ЛЕКЦІЯ 10. МЕРЕЖНІ ЗАСОБИ ОПЕРАЦІЙНИХ СИСТЕМ

План

16.1. Загальні принципи мережної підтримки
16.2. Реалізація стека протоколів Інтернет
16.3. Система імен DNS
16.4. Архітектура мережної підтримки Linux
16.5. Архітектура мережної підтримки Windows XP

16.1. Загальні принципи мережної підтримки

Під *мережею* розуміють набір комп'ютерів або апаратних пристроїв *(вузлів,* nodes), пов'язаних між собою каналами зв'язку, які можуть передавати інформацію один одному. Мережа має конкретну фізичну структуру (спосіб з'єднання вузлів, топологію), усі вузли підключають до мережі із використанням апаратного забезпечення, яке відповідає цій структурі. Звичайно мережа об'єднує обмежену кількість вузлів.

Під *інтернетом* (з малої літери) розуміють сукупність мереж, які використовують один і той самий набір *мережних протоколів* — правил, що визначають формат даних для пересилання мережею. Фізична структура окремих мереж, які входять до складу інтернету, може різнитися. Такі різнорідні мережі пов'язують одну з одною *маршрутизатори* (routers), які переадресовують пакети з однієї мережі в іншу, залежно від їхньої адреси призначення (маршрутизують їх) і при цьому перетворюють пакети між форматами відповідних мереж. Маршрутизатори підтримують міжмережну взаємодію (internetworking).

Відомий усім Інтернет (з великої літери)— це, фактично, сукупність пов'язаних між собою інтернетів, відкритих для публічного доступу, які використовують визначений набір протоколів (стек протоколів TCP/IP) і охоплюють увесь світ.

16.1.1. Рівні мережної архітектури і мєрежні сервіси

Функції забезпечення зв'язку між вузлами є досить складними. Для спрощення їхньої реалізації широко використовують багаторівневий підхід — вертикальний розподіл мережних функцій і можливостей. Він дає змогу приховувати складність реалізації функцій зв'язку: кожен рівень приховує від вищих рівнів деталі реалізації своїх функцій та функцій, реалізованих на нижчих рівнях.

У разі використання цього підходу для кожного типу мереж проектують еталонну модель протоколів, що описує функції окремих рівнів і зв'язки між рівнями. Фактично ця модель визначає мережну архітектуру, а рівні є її складовими частинами.

Мережний сервіс — це набір операцій, які надає рівень мережної архітектури для використання її на вищих рівнях. Сервіси визначено як частину специфікації інтерфейсу рівня.

Розрізняють сервіси, *орієнтовані на з'єднання* (connection-oriented services), і *без з'єднань,* або *дейтаграмні* сервіси (connectionless services).

Сервіси, *орієнтовані на з'єднання*, реалізують три фази взаємодії із верхнім рівнем: встановлення з'єднання, передавання даних і розрив з'єднання. При цьому передавання даних на верхніх рівнях здійснюють у вигляді неперервного потоку байтів.

Дейтаграмні сервіси реалізують пересилання незалежних повідомлень, які можуть переміщатися за своїми маршрутами і приходити у пункт призначення в іншому порядку.

Приклади реалізації сервісів різного типу опишемо нижче. Зазначимо, що реалізацію сервісу на рівні ОС або у вигляді прикладної програми, що надає доступі до деякої системної функціональності через мережу, називають *мережною службою.* Далі вживатимемо цей термін для позначення конкретної програмної реалізації сервісу.

16.1.2. Мережні протоколи

Визначення мережного сервісу для конкретного рівня мережної архітектури описує функціональність цього рівня, але не задає її реалізацію. Реалізацію функціональності для конкретного сервісу визначають мережні протоколи.

Мережний протокол — це набір правил, що задають формат повідомлень, порядок обміну повідомленнями між сторонами та дії, необхідні під час передавання або приймання повідомлень.

Кажуть, що мережний протокол А працює поверх мережного протоколу В (А — протокол вищого рівня, а В — нижчого), коли пакети з інформацією, що відповідають протоколу А, під час передавання мережею розміщені всередині пакетів протоколу В. Процес розміщення одних пакетів усередині інших називають *інкапсуляцією пакетів* (packet encapsulation). У разі приходу пакета за призначенням відповідне програмне забезпечення по черзі «знімає конверти», переглядаючи заголовки пакетів, приймаючи рішення на основі їхнього вмісту і вилучаючи їх. Процес визначення адресата пакета за інформацією із його заголовків називають *демультиплексуванням пакетів* (packet demultiplexing). Мережний протокол надає два інтерфейси.

1. *Однорівневий,* або *інтерфейс протоколу* (peer-to-peer interface) призначений для організації взаємодії із реалізацією протоколу того самого рівня на віддаленому мережному вузлі. Це найважливіший інтерфейс протоколу, що реалізує безпосереднє передавання даних на віддалений вузол. Такий інтерфейс звичайно забезпечують заголовком пакета, який доповнюють реалізацією цього протоколу перед передаванням пакета мережею.

2. *Інтерфейс сервісу* (service interface) призначений для взаємодії із засобами вищого рівня; за його допомогою фактично реалізують мережний сервіс. Інтерфейс сервісу забезпечують правилами інкапсуляції пакетів вищого рівня в пакети цього протоколу.

Набір протоколів різного рівня, що забезпечують реалізацію певної мережної архітектури, називають стеком протоколів (protocol stack) або набором протоколів (protocol suite).

16.2. Реалізація стека протоколів Інтернет

Сукупність протоколів, які лежать в основі сучасного Інтернету, називають *набором протоколів Інтернету* (Internet Protocol Suite, IPS) або *стеком протоколів TCP/IP* за назвою двох основних протоколів. У цьому розділі наведемо основні характеристики такого набору та особливості його реалізації у сучасних ОС.

16.2.1. Рівні мережної архітектури TCP/IP

Мережна архітектура TCP/IP має чотири рівні.

Канальний рівень (data link layer) відповідає за передавання кадру даних між будь-якими вузлами в мережах із типовою апаратною підтримкою (Ethernet, FDDI тощо) або між двома сусідніми вузлами у будь-яких мережах (SLIP, PPP). При цьому забезпечуються формування пакетів, корекція апаратних помилок, спільне використання каналів. Крім того, на більш низькому рівні він забезпечує передавання бітів фізичними каналами, такими як коаксіальний кабель, кручена пара або оптоволоконний кабель (іноді для опису такої взаємодії виділяють окремий *фізичний рівень* — physical layer).

Перш ніж перейти до наступного рівня, дамо два означення. *Хостом* (host) є вузол мережі, де використовують стек протоколів TCP/IP. *Мережним інтерфейсом* (network interface) є абстракція віртуального пристрою для зв'язку із мережею, яку надає програмне забезпечення канального рівня. Хост може мати декілька мережних інтерфейсів, зазвичай вони відповідають його апаратним мережним пристроям.

На *мережному рівні* (network layer) відбувається передавання пакетів із використанням різних транспортних технологій. Він забезпечує доставлення даних між мережними інтерфейсами будь-яких хостів у неоднорідній мережі з довільною топологією, але при цьому не бере на себе жодних зобов'язань щодо надійності передавання даних. На цьому рівні реалізована адресація інтерфейсів і маршрутизація пакетів. Основним протоколом цього рівня у стеку TCP/IP є IP (Internet Protocol).

Транспортний рівень (transport layer) реалізує базові функції з організації зв'язку між *процесами,* що виконуються на віддалених хостах. У стеку TCP/IP на цьому рівні функціонують протоколи *TCP* (Transmission Control Protocol) і *UDP* (User Datagram Protocol). TCP забезпечує надійне передавання повідомлень між віддаленими процесами користувача за рахунок утворення віртуальних з'єднань. UDP забезпечує ненадійне передавання прикладних пакетів (подібно до IP), виконуючи винятково функції сполучної ланки між IP і процесами користувача (далі на ньому зупинятися не будемо).

Прикладний рівень (application layer) реалізує набір різноманітних мережних сервісів, наданих кінцевим користувачам і застосуванням. До цього рівня належать протоколи, реалізовані різними мережними застосуваннями (службами), наприклад, HTTP (основа організації Web), SMTP (основа організації пересилання електронної пошти).

Основна відмінність прикладного рівня полягає в тому, що у більшості випадків його підтримка реалізована в режимі користувача (звичайно за це відповідають різні прикладні програми-сервери), а підтримка інших рівнів - у ядрі ОС. Завдання мережної служби прикладного рівня — реалізувати сервіс для кінцевого користувача (пересилання електронної пошти, передавання файлів тощо), який не має інформації про особливості переміщення даних мережею. Інші рівні, навпаки, не мають інформації про особливості застосувань, які обмінюватимуться даними за їхньою допомогою.

16.2.2. Канальний рівень

Реалізація канального рівня звичайно включає драйвер мережного пристрою ОС і апаратний мережний пристрій та приховує від програмного забезпечення верхнього рівня та прикладних програм деталі взаємодії з фізичними каналами, надаючи їм абстракцію мережного інтерфейсу. Передавання даних мережею у програмному забезпеченні верхнього рівня відбувається між мережними інтерфейсами.

Як зазначено вище, кількість мережних інтерфейсів звичайно співвідноситься з кількістю мережних апаратних пристроїв хоста. Крім того, виділяють спеціальний *інтерфейс зворотного зв'язку* (loopback interface); усі дані, передані цьому інтерфейсу, надходять на вхід реалізації стека протоколів того самого хоста.

16.2.3. Мережний рівень

У цьому розділі йтиметься про особливості протоколів мережного рівня.

Протокол IPv4

Протокол IP надає засоби доставлення дейтаграм неоднорідною мережею без встановлення з'єднання. Він реалізує доставлення за заданою адресою, але при цьому надійність, порядок доставлення і відсутність дублікатів не гарантовані. Усі засоби щодо забезпечення цих характеристик реалізуються у протоколах вищого рівня (наприклад, TCP).

Кожний мережний інтерфейс в IP-мережі має унікальну адресу. Такі адреси називають *IP-адресами.* Стандартною версією IP, якою користуються від початку 80-х років XX століття, є IP версії 4 (IPv4), де використовують адреси завдовжки 4 байти. їх зазвичай записують у крапково-десятковому поданні (чотири десяткові числа, розділені крапками, кожне з яких відображає один байт адреси). Прикладом може бути 194.41.233.1. Спеціальну адресу зворотного зв'язку 127.0.0.1 (loopback

address) присвоюють інтерфейсу зворотного зв'язку і використовують для зв'язку із застосуваннями, запущеними на локальному хості.

Як зазначалося, IP доставляє дейтаграми мережному інтерфейсу. Пошук процесу на відповідному хості забезпечують протоколи транспортного рівня (наприклад, TCP). Пакети цих протоколів інкапсулюють в IP-дейтаграми.

Протокол IPv6

Суттєвим недоліком протоколу IPv4 є незначна довжина IP-адреси. Кількість адрес, які можна відобразити за допомогою 32 біт, є недостатньою з огляду на сучасні темпи росту Інтернету. Сьогодні нові IP-адреси виділяють обмежено.

Для вирішення цієї проблеми було запропоновано нову реалізацію IP-протоколу — *IP версії 6* (IPv6), основною відмінністю якої є довжина адреси — 128 біт (16 байт).

Інші протоколи мережного рівня

Крім IP, на мережному рівні реалізовано й інші протоколи. Для забезпечення мережної діагностики застосовують протокол ICMP (Internet Control Message Protocol), який використовують для передавання повідомлень про помилки під час пересилання IP-дейтаграм, а також для реалізації найпростішого *луна-протпоколу,* що реалізує обмін запитом до хосту і відповіддю на цей запит. ICMP-повідомлення інкапсулюють в IP-дейтаграми.

Більшість сучасних ОС мають утиліту ping, яку використовують для перевірки досяжності віддаленого хоста. Ця утиліта використовує луна-протокол у рамках ICMP.

Підтримка мережного рівня

Засоби підтримки мережного рівня, як зазначалося, є частиною реалізації стека протоколів у ядрі ОС. Головними їхніми завданнями є інкапсуляція повідомлень транспортного рівня (наприклад, TCP) у дейтаграми мережного рівня (наприклад, IP) і передавання підготовлених дейтаграм драйверу мережного пристрою, отримання дейтаграм від драйвера мережного пристрою і демультиплексування повідомлень транспортного рівня, маршрутизація дейтаграм.

16.2.4. Транспортний рівень

Темою цього розділу будуть особливості реалізації протоколів транспортного рівня на прикладі TCP.

Протокол TCP

Пакет з TCP-заголовком називають *TCP-сегментом.* Основні характеристики протоколу TCP такі.

Підтримка комунікаційних каналів між клієнтом і сервером, які називають *з'єднаннями* (connections). TCP-клієнт встановлює з'єднання з конкретним сервером, обмінюється даними з сервером через це з'єднання, після чого розриває його.

Забезпечення надійності передавання даних. Коли дані передають за допомогою TCP, потрібне підтвердження їхнього отримання. Якщо воно не отримане впродовж певного часу, пересилання даних автоматично повторюють, після чого протокол знову очікує підтвердження. Час очікування зростає зі збільшенням кількості спроб. Після певної кількості безуспішних спроб з'єднання розривають. Неповного передавання даних через з'єднання бути не може: або воно надійно пересилає дані, або його розривають.

Встановлення послідовності даних (data sequencing). Для цього кожний сегмент, переданий за цим протоколом, супроводжує номер послідовності (sequence number). Якщо сегменти приходять у невірному порядку, TCP на підставі цих номерів може переставити їх перед тим як передати повідомлення в застосування.

Керування потоком даних (flow control). Протокол TCP повідомляє віддаленому застосуванню, який обсяг даних можливо прийняти від нього у будь-який момент часу. Це значення називають *оголошеним вікном* (advertised win-dow), воно дорівнює обсягу вільного простору у буфері, призначеному для отримання даних. Вікно динамічно змінюється: під час читання застосуванням даних із буфера збільшується, у разі надходження даних мережею — зменшується. Це гарантує, що буфер не може переповнитися. Якщо буфер заповнений повністю, розмір вікна зменшують до нуля. Після цього TCP, пересилаючи дані, очікуватиме, поки у буфері не вивільниться місце.

TCP-з'єднання є *повнодуплексними* (full-duplex). Це означає, що з'єднання у будь-який момент часу можна використати для пересилання даних в обидва боки. TCP відстежує номери послідовностей і розміри вікон для кожного напрямку передавання даних.

Порти

Для встановлення зв'язку між двома процесами на транспортному рівні (за допомогою TCP або UDP) недостатньо наявності IP-адрес (які ідентифікують мережні інтерфейси хостів, а не процеси, що на цих хостах виконуються). Щоб розрізнити процеси, які виконуються на одному хості, використовують концепцію *портів* (ports).

Порти ідентифікують цілочисловими значеннями розміром 2 байти (від 0 до 65 535). Кожний порт унікально ідентифікує процес, запущений на хості: для того щоб TCP-сегмент був доставлений цьому процесові, у його заголовку зазначається цей порт. Процес-сервер звичайно використовує заздалегідь визначений порт, на який можуть вказувати клієнти для зв'язку із цим сервером. Для клієнтів порти зазвичай резервують динамічно (оскільки вони потрібні тільки за наявності з'єднання, щоб сервер міг передавати дані клієнтові).

Для деяких сервісів за замовчуванням зарезервовано конкретні номери портів у діапазоні від 0 до 1023 (відомі порти, well-known ports); наприклад, для протоколу HTTP (веб-серверів) це порт 80, а для протоколу

SMTP — 25. В UNIX-системах відомі порти є привілейованими — їх можуть резервувати тільки застосування із підвищеними правами. Відомі порти розподіляються централізовано, подібно до IP-адрес.

Якщо порт зайнятий (зарезервований) деяким процесом, то жодний інший процес на тому самому хості повторно зайняти його не зможе.

Підтримка транспортного рівня

Засоби підтримки транспортного рівня у ядрі призначені для реалізації сервісів, обумовлених цим рівнем. Вони інкапсулюють повідомлення прикладного рівня у сегменти або дейтаграми транспортного рівня, забезпечують необхідні характеристики відповідного протоколу (для TCP до них належать надійність, керування потоком даних тощо), отримують сегменти або дейтаграми від засобів підтримки мережного рівня і демультиплексують їх. Крім того, ці засоби надають інтерфейс системних викликів для використання у прикладних програмах (інтерфейс сокетів, про який ітиметься у розділі 16.4).

16.2.5. Передавання даних стеком протоколів Інтернету

Мережні протоколи стека TCP/IP можна використовувати для зв'язку між рівноправними сторонами, але найчастіше такий зв'язок відбувається за принципом «клієнт-сервер», коли одна сторона (сервер) очікує появи дейтаграм або встановлення з'єднання, а інша (клієнт) відсилає дейтаграми або створює з'єднання.

Розглянемо основні етапи процесу обміну даними між клієнтом і сервером із використанням протоколу прикладного рівня, що функціонує в рамках стека протоколів TCP/IP. Як приклад такого протоколу візьмемо HTTP, при цьому сторонами, що взаємодіють, будуть веб-браузер (клієнт) і веб-сервер. Припустимо, що локальний комп'ютер зв'язаний з Інтернетом за допомогою мережного пристрою Ethernet. Для простоти вважатимемо, що під час передавання повідомлення не піддають фрагментації.

1. Застосування-клієнт (веб-браузер) у режимі користувача формує HTTP-запит до веб-сервера. Формат запиту визначений протоколом прикладного рівня (HTTP), зокрема у ньому зберігають шлях до потрібного документа на сервері. Після цього браузер виконує ряд системних викликів (визначених інтерфейсом сокетів, який розглянемо у розділі 16.4). При цьому у ядро ОС передають вміст HTTP-запиту, IP-адресу комп'ютера, на якому запущено веб-сервер, і номер порту, що відповідає цьому серверу.

Далі перетворення даних пакета відбувається в ядрі.

2. Спочатку повідомлення обробляють засобами підтримки протоколу транспортного рівня (TCP). У результаті його доповнюють TCP-заголовком, що містить номер порту веб-сервера та інформацію, необхідну для надійного пересилання даних (номер послідовності тощо). HTTP-запит інкапсулюється у TCP-сегмент та стає корисним навантаженням (payload) - даними, які пересилають для обробки в режимі користувача.

3. TCP-сегмент обробляють засобами підтримки протоколу мережного рівня (IP). При цьому він інкапсулюється в IP-дейтаграму (його доповнюють IP-заголовком, що містить IP-адресу віддаленого комп'ютера та іншу інформацію, необхідну для передавання мережею).

4. IP-дейтаграма надходить на рівень драйвера мережного пристрою (Ethernet), який додає до неї інформацію (заголовок і трейлер), необхідну для передавання за допомогою Ethernet-пристрою. Пакет з Ethernet-інформацією називають Ethernet-фрейм. Фрейм передають мережному пристрою, який відсилає його мережею. Фрейм містить адресу призначення у Ethernet, що є адресою мережної карти комп'ютера в тій самий локальній мережі (або іншого пристрою, який може переадресувати пакет далі в напрямку до місця призначення). Апаратне забезпечення Ethernet забезпечує реалізацію передавання даних фізичною мережею у вигляді потоку бітів.

Дотепер пакет переходив від засобів підтримки протоколів вищого рівня до протоколів нижчого. Кажуть, що пакет опускався у стеку протоколів.

5. Тепер пакет переміщатиметься мережею. При цьому можуть здійснюватися різні його перетворення. Наприклад, коли Ethernet-фрейм доходить до адресата в мережі Ethernet, відповідне програмне або апаратне забезпечення виділяє IP-дейтаграму із фрейму, за IP-заголовком визначає, яким каналом відправляти її далі, інкапсулює дейтаграму відповідно до характеристик цього каналу (наприклад, знову в Ethernet-фрейм) і відсилає її в наступний пункт призначення. На шляху повідомлення може перейти в мережі, зв'язані модемам і тоді формат зовнішньої оболонки буде змінено (наприклад, у формат протоколів SLIP або PPP), але вміст (IP-дейтаграми) залишиться тим самим.

Зрештою, пакет доходить до адресата. Його формат залежить від мережного апаратного забезпечення, встановленого на сервері. Якщо сервер теж підключений до мережі за допомогою мережного адаптера Ethernet, він отримає Ethernet-фрейм, подібний до відісланого клієнтом. Далі відбувається декілька етапів демультиплексування пакетів. Кажуть, що пакет піднімається у стеку протоколів.

6. Драйвер мережного пристрою Ethernet виділяє IP-дейтаграму із фрейму і передає її засобам підтримки протоколу IP.

7. Засоби підтримки IP перевіряють IP-адресу в заголовку, і, якщо вона збігається з локальною IP-адресою (тобто IP-дейтаграма дійшла за призначенням), виділяють TCP-сегмент із дейтаграми і передають його засобам підтримки TCP.

8. Засоби підтримки TCP визначають застосування-адресат за номером порту, заданим у TCP-заголовку (це веб-сервер, що очікує запитів від клієнтів). Після цього виділяють HTTP-запит із TCP-сегмента і передають його цьому застосуванню для обробки в режимі користувача.

9. Сервер обробляє HTTP-запит (наприклад, відшукує на локальному диску відповідний документ).

16.3. Система імен DNS

16.3.1. Загальна характеристика DNS

Доменна система імен (Domain Name System, DNS) — це розподілена база даних, яку застосування використовують для організації відображення символьних імен хостів (доменних імен) на IP-адреси. За допомогою DNS завжди можна знайти IP-адресу, що відповідає заданому доменному імені. Розподіленість DNS полягає в тому, що немає жодного хосту в Інтернеті, який би мав усю інформацію про це відображення. Кожна група хостів (наприклад, та, що пов'язує всі комп'ютери університету) підтримує свою власну базу даних імен, відкриту для запитів зовнішніх клієнтів та інших серверів. Підтримку бази даних імен здійснюють за допомогою застосування, яке називають DNS-сервером або сервером імен (name server).

Доступ до DNS з прикладної програми здійснюють за допомогою розпізнавача (resolver) — клієнта, який звертається до DNS-серверів для перетворення доменних імен в IP-адреси (цей процес називають *розв'язанням доменних імен* -domain name resolution). Звичайно розпізнавач реалізований як бібліотека, компонована із застосуваннями. Він використовує конфігураційний файл (в UNIX-системах це — /etc/resolv.conf), у якому зазначені IP-адреси локальних серверів імен. Якщо застосування потребує розв'язання доменного імені, код розпізнавача відсилає запит на локальний сервер імен, отримує звідти інформацію про відповідну IP-адресу і повертає її у застосування.

Зазначимо, що і розпізнавач, і сервер імен зазвичай виконуються в режимі користувача (щодо серверів імен це не завжди справедливо: так, у Windows-системах частина реалізації такого сервера виконується в режимі ядра). Стек TCP/IP у ядрі інформацією про DNS не володіє.

В UNIX-системах реалізація сервера імен є окремим продуктом, який називають bind.

16.3.2. Простір імен DNS

Простір імен DNS є ієрархічним. Кожний вузол супроводжує символьна позначка. Коренем дерева є вузол із позначкою нульової довжини. Доменне ім'я будь-якого вузла дерева — це список позначок, починаючи із цього вузла (зліва направо) і до кореня, розділених символом «крапка». Доменні імена мають бути унікальними.

Доменом (domain) називають піддерево ієрархічного простору імен. Для позначення домену (яке ще називають суфіксом домену) використовують доменне ім'я кореня цього піддерева: так, хост www.kpi.kharkov.ua. належить домену із суфіксом kpi.kharkov.ua., той, у свою чергу, — домену із суфіксом kharkov.ua. і т. д

Доменне ім'я, що завершується крапкою, називають *повним доменним іменем* (Fully Qualified Domain Name, FQJDN). Якщо крапка наприкінці імені відсутня, вважають, що це ім'я може бути доповнене (до нього може бути доданий суфікс відповідного домену). Такі імена можуть використовуватись у рамках домену.

Серед доменів верхнього рівня (суфікс для яких не містить крапок, окрім кінцевої) виділяють усім відомі com, edu, org тощо, а також домени для країн (**ua** для України). Є спеціальний домен агра, який використовують для зворотного перетворення IP-адрес у DNS-імена.

16.3.3. Розподіл відповідальності

Розподіл відповідальності за зони DNS-дерева — найважливіша характеристика доменної системи імен. Немає жодної організації або компанії, яка б керувала відображенням для всіх позначок дерева. Є спеціальна організація (Network Information Center, NIC), що керує доменом верхнього рівня і делегує відповідальність іншим організаціям за інші зони. Зоною називають частину DNS-дерева, що адмініструється окремо. Прикладом зони є домен другого рівня (наприклад, kharkov.ua). Багато організацій розділяють свої зони на менші відповідно до доменів наступного рівня (наприклад, kpi.kharkov.ua, kture.kharkov.ua тощо), аналогічним чином делегуючи відповідальність за них. У цьому разі зоною верхнього рівня вважають частину домена, що не включає виділені в ній зони.

Після делегування відповідальності за зону для неї необхідно встановити кілька серверів імен (як мінімум два — основний і резервний). Під час розміщення в мережі нового хоста інформація про нього повинна заноситься у базу даних основного сервера відповідної зони. Після цього інформацію автоматично синхронізують між основним і резервним серверами.

16.3.4. Отримання IP-адрес

Якщо сервер імен не має необхідної інформації, він її шукає на інших серверах. Процес отримання такої інформації називають ітеративним запитом (iterative query).

Розглянемо ітеративний запит отримання IP-адреси для імені www.kpi.kharkov.ua. Спочатку локальний сервер зв'язується із кореневим сервером імен (root name server), відповідальним за домен верхнього рівня (.). Станом на 2004 рік в Інтернеті було 13 таких серверів, кожен із них мав бути відомий усім іншим серверам імен. Кореневий сервер імен зберігає інформацію про сервери першого рівня. Отримавши запит на відображення імені, він визначає, що це ім'я належить не до його зони відповідальності, а до домену .ua, і повертає локальному серверу інформацію про адреси та імена всіх серверів відповідної зони. Далі локальний сервер звертається до одного із цих серверів з аналогічним запитом. Той сервер містить інформацію про те, що для зони kharkov.ua є свій сервер імен, у результаті

локальний сервер отримує адресу цього сервера. Процес повторюють доти, поки запит не надійде на сервер, відповідальний за домен kpi.kharkov.ua, що може повернути коректну IP-адресу.

16.3.5. Кешування IP-адрес

Кешування IP-адрес дозволяє значно зменшити навантаження на мережу. Коли сервер імен отримує інформацію про відображення (наприклад, IP-адресу, що відповідає доменному імені), він зберігає цю інформацію локально (у спеціальному кеші). Наступний аналогічний запит отримає у відповідь дані з кеша без звертання до інших серверів. Інформацію зберігають у кеші обмежений час. Зазначимо, що, коли для сервера не задана зона, за яку він відповідає, кешування є його єдиним завданням. Це поширений *сервер кешування* (caching-only server).

16.3.6. Типи DNS-ресурсів

Елемент інформації у базі даних DNS називають *ресурсним записом* (Resource Record, RR). Кожний такий запис має клас і тип. Для записів про відображення IP-адрес класом завжди є IN.

Розглянемо деякі типи DNS-ресурсів. Найважливішим із них є *A-запис,* що пов'язує повне доменне ім'я з IP-адресою. Саме на основі таких записів сервери імен повертають інформацію про відображення. У конфігураційному файлі сервера імен bind для домену kpi.kharkov.ua A-запис для www.kpi.kharkov.ua задають так:

```
www     IN    A  144.91.1.21
```

Ще одним типом запису є *CNAME-запис,* що задає *аліас доменного імені.* Одній і тій самий IP-адресі (A-запису) може відповідати кілька таких аліасів. Аліаси ftp.kpi.kharkov.ua і mail.kpi.kharkov.ua задають так:

```
ftp IN CNAME www mail     IN    CNAME     www
```

16.4. Архітектура мережної підтримки Linux

Тут зупинимося на деяких особливостях реалізації мережної підтримки в Linux, зокрема на базовій мережній архітектурі цієї ОС, особливостях підтримки інтерфейсу сокетів, а також формуванні пакетів під час реалізації пересилання і отримання даних.

16.4.1. Рівні мережної підтримки

Мережна архітектура Linux складається із кількох рівнів.

1. На верхньому рівні перебувають прикладні програми. Зв'язок цих програм із ядром здійснено через програмний інтерфейс сокетів.

2. Інтерфейс сокетів складається із двох рівнів: універсального рівня сокетів Берклі і рівня INET-сокетів, специфічного для підтримки TCP/IP.

3. Нижче від інтерфейсу сокетів розташовані засоби підтримки різних рівнів мережної архітектури:

- транспортного, які відповідають за підтримку протоколів транспортного рівня (TCP і UDP);

- мережного, які відповідають за підтримку протоколів мережного рівня (насамперед, IP);

- канального, які відповідають за формування пакетів канального рівня і за відображення адрес канального рівня на IP-адреси.

На найнижчому рівні перебувають драйвери мережних пристроїв, що підтримують пристрої конкретного типу.

Зазначимо, що хоча мережним пристроям в Linux, як і в більшості UNIX-систем, надають спеціальні імена, ці імена не відповідають файлам на файловій системі. Основне завдання драйвера мережного пристрою — пересилання даних із ділянок пам'яті ядра у пам'ять мережного пристрою і назад.

16.4.2. Підтримка інтерфейсу сокетів

Рівні підтримки інтерфейсу сокетів

Ядро Linux розрізняє два рівні підтримки інтерфейсу сокетів.

1. На верхньому перебуває *інтерфейс сокетів Берклі,* реалізований за допомогою універсальних *об'єктів сокетів Берклі* (структур типу socket). Об'єкти сокетів Берклі містять інформацію, що не залежить від реалізації конкретної мережної архітектури. Із такими об'єктами працюють реалізації системних викликів інтерфейсу сокетів.

2. На нижньому перебувають реалізації інтерфейсу сокетів, що залежать від конкретної мережної архітектури. Прикладом є *інтерфейс INET-сокетів,* що складається з об'єктів INET-сокетів (структур типу sock), які містять інформацію, специфічну для стека протоколів TCP/IP. Далі як архітектурно-залежні сокети розглядатимуться тільки INET-сокети.

Об'єкти сокетів

Об'єкти сокетів Берклі реалізовані як файли: для кожного сокета ядро створює індексний дескриптор на спеціальній файловій системі sockfs. Крім інформації, що належить до файлової системи, такий об'єкт містить:

- тип сокета (потоковий або дейтаграмний);

- стан сокета (зі з'єднанням або без нього);

- універсальні операції сокетів, що відповідають базовим системним викликам інтерфейсу сокетів Берклі (bindeO, listenO, connecteO, accepteO тощо);

- покажчик на відповідний об'єкт INET-сокета.

Об'єкт INET-сокета, в свою чергу, містить:

- локальні та віддалені IP-адреси, номери портів та іншу інформацію, необхідну для підтримки TCP/IP;

- черги пакетів, які одержані із сокета і очікують відсилання через сокет;

- чергу очікування для процесів, що очікують на цьому сокеті (наприклад, внаслідок виконання блокувального системного виклику recvO);

- реалізації універсальних операцій сокетів для стека TCP/IP.

Реалізація універсальних операцій об'єкта сокета Берклі звичайно зводиться до виклику відповідних методів пов'язаного з ним об'єкта INET-сокета

16.5. Архітектура мережної підтримки Windows XP

Компоненти мережної підтримки

Розглянемо основні компоненти мережної підтримки Windows XP.

Мережні API надають засоби для доступу до мережі із прикладних програм, незалежні від протоколів. Звичайно такі API частково реалізовані в режимі користувача (компоненти їхньої підтримки в режимі ядра — це драйвери, які називають драйверами мережних API). Серед мережних API найширше використовують інтерфейси поіменованих каналів, Windows Sockets (описаний далі в цьому розділі) та віддаленого виклику процедур (RPC).

Драйвери транспортних протоколів приймають IP-пакети від драйверів мережних API і обробляють запити, що містяться там, як вимагають реалізовані в них протоколи (TCP, IP тощо). Під час цієї обробки може знадобитися обмін даними через мережу, додавання або вилучення заголовків пакетів, взаємодія із драйверами мережних апаратних пристроїв. Драйвери протоколів відповідають за підтримку мережної взаємодії організацією повторного збирання пакетів, встановлення їхньої послідовності, відсилання повторних пакетів і підтверджень. Усі драйвери транспортних протоколів надають драйверам мережних API універсальний *інтерфейс транспортного драйвера* (Transport Driver Interface, TDI). Прикладом драйвера транспортного протоколу є драйвер підтримки TCP/IP (tcpip.sys).

Мініпорт-драйвери NDIS відповідають за організацію взаємодії драйверів транспортних протоколів і мережного апаратного забезпечення. Вони взаємодіють з іншими компонентами режиму ядра винятково за допомогою функцій спеціальної бібліотеки NDIS (ndis.sys), що реалізують універсальну *специфікацію інтерфейсу мережного драйвера* (Network Driver Interface Specification, NDIS). Деякі з цих функцій призначені для доступу із драйверів протоколів до засобів мініпорт-драйверів, інші мініпорт-драйвери викликають для безпосереднього доступу до мережних апаратних пристроїв через HAL. Прикладами мініпорт-драйверів NDIS є драйвери конкретних мережних адаптерів.

Спільне використання файлів і принтерів

Основою підтримки спільного використання файлів і принтерів є спеціальний протокол *спільної файлової системи Інтернету* (Common Internet File System, CIFS). Він визначає правила взаємодії з файловими клієнтами і серверами. Реалізація CIFS для Windows XP складається із двох частин — клієнтської (редиректор) і серверної (сервер), реалізованих як драйвери файлових систем.

Драйвер редиректора перехоплює запити на виконання файлового введення-виведення (наприклад, виклик функцій ReadFile(), WriteFile() або звертання до драйвера принтера), перетворює їх у CIFS-повідомлення і пересилає на віддалений хост, використовуючи драйвер транспортного протоколу. Драйвер сервера приймає повідомлення від драйвера протоколу, перетворює їх назад у запити введення-виведення і передає драйверу локальної файлової системи (наприклад, NTFS) або драйверу принтера. Повернення даних відбувається у зворотному порядку.

Інтерфейс керування драйверами CIFS реалізовано у вигляді фонових процесів Workstation (для редиректора) і Server (для сервера). Ці процеси перехоплюють запити на створення і зміну *мережних ресурсів* (network shares), які адміністратор системи відкрив для спільного використання (цими ресурсами можуть бути розділи диска, каталоги, принтери тощо). Для керування такими ресурсами можна використовувати утиліту net. exe, що входить у поставку Windows XP.

Висновки

Під час розгляду сучасних мережних архітектур використовують багаторівневий підхід. Найрозповсюдженішою моделлю мережної архітектури є TCP/IP, у якій виділяють чотири рівні: канальний (фізичний), мережний, транспортний і прикладний. Засоби підтримки перших трьох рівнів звичайно реалізують у ядрі ОС.

Реалізацією мережної архітектури TCP/IP є набір протоколів Інтернету (стек протоколів TCP/IP), який містить, зокрема, протокол IP на мережному рівні та протокол TCP на транспортному рівні. Цей набір протоколів підтримує більшість сучасних ОС.

Під час передавання пакета його спочатку опускають униз у стеку протоколів операційної системи хоста-відправника (при цьому відбувається його інкапсуляція в пакети протоколів нижнього рівня — спочатку транспортного, потім мережного, потім канального). Потім пакет канального рівня передають фізичною мережею, а після прибуття на хост-одержувач — піднімають у стеку протоколів ОС цього хоста (при цьому відбувається його послідовне демультиплексування із пакета канального рівня, мережного і транспортного). У результаті застосування-адресат отримує пакет протоколу прикладного рівня.

Ядро ОС може працювати лише з IP-адресами. Для того щоб можна було задавати символьні імена хостів, використовують розподілену систему імен DNS.

Основою для реалізації доступу із режиму користувача до засобів підтримки стека протоколу TCP/IP є програмний інтерфейс сокетів Берклі. Він розташований між прикладним і транспортним рівнями мережної архітектури TCP/IP. Його аналогом для Windows-систем є інтерфейс Windows Sockets.

Контрольні запитання та завдання

1. Які рівні мережної архітектури повинні бути реалізовані у програмному забезпеченні маршрутизатора?
2. Який із транспортних протоколів (TCP або UDP) краще використовувати як базовий протокол для передачі мультимедіа даних через Інтернет у реальному режимі (Real Audio і т. д.)?
3. Чи може користувач одночасно працювати з веб-браузером і клієнтом електронної пошти, використовуючи одне модемне з'єднання? Якщо така робота можлива, як розрізняти дані, призначені для кожного з цих застосувань?
4. Як використання кешування може підвищити надійність системи іменування (наприклад, DNS)?
5. Що таке мережний сервіс?
6. Чим дейтаграмні сервіси відрізняються від потокових?
7. Що таке протокол? Як протоколи розподілені за рівнями?
8. Опишіть роботу стеку протоколів Інтернету (HTTP, TCP/IP)
9. Опишіть призначення портів . Чому для встановлення зв'язку недостатньо IP-адрес?
10. Назвіть рівні підтримки інтерфейсу сокетів в Linux