку звільняються від функції створення таблиць маршрутизації, або створюють тільки частини цих таблиць. З'явилися спеціальні протоколи взаємодії маршрутизаторів з серверами маршрутів, наприклад протокол Next Hop Resolution Protocol (NHRP).

Адаптивні алгоритми маршрутизації повинні відповідати кільком важливим *вимогам*.

По-перше, вони *повинні забезпечувати*, якщо не *оптимальність*, то хоча б *раціональність маршруту*.

По-друге, алгоритми *повинні бути досить простими*, щоб при їх реалізації не витрачалося занадто багато мережевих ресурсів, зокрема вони не повинні вимагати занадто великого обсягу обчислень або породжувати інтенсивний службовий трафік.

Й нарешті, алгоритми маршрутизації *повинні мати властивість збіжності*, тобто завжди призводити до однозначного результату за сприятливий час.

Адаптивні протоколи обміну маршрутною інформацією, що застосовуються в даний час в обчислювальних мережах, в свою чергу поділяються на дві групи, кожна з яких пов'язана з одним з наступних типів алгоритмів:

- дистанційно-векторні алгоритми (Distance Vector Algorithms , DVA);
- алгоритми стану зв'язків (Link State Algorithms , LSA).

В ***алгоритмах дистанційно-векторного типу*** кожен маршрутизатор періодично та широкомовно розсилає по мережі вектор, компонентами якого є відстані від даного маршрутизатора до усіх відомих йому мереж. Під відстанню зазвичай розуміється кількість хопів (кроків). Можлива й інша метрика, яка враховує не тільки кількість проміжних маршрутизаторів, а й час проходження пакетів по мережі між сусідніми маршрутизаторами. При отриманні вектору від сусіда маршрутизатор нарощує відстані до зазначених у векторі мереж на відстань

до даного сусіда. Отримавши вектор від сусіднього маршрутизатора, кожен маршрутизатор додає до нього інформацію про відомі йому інші мережі, про які він дізнався безпосередньо (якщо вони підключені до його портів) або з аналогічних оголошень інших маршрутизаторів, а потім знову розсилає нове значення вектору по мережі. Зрештою, кожен маршрутизатор дізнається інформацію про усі наявні в інтермережі мережі та про відстань до них через сусідні маршрутизатори.

Дистанційно-векторні алгоритми добре працюють тільки в невеликих мережах. У великих мережах вони навантажують лінії зв'язку інтенсивним широкомовним трафіком. До того ж зміни конфігурації можуть відпрацьовуватися за цим алгоритмом не завжди коректно, так як маршрутизатори не мають точного уявлення про топологію зв'язків в мережі, а мають у своєму розпорядженні тільки узагальнену інформацію − вектор дистанцій. До того ж ця інформація отримана через посередників. Робота маршрутизатора відповідно до дистанційно-векторного протоколу нагадує роботу моста, оскільки точної топологічної картини мережі такий маршрутизатор не має.

Найбільш поширеним протоколом, заснованим на дистанційно-векторному алгоритмі, є ***протокол RIP***, який поширений в двох версіях − RIP IP, що працює з протоколом IP, та RIP IPX, що відповідно працює з протоколом IPX.

Алгоритми стану зв'язків забезпечують кожен маршрутизатор інформацією, достатньою для побудови точного графа зв'язків мережі. Усі маршрутизатори працюють на підставі однакових графів. Це робить процес маршрутизації більш стійким до змін конфігурації. «Широкомовна» розсилка (тобто передача пакета усім безпосереднім сусідам маршрутизатора) використовується в таких алгоритмах тільки при змінах стану зв'язків, що відбувається в надійних мережах не так часто. Вершинами графа є як маршрутизатори, так й мережі, що з'єднуються за допомогою цих маршрутизаторів. Інформація, що поширюється мережею, складається з опису зв'язків різних типів, наприклад: маршрутизатор − маршрутизатор, маршрутизатор − мережа. Щоб зрозуміти, в якому стані знаходяться лінії зв'язку, які підключені до портів, маршрутизатор періодично обмінюється короткими пакетами HELLO зі своїми найближчими сусідами. Цей службовий трафік також навантажує мережу, але не в такій мірі як, наприклад, RIP − пакети, так як пакети HELLO мають набагато менший обсяг.

Протоколами, заснованими на алгоритмі стану зв'язків, є протоколи IS − IS (Intermediate System to Intermediate System) стека OSI, ***OSPF (Open Shortest Path First)*** стека TCP/IP та недавно реалізований протокол NLSP стека Novell.

6.11.2 Протокол маршрутизації OSPF

Як й протокол RIP, протокол OSPF (Open Shortest Path First − відкритий протокол вибору найкоротшого маршруту) використовується для маршрутизації всередині автономної системи. Слово «Open» в назві протоколу означає, що специфікація протоколу маршрутизації вільно поширюється (на відміну від, наприклад, специфікації протоколу EIGRP корпорації Cisco). Остання (друга) версія протоколу OSPF визначена в RFC 2328.

Протокол OSPF вважається наступником протоколу RIP та має ряд додаткових функцій. Однак, протокол OSPF являє собою протокол, заснований на обліку стану ліній. Цей протокол використовує метод лавинної розсилки для поширення інформації про стан ліній, а також ***алгоритм визначення шляху найменшої вартості Дейкстри***.

У протоколі OSPF процес побудови таблиці маршрутизації розбивається на два крупних етапи.

На першому етапі *кожен маршрутизатор будує граф зв'язків мережі*, в якому вершинами графа є маршрутизатори та IP-мережі, а ребрами − інтерфейси маршрутизаторів. Усі маршрутизатори для цього обмінюються зі своїми сусідами тією інформацією про граф мережі, якою вони володіють на даний момент. Цей процес схожий на процес поширення векторів відстаней до мереж в протоколі RIP. Проте ця інформація у протоколі OSPF є якісно іншою − це інформація про топологію мережі. Ці повідомлення називаються *router links advertisement − оголошення про зв'язки маршрутизатора*. Крім того, при передачі топологічної інформації маршрутизатори її не змінюють, як це роблять RIP-маршрутизатори, а передають в незмінному вигляді. У результаті поширення топологічної інформації усі маршрутизатори мережі мають ідентичні відомості про графи мережі, які зберігаються в топологічної базі даних маршрутизатора.

Другий етап полягає в *знаходженні оптимальних маршрутів за допомогою отриманого графа*. Кожен маршрутизатор вважає себе центром мережі й шукає оптимальний маршрут до кожної мережі, яка йому відома. У кожному знайденому таким способом маршруті запам'ятовується тільки один крок − до наступного маршрутизатора, відповідно до принципу *однокрокової маршрутизації*. Дані про

цей крок й потрапляють в таблицю маршрутизації. Завдання визначення оптимального шляху на графі є досить складним та трудомістким. У протоколі OSPF для її вирішення використовується ітеративний алгоритм Дейкстри. Якщо кілька маршрутів мають однакову метрику до мережі призначення, то в таблиці маршрутизації запам'ятовуються перші кроки усіх цих маршрутів.

Після початкової побудови таблиці маршрутизації необхідно відстежувати зміни стану мережі та вносити корективи в таблицю маршрутизації. Для контролю стану зв'язків та сусідніх маршрутизаторів OSPF−маршрутизатори не використовують обмін повною таблицею маршрутизації. Замість цього вони передають спеціальні короткі повідомлення HELLO. Якщо стан мережі не змінюється, то OSPF-маршрутизатори не здійснюють коректування таблиць маршрутизації і не надсилають сусіднім вузлам оголошення про зв'язки. Якщо стан зв'язку змінюється, то до найближчих маршрутизаторів надсилається нове оголошення. Таке оголошення стосується тільки зв'язку, що змінюється. Це, звичайно, економить пропускну здатність мережі.

6.11.3 Алгоритм маршрутизації Дейкстри

Алгоритм пошуку найкоротшого маршруту OSPF заснований на алгоритмі пошуку «найкоротшого шляху» Дейкстри. Цей алгоритм визначає найкоротший шлях від заданої вершини графа мережі до усіх інших вершин. Довжина шляху визначається як сума довжин ребер графа, що з'єднують вершини, через які проходить шлях.

Фізична модель алгоритму Дейкстри

Для пояснення функціонування алгоритму Дейкстри розглянемо його фізичну модель. Уявимо набір з N куль, з'єднаних між собою нитками. Нитки мають різну довжину. Ми вибираємо будь-яку одну кулю й називаємо її «куля 1». Нашою метою є знаходження найкоротшого шляху від «кулі 1» до кожної з решти куль. Для знаходження цих шляхів ми кладемо усі кулі на поверхню і починаємо повільно піднімати «кулю 1». Наступну кулю, що піднялася над поверхнею, ми називаємо «куля 2». Коли «куля 2» піднялася над поверхнею ми бачимо, що найкоротший шлях між кулями 1 та 2 − це нитка, яка з'єднує ці кулі. У міру того, як ми продовжуємо піднімати «кулю 1» інша куля - «куля 3» піднімається над поверхнею. Найкоротший шлях між кулями 1 та 3 − це або нитка між кулями 1 та

3, або складений шлях між кулями 1, 2 та 2, 3. Продовжуючи діяти таким чином, ми піднімемо усі кулі з поверхні. Коли піднімається чергова «куля (n+1)» ми знаходимо найкоротший шлях між кулями 1 та ($n+1$). Крок за кроком ми знайдемо усі найкоротші шляхи між кулею 1 та іншими кулями.

Алгоритмічна форма розглянутого вище фізичного процесу називається *алгоритмом «найкоротшого шляху» Дейкстри*. Алгоритм заснований на наступному спостереженні: відстань від кулі 1 до кулі n не може скорочуватися після того, як куля n підніметься з поверхні. Отже, якщо ми розглядаємо оцінку відстані від кулі 1 до кулі n, ми знаємо, що ця оцінка ґрунтується на найкоротшій відстані, яку ми визначаємо, коли куля n піднімається з поверхні.

Звідки ми знаємо яка куля підніметься наступною? Розглянемо кулю $\boldsymbol{k}$, яка піднімається з поверхні, та відстань між кулею **1** та кулею $\boldsymbol{n}$, яка становить $\boldsymbol{d}$. Ми оновимо наші оцінки відстаней від кулі **1** до усіх куль, з'єднаних з $\boldsymbol{k}$, таким чином. Наприклад, деяка куля $\boldsymbol{j}$ ще не піднялася з поверхні та з'єднана з кулею $\boldsymbol{k}$ ниткою довжиною $\boldsymbol{s}$. Тоді ми замінюємо оцінку відстані $\boldsymbol{x}$ від кулі **1** до кулі $\boldsymbol{j}$ на менше із значень $\boldsymbol{x}$ і $\boldsymbol{(d+s)}$. Після того, як ми оновили усі оцінки відстаней до куль, з'єднаних з $\boldsymbol{k}$, ми повинні знайти наступну кулю, яка підніметься з поверхні. Це куля на поверхні (назвемо її «куля $\boldsymbol{p}$»), у якої найменша оцінка відстані до кулі **1**. Ми можемо продовжити процес далі з кулею $\boldsymbol{p}$.

Алгоритм Дейкстри

Для формалізації алгоритму введемо такі *позначення*:

– **c(i,j)** – **вартість лінії** від вузла i до вузла j.

Якщо вузли не з'єднані безпосередньо, то c(i,j)=∞. Нехай c(i,j)=c(j,i). Але, можливо, й c(i,j)≠c(j,i). Алгоритм є працездатним в обох випадках.

– **D(*v*)** – **вартість шляху** від вузла-джерела до вузла-адресата v, в якого на даний момент (на поточній ітерації алгоритму) вартість мінімальна.

– **p(*v*)** – попередній вузол (сусідній з вузлом v) на поточному шляху з найменшою вартістю від джерела до вузла v.

– **N** – множина вузлів, для котрих на даної ітерації відомі шляхи з найменшою вартістю.

Алгоритм Дейкстри для вузла-джерела A:

1. Ініціалізація:

1.1. N={A}.

1.2. Для усіх вузлів *v*:

Якщо вузол *v* суміжний з вузлом A, тоді D(*v*)=c(A,*v*).

Інакше D(*v*)=∞.

2. Цикл:

2.1. Знайти вузол *w*, який не входить в множину N, такий що D(*w*) є мінімальною.

2.2. Додати *w* до N.

2.3. Оновити D(*v*) для усіх *v*, які є суміжними з *w* та не входять в N:

D(*v*)=min(D(*v*); D(*w*)+c(*w*,*v*)).

Нова вартість шляху до вузла v дорівнює попередньої вартості до v або вартості відомого найкоротшого шляху до w плюс вартість шляху від w до v.

3. Повторити цикл по усім вузлам в множині N.

Приклад:

За допомогою алгоритму Дейкстри необхідно визначити найкоротші маршрути між вузлом А та усіма іншими вузлами мережі, граф якої наведений на рисунку 6.14.

Виконання етапів знаходження найкоротших шляхів проілюстровано за допомогою таблиці 6.7.

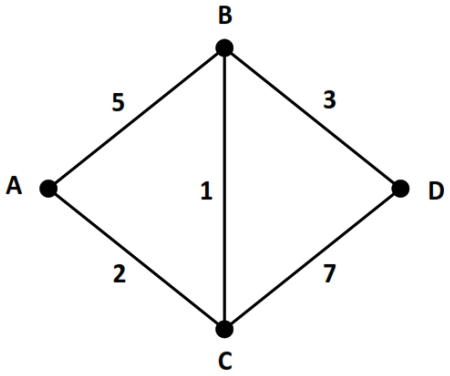

Рисунок 6.14 - Граф мережі

Таблиця 6.7 - Визначення найкоротших шляхів від вузла А до усіх інших вузлів мережі

Етап	**N**	**D(B), p(B)**	**D(C), p(C)**	**D(D), p(D)**
1	**2**	**3**	**4**	**5**
0	A	5, A	**2, A**	∞
1	AC	**3, C**		9, C
2	ACB			**6, B**
3	ACBD			

Етап 0.

Так як ми визначаємо найкоротші маршрути між ***вузлом А*** та усіма іншими вузлами мережі, то до множини вузлів N, для котрих на даної ітерації відомі шляхи з найменшою вартістю, поки входить лише ***вузол А***.

Далі необхідно для усіх вузлів, суміжних з ***вузлом А*** знайти та записати у таблицю вартість шляху D(*v*) від вузла-джерела А до вузла-адресата, а також вказати який вузол на даній (поточній) ітерації алгоритму є попереднім (сусіднім) p(*v*).

На графі мережі показано, що вузол A безпосередньо зв’язаний з вузлами B та C. Вартість шляхів до цих вузлів від вузла A відповідно дорівнює 5 та 2. Попереднім вузлом до вузлів B та C на даній ітерації алгоритму є вузол A. Тому, у стовпець 3 необхідно записати «5, A», а у стовпець 4 – «2, A». Вузол D не зв’язаний безпосередньо з вузлом A, тому вартість шляху між вузлами A та D поки що невідома. Позначимо її такою, що дорівнює нескінченості (стовпець 5).

Далі необхідно визначити найменшу з визначених вартість шляху в результаті виконання етапу «0». Звичайно, 2 менш ніж 5. Для наочності виділимо її жирним шрифтом. Можна виділити будь-яким іншим способом.

Найкоротший маршрут між вузлом A та вузлом C мережі вважаємо знайденим. Будь-які подальші обчислення не призведуть до зменшення вартості цього шляху. Тому, надалі не має сенсу переглядати шлях від вузла A до вузла C й протягом наступних етапів виконання алгоритму стовпець 4 таблиці заповняти не потрібно.

Етап 1.

Так як ми визначили найкоротший маршрут між вузлом A та вузлом C, то до множини N додаємо вузол C.

Далі необхідно для усіх вузлів, суміжних з ***вузлом C*** знайти та записати у таблицю вартість шляху D(v) від вузла-джерела A до вузла-адресата, а також вказати який вузол на даній (поточній) ітерації алгоритму є попереднім (сусіднім) p(v).

Для цього необхідно скористатись формулою

$$D(v)=\min(D(v); D(w)+c(w,v)).$$

Нова вартість шляху до вузла v дорівнює попередньої вартості до v або вартості відомого найкоротшого шляху до w плюс вартість шляху від w до v.

Так як ми вже визначили найкоротший шлях до вузла C, то на цьому етапі необхідно визначити можливість зменшення вартості шляхів від вузла A до усіх інших вузлів (B та D).

На графі мережі показано, що вузол C безпосередньо зв’язаний з вузлом B. На попередньому етапі вартість шляху до вузла B від вузла A була визначена рівною 5 (попередній вузол A). Вартість шляху до вузла C від вузла A була визначена як 2. Згідно графу мережі вартість шляху від вузла C до вузла B дорівнює 1. Тому

$$D(v)=\min(D(v); D(w)+c(w,v))= \min(5; (2+1))=3.$$

Таким чином, вартість шляху від вузла A до вузла B оновлюється й тепер дорівнює 3. Попередній вузол - C. Тому, у стовпець 3 необхідно записати «3, C».

Якщо в результаті обчислень попереднє значення вартості шляху виявиться менш ніж нове значення, то запис в таблиці не оновлюється, а тобто необхідно буде повторити в цьому стовпці попередній запис.

На графі мережі показано, що вузол C безпосередньо зв'язаний також з вузлом D. На попередньому етапі вартість шляху до вузла D від вузла A була визначена рівною нескінченості. На цьому етапі стає зрозумілим, що вартість шляху від вузла A до вузла D можна оновити й записати її нове значення у таблицю (стовпець 5). Вона буде дорівнювати вартості шляху від вузла A до вузла C плюс вартість шляху від вузла C до вузла D, тобто

$D(v)=\min(D(v); D(w)+c(w,v))= \min(\infty; (2+7))=9$.

Далі необхідно визначити найменшу з визначених вартість шляху в результаті виконання етапу «1». Звичайно, 3 менш ніж 9.

Найкоротший маршрут між вузлом A та вузлом B мережі вважаємо знайденим.

Етап 2.

Так як ми визначили найкоротший маршрут між вузлом A та вузлом B, то до множини N додаємо вузол B.

На цьому етапі необхідно визначити можливість зменшення вартості шляху від вузла A до вузла D.

На графі мережі показано, що вузол B безпосередньо зв'язаний з вузлом D. На попередньому етапі вартість шляху до вузла D від вузла A була визначена рівною 9 (попередній вузол C). Вартість шляху до вузла B від вузла A була визначена як 3. Згідно графу мережі вартість шляху від вузла B до вузла D дорівнює 3. Тому

$D(v)=\min(D(v); D(w)+c(w,v))= \min(9; (3+3))=6$.

Таким чином, вартість шляху від вузла A до вузла D оновлюється й тепер дорівнює 6. Попередній вузол - B. Тому, у стовпець 5 необхідно записати «6, B».

Найкоротші маршрути між вузлом A та усіма іншими вузлами мережі знайдені.

Етап 3.

До множини N додаємо вузол D.

У результаті виконання алгоритму множина N повинна містити **усі вузли мережі**.

Етап 4.

Залишилось побудувати граф мережі з врахуванням найкоротших маршрутів. Для цього потрібно проаналізувати сформовану таблицю 6.7 та з'єднати відповідні вузли. Згідно стовпця 3 попереднім вузлом до вузла B є вузол C. У стовпці 4 вказано, що через до вузла C зручно прийти через вузол A. Стовпець 5 вказує на з'єднання вузлів D та B.

Підсумковий граф мережі наведений на рисунку 6.15.

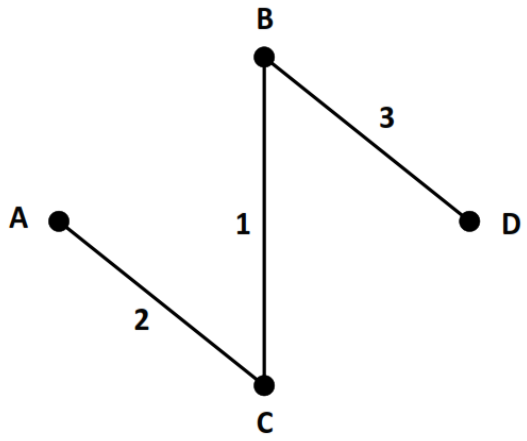

Рисунок 6.15 – Підсумковий граф мережі