ТЕМА 7. ІНСТРУМЕНТАЛЬНІ ТА ПРИКЛАДНІ ЗАСТОСУНКИ В ІНФОРМАЦІЙНІЙ ТА/АБО КІБЕРБЕЗПЕЦІ

7.1. Мережева модель OSI.
7.2. Основні протоколи стеку TCP/IP.
7.3. Віртуалізація (принципи, гіпервізори).
7.4. Архітектура комп'ютерів.

7.1. Мережева модель OSI

Комп'ютерна мережа – це об'єднання двох або більше комп'ютерів та/або комп'ютерного обладнання (сервери, маршрутизатори та ін.) з метою спільної обробки, зберігання або передачі даних.

За призначенням комп'ютерні мережі розподіляються на:
1. Обчислювальні.
2. Інформаційні.
3. Змішані.

Обчислювальні мережі призначені головним чином для рішення завдань користувачів з обміном даними між їхніми абонентами. **Інформаційні мережі** орієнтовані в основному на надання інформаційних послуг користувачам.

Змішані мережі сполучають функції перших двох.

Мережева модель – теоретичний опис принципів роботи набору взаємодіючих один з одним мережевих протоколів.

Модель звичайно ділиться на рівні, так щоб протоколи більш високого рівня використовували протоколи більш низького рівня, точніше, дані протоколу вищерозташованого рівня передавалися за допомогою протоколів нижчерозташованих рівнів – цей процес називають **інкапсуляцією**, в свою чергу, процес добування даних вищерозташованого рівня з даних нижчерозташованого називають **деінкапсуляцією**).

Моделі бувають як практичні (що використовуються в мережах, вони іноді заплутані і неповні, але вирішують поставлені задачі), так і теоретичні (що показують принципи реалізації мережевих моделей, вони більш наглядні і повні, але заради наглядності жертвують продуктивністю та деякими можливостями практичних моделей).

Багаторівнева мережева модель описує взаємодію між різними протоколами усередині кожного рівня, а також взаємодію з верхніми й нижніми рівнями.

Використання багаторівневої мережевої моделі дає ряд переваг:

– спрощує розробку протоколів, тому що протоколи, що працюють на певному рівні, визначають формат оброблюваних даних і надають інтерфейс до верхніх і нижніх рівнів;

– змушує постачальників конкуруючих продуктів створювати уніфіковані рішення;

– виключає можливості зміни технологій або функцій одного рівня без врахування наслідків для верхніх і нижніх рівнів;

– надає загальну мову для опису функцій мережевої взаємодії.

Найбільш відомі мережеві моделі:

– Модель OSI (вона ж ЕМ ВВС – еталонна модель взаємозв'язку відкритих систем) – еталонна теоретична модель, описана в міжнародних стандартах і ДСТ.

– Модель TCP/IP (Модель DOD) – модель, що використовується на практиці, прийнята для роботи в Інтернеті.

– Модель SPX/IPX – модель стека SPX/IPX (сімейство протоколів для ЛОМ)

– Модель AppleTalk – модель для мереж AppleTalk (протоколи для роботи мереж з устаткуванням Apple)

– Модель Fibre Channel – модель для високошвидкісних мереж Fibre Channel.

Мережева модель TCP/IP

Перша багаторівнева еталонна модель мережевої взаємодії була створена на початку 70-х років і називається моделлю мережі Інтернет. У ній визначені чотири обов'язкових категорії функцій, необхідних для успішної взаємодії. Архітектура протоколів TCP/IP побудована на основі цієї моделі. Тому модель мережі Інтернет звичайно називають моделлю TCP/IP.

Протоколи працюють один з одним у стеку – це означає, що протокол, що розташовується на рівні вище, працює «поверх» нижнього, використовуючи механізми інкапсуляції.

Мережева модель TCP/IP представлена на рисунку 7.1.

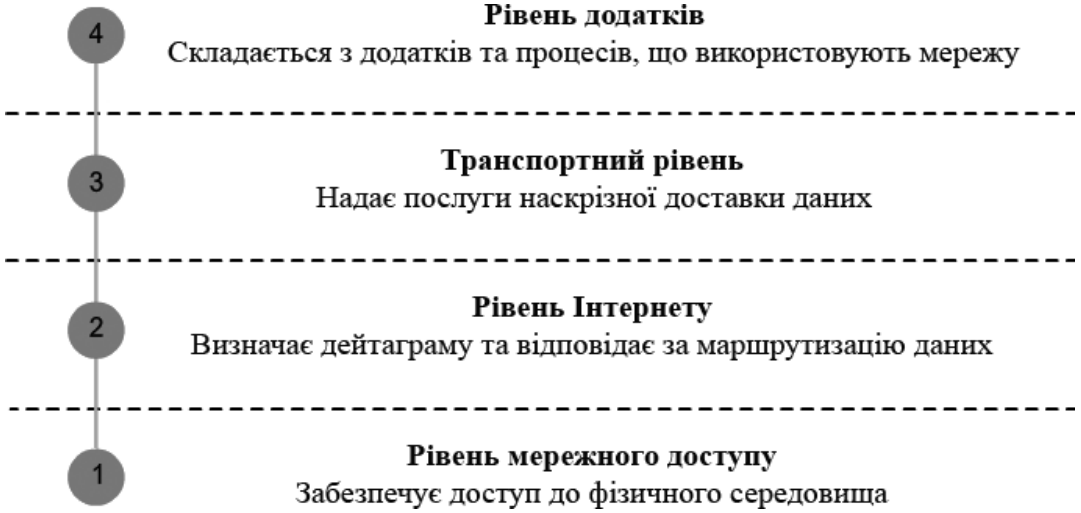

Рисунок 7.1 – Модель TCP/IP

Модель TCP/IP складається із чотирьох рівнів (знизу нагору):

- Рівень мережевого доступу (*Network Access*).
- Рівень Інтернету (*Internet*).
- Транспортний рівень (*Transport*).
- Рівень додатків (*Process/Application*).

Кожний із чотирьох рівнів моделі TCP/IP виконує свої функції.

Прикладний рівень (рівень додатків). Верхній рівень моделі, що включає протоколи, які обробляють дані користувачів і здійснюють керування обміном даними між додатками. На цьому рівні стандартизується представлення даних. Приклади протоколів: HTTP, SMTP, POP, IMAP, FTP, DNS, DHCP, Telnet.

Транспортний рівень. Містить протоколи для забезпечення цілісності даних при наскрізній передачі. Забезпечує керування ініціалізацією й закриттям з'єднань. Приклади протоколів: TCP та UDP.

Рівень Інтернету. Містить протоколи для маршрутизації повідомлень у мережі та служить для розміщення даних у дейтаграмі. Приклади протоколів: IP, NAT, ICMP, OSPF, RIP, BGP.

Рівень мережевого доступу. Нижній рівень моделі. Містить протоколи для фізичної доставки даних до мережевих пристроїв. Цей рівень розміщує дані в кадрі. Приклади протоколів: PPP, ARP, Ethernet, ATM.

Мережева модель OSI

Мережева модель OSI (англ. Open Systems Interconnection Basic Reference Model – базова еталонна модель взаємодії відкрітіх систем) – абстрактна мережева модель для комунікацій и розробки мережевих протоколів. Пропонує погляд на комп'ютерну мережу з точки зору вимірів. Кожний вимір обслуговує свою частину процесу взаємодії. Завдякі такій структурі спільна робота мережевого обладнання и програмного забезпечення стає набагато простішою й прозорішою.

Модель взаємодії відкритих систем була розроблена Міжнародною Організацією по Стандартизації (ISO) в 1984 році. На відміну від моделі TCP/IP, вона не описує взаємодій між окремими протоколами. Вона була створена як базова архітектура, яку розробники використовували для створення протоколів мережевої взаємодії. Хоча в далеко не всіх стеках протоколів у точності реалізовані всі сім рівнів моделі взаємодії відкритих систем, на сьогоднішній день вона вважається еталонною моделлю міжкомп’ютерних взаємодій.

У моделі OSI представлені всі функції або завдання, асоційовані з мережевими взаємодіями, а не тільки з певними протоколами TCP/IP. На відміну від моделі TCP/IP, у якій представлено тільки чотири рівні, модель OSI складається з семи більш специфічних рівнів (див. рисунок 7.2).

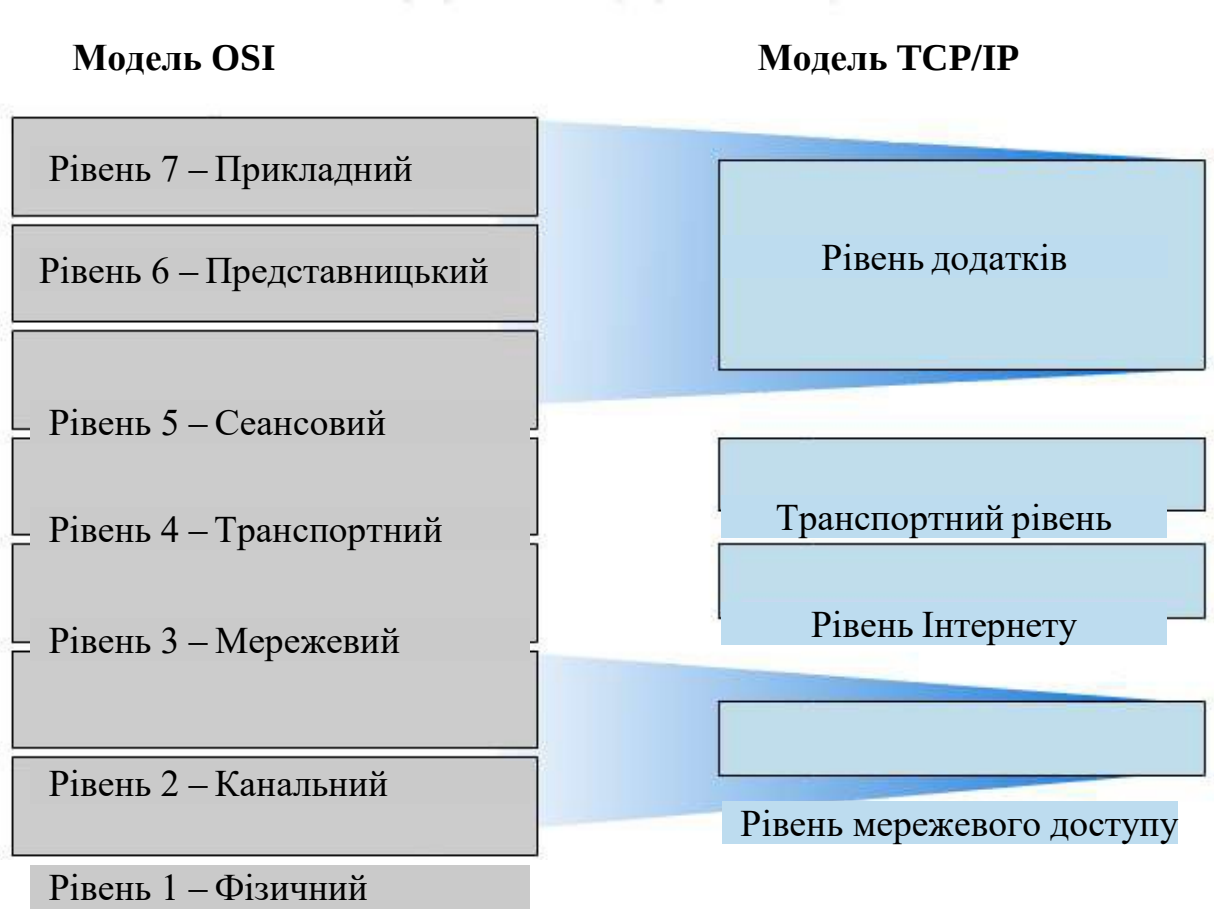

Рисунок 7.2 – Порівняння мережевих моделей OSI та TCP/IP

Кожному рівню присвоюється своя задача або група задач. Поділ функцій забезпечує незалежне функціонування кожного рівня в стеці. Наприклад, доступ до веб-сайту можливий як з портативного комп'ютера, підключеного до домашнього модему, так і з портативного комп'ютера за допомогою бездротового або мобільного телефону з підтримкою функцій бездротового доступу. Нижні рівні не впливають на ефективність роботи рівня додатка.

Точно так само нижні рівні не залежать від інших рівнів. Наприклад, на швидкість з'єднання з Інтернетом не впливає одночасний запуск декількох додатків, наприклад, електронна пошта, перегляд веб-сторінок, обмін миттєвими повідомленнями й завантаження музичних файлів.

7.2. Основні протоколи стеку TCP/IP

На рисунку 3.3 представлені рівні мережевої моделі OSI та їх призначення.

Група	Номер рівня	Назва рівня	Протоколи та технології	Розподілені мережеві компоненти даного рівня
Верхні рівні	7	Прикладний	DNS, NFS, DHCP, SNMP, FTP, TFTP, SMTP, POP3, IMAP, HTTP, Telnet	Додатки для контролю стану мережі, електронна пошта, веб-браузери, передача файлів тощо
	6	Представницький	SSL, оболонки і редиректори, MIME	
	5	Сеансовий	NetBIOS, інтерфейси програм, віддалені виклики процедури	
Нижні рівні	4	Транспортний	TCP та UDP	Механізни потокового відео та голосового зв’язку, списки фільтрів міжмережевих екранів
	3	Мережевий	IPv4, IPv6, IP NAT	IP-адресація, маршрутизація
	2	Канальний	Сімейство протоколів Ethernet, WLAN, Wi-Fi, ATM, PPP	Мережеві інтерфейсні плати та накопичувачі, мережева комутація, підключення до глобальної мережі
	1	Фізичний	Передача електричних сигналів, форми світлових хвиль, форми радіохвиль	Фізичне середовище передачі даних (мідна вита пара, оптоволоконний кабель, бездротові передавачі), концентратори і повторювачі

Рисунок 7.3 – Мережева модель OSI

Сім рівнів моделі OSI можна розділити на дві групи: верхні й нижні.

Верхніми рівнями часто називають усе, що перебуває вище транспортного рівня моделі OSI. Верхні рівні відносяться до роботи додатків і звичайно реалізуються тільки на програмному рівні. У рамках моделі OSI найвищий рівень – це прикладний рівень і він найближче розташований до кінцевого користувача.

Нижні рівні моделі OSI відносяться до передачі даних. Фізичний та канальний рівні реалізовані на апаратному й програмному рівнях. Фізичний рівень ближче всього до фізичного середовища передачі, до мережевих дротів. Він фактично поміщає інформацію в середовище.

Кінцеві станції, наприклад, клієнти й сервери, звичайно працюють на всіх семи рівнях. Мережеві пристрої працюють тільки на нижніх рівнях. Концентратори – це рівень 1, комутатори – рівні 1 і 2, маршрутизатори – рівні від 1 до 3, мережеві екрани – рівні 1, 2, 3 і 4.

Будь-який протокол моделі OSI повинен взаємодіяти або із протоколами свого рівня, або із протоколами на одиницю вище й/або нижче свого рівня. Взаємодії із протоколами свого рівня називаються горизонтальними, а з рівнями на одиницю вище або нижче – вертикальними. Будь-який протокол моделі OSI може виконувати тільки функції свого рівня й не може виконувати функцій іншого рівня, що не виконується в протоколах альтернативних моделей.

На кожному з рівнів одиниці інформації називаються по-різному. На фізичному рівні найменша одиниця інформації – біт. На канальному рівні інформація об'єднана у фрейми (або пакети). На мережевому рівні говорять про дейтаграми. На транспортному рівні одиницею вимірювання є сегмент. Прикладні рівні обмінюються повідомленнями. Пряма паралель із файловою системою на диску: локальні зміни намагніченості – біти об'єднані в сектори, що мають заголовки, сектори поєднуються в блоки, а ті, у свою чергу, у файли, що теж мають заголовки, що містять службову інформацію.

Таблиця 7.1 – Функції та одиниці вимірювання інформації різних рівнів моделі OSI

Рівень	**Функції**	**Тип даних**
Прикладний	Доступ до мережевих служб	Дані
Представницький	Подання й кодування даних	
Сеансовий	Керування сеансом зв'язку	
Транспортний	Прямий зв'язок між кінцевими пунктами й надійність	Сегменти
Мережевий	Визначення маршруту й логічна адресація	Пакети
Канальний	Фізична адресація	Кадри
Фізичний	Робота із середовищем передачі, сигналами й двійковими даними	Біти

Розглянемо кожний рівень моделі OSI більш докладно.

Фізичний рівень

Фізичний рівень має справу з передачею бітів по фізичних каналах зв'язку, таких, наприклад, як коаксіальний кабель, кручена пара, оптоволоконний кабель або цифровий територіальний канал. До цього рівня мають відношення характеристики фізичних середовищ передачі даних, такі як смуга пропускання, перешкодозахищеність, хвильовий опір і ін. На цьому ж рівні визначаються характеристики електричних сигналів, що передають дискретну інформацію, наприклад крутизна фронтів імпульсів, рівні напруги або струму переданого сигналу, тип кодування, швидкість передачі сигналів. Крім цього, тут стандартизуються типи роз'ємів і призначення кожного контакту.

Функції фізичного рівня реалізуються у всіх пристроях, підключених до мережі. З боку комп'ютера функції фізичного рівня виконуються мережевим адаптером або послідовним портом.

Прикладом протоколу фізичного рівня може служити специфікація 10Base-T технології Ethernet, що визначає як використовуваний кабель неекрановану кручену пару категорії 3 із хвильовим опором 100 Ом, роз'ємом RJ-45, максимальну довжину фізичного сегмента 100 м, манчестерский код для представлення даних у кабелі, а також деякі інші характеристики середовища й електричних сигналів.

Канальний рівень

На фізичному рівні просто пересилаються біти. При цьому не враховується, що в деяких мережах, у яких лінії зв'язку використовуються (розділяються) поперемінно декількома парами взаємодіючих комп'ютерів, фізичне середовище передачі може бути зайняте. Тому однією із задач канального рівня є перевірка доступності середовища передачі. Іншим завданням канального рівня є реалізація механізмів виявлення й корекції помилок. Для цього на канальному рівні біти групуються в набори, так звані *кадри (frames).* Канальний рівень забезпечує коректність передачі кожного кадру, поміщаючи спеціальну послідовність бітів у початок і кінець кожного кадру для його виділення, а також обчислює контрольну суму, обробляючи всі байти кадру певним способом і додаючи контрольну суму до кадру. Коли кадр приходить по мережі, одержувач знову обчислює контрольну суму отриманих даних і порівнює результат з контрольною сумою з кадру. Якщо вони збігаються, кадр вважається правильним і приймається. Якщо ж контрольні суми не збігаються, то фіксується помилка. Канальний рівень може не тільки виявляти помилки, але й виправляти їх за рахунок повторної передачі ушкоджених кадрів. Необхідно відзначити, що функція виправлення помилок не є обов'язковою для канального рівня, тому в деяких протоколах цього рівня вона відсутній, наприклад в Ethernet і Frame Relay.

У протоколах канального рівня, використовуваних у локальних мережах, закладена певна структура зв'язків між комп'ютерами й способи їхньої адресації.

Хоча канальний рівень і забезпечує доставку кадру між будь-якими двома вузлами локальної мережі, він це робить тільки в мережі із зовсім певною топологією зв'язків, саме тією топологією, для якої він був розроблений. До

таких типових топологій, підтримуваних протоколами канального рівня локальних мереж, відносяться загальна шина, кільце й зірка, а також структури, отримані з них за допомогою мостів і комутаторів. Прикладами протоколів канального рівня є протоколи Ethernet, Token Ring, FDDI.

У локальних мережах протоколи канального рівня використовуються комп'ютерами, мостами, комутаторами й маршрутизаторами. У комп'ютерах функції канального рівня реалізуються спільними зусиллями мережевих адаптерів і їхніх драйверів.

У глобальних мережах, які рідко мають регулярну топологію, канальний рівень часто забезпечує обмін повідомленнями тільки між двома сусідніми комп'ютерами, з'єднаними індивідуальною лінією зв'язку. Прикладами протоколів «точка-точка» (як часто називають такі протоколи) можуть служити широко розповсюджені протоколи PPP і LAP-B. У таких випадках для доставки повідомлень між кінцевими вузлами через всю мережу використовуються засоби мережевого рівня. Саме так організовані мережі X.25. Іноді в глобальних мережах функції канального рівня в чистому вигляді виділити важко, тому що в тому самому протоколі вони поєднуються з функціями мережевого рівня. Прикладами такого підходу можуть служити протоколи технологій ATM і Frame Relay.

У цілому канальний рівень являє собою досить потужний і закінчений набір функцій по пересиланню повідомлень між вузлами мережі. У деяких випадках протоколи канального рівня виявляються самодостатніми транспортними засобами й можуть допускати роботу поверх себе безпосередньо протоколів прикладного рівня або додатків, без залучення засобів мережевого й транспортного рівнів. Наприклад, існує реалізація протоколу керування мережею SNMP безпосередньо поверх Ethernet, хоча стандартно цей протокол працює поверх мережевого протоколу IP і транспортного протоколу UDP. Природно, що застосування такої реалізації буде обмеженим – вона не підходить для складених мереж різних технологій, наприклад Ethernet і X.25, і навіть для такої мережі, у якій у всіх сегментах застосовується Ethernet, але між сегментами існують петлевидні зв'язки. А от у двосегментній мережі Ethernet, об'єднаній мостом, реалізація SNMP над канальним рівнем буде цілком працездатною.

Проте для забезпечення якісного транспортування повідомлень у мережах будь-яких топологій і технологій функцій канального рівня виявляється недостатньо, тому в моделі OSI рішення цього завдання покладає на два наступні рівні – мережевий і транспортний.

Канальний рівень забезпечує передачу пакетів даних, що надходять від протоколів верхніх рівнів, вузлу призначення, адреса якого також указує протокол верхнього рівня. Протоколи канального рівня оформляють передані їм пакети в кадри власного формату, поміщаючи зазначену адресу призначення в одне з полів такого кадру, а також супроводжуючи кадр контрольною сумою. Протокол канального рівня має локальний сенс, він призначений для доставки кадрів даних, як правило, у межах мереж із простою топологією зв'язків і однотипною або близькою технологією, наприклад в односегментних мережах Ethernet або ж у багатосегментних мережах Ethernet і Token Ring ієрархічної

топології, розділених тільки мостами й комутаторами. У всіх цих конфігураціях адреса призначення має локальний сенс для даної мережі й не змінюється при проходженні кадру від вузла-джерела до вузла призначення. Можливість передавати дані між локальними мережами різних технологій пов'язана з тим, що в цих технологіях використовуються адреси однакового формату, до того ж виробники мережевих адаптерів забезпечують унікальність адрес незалежно від технології.

Іншою областю дії протоколів канального рівня є зв'язки типу «точка-точка» глобальних мереж, коли протокол канального рівня відповідальний за доставку кадру безпосередньому сусідові. Адреса в цьому випадку не має принципового значення, а на перший план виходить здатність протоколу відновлювати викривлені й загублені кадри, тому що погана якість територіальних каналів, особливо телефонних, часто вимагає виконання подібних дій.

Якщо ж перераховані вище умови не дотримуються, наприклад зв'язки між сегментами Ethernet мають петлевидну структуру, або поєднувані мережі використовують різні способи адресації, як це має місце в мережах Ethernet і X.25, то протокол канального рівня не може самостійно впоратися із задачею передачі кадру між вузлами й вимагає допомоги протоколу мережевого рівня.

Мережевий рівень

Мережевий рівень служить для утворення єдиної транспортної системи, що поєднує декілька мереж, причому ці мережі можуть використовувати зовсім різні принципи передачі повідомлень між кінцевими вузлами й мати довільну структуру зв'язків. Функції мережевого рівня досить різноманітні. Почнемо їхній розгляд на прикладі об'єднання локальних мереж.

Протоколи канального рівня локальних мереж забезпечують доставку даних між будь-якими вузлами тільки в мережі з відповідною типовою топологією, наприклад топологією ієрархічної зірки. Це дуже жорстке обмеження, що не дозволяє будувати мережі з розвинутою структурою, наприклад мережі, що поєднують кілька мереж підприємства в єдину мережу, або високонадійні мережі, у яких існують надлишкові зв'язки між вузлами. Можна було б ускладнювати протоколи канального рівня для підтримки петлевидних надлишкових зв'язків, але принцип розподілу обов'язків між рівнями приводить до іншого рішення. Щоб, з одного боку, зберегти простоту процедур передачі даних для типових топологий, а з іншого боку – допустити використання довільних топологий, вводиться додатковий мережевий рівень.

На мережевому рівні сам термін *мережа* наділяють специфічним значенням. У цьому випадку під мережею розуміється сукупність комп'ютерів, з'єднаних між собою відповідно до однієї зі стандартних типових топологій і використовуючих для передачі даних один із протоколів канального рівня, певний для цієї топології.

Усередині мережі доставка даних забезпечується відповідним канальним рівнем, а от доставкою даних між мережами займається мережевий рівень, що і підтримує можливість правильного вибору маршруту передачі повідомлення

навіть у тому випадку, коли характер структури зв'язків між складовими мережами відрізняється від прийнятого в протоколах канального рівня.

Мережі з'єднуються між собою спеціальними пристроями, що називаються маршрутизаторами. ***Маршрутизатор*** – це пристрій, що збирає інформацію про топологію мережевих з'єднань і на її підставі пересилає пакети мережевого рівня в мережу призначення. Щоб передати повідомлення від відправника, що перебуває в одній мережі, одержувачеві, що перебуває в іншій мережі, потрібно зробити деяку кількість *транзитних передач між мережами,* або *хопів (hop* – стрибок), щоразу вибираючи підходящий маршрут. Таким чином, маршрут являє собою послідовність маршрутизаторів, через які проходить пакет.

На рисунку 3.4 показані чотири мережі, зв'язані трьома маршрутизаторами. Між вузлами А та В даної мережі пролягають два маршрути: перший через маршрутизатори 1 та 3, а другий через маршрутизатори 1, 2 і 3.

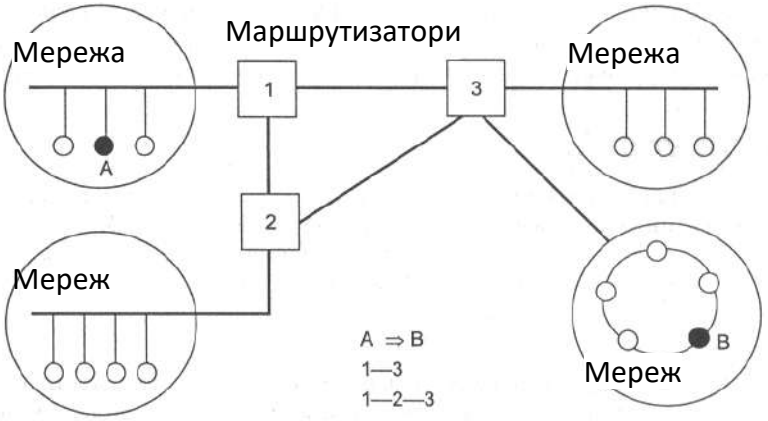

Рисунок 7.4 – Приклад складеної мережі

Проблема вибору найкращого шляху називається ***маршрутизацією***, і її рішення є однією з головних завдань мережевого рівня. Ця проблема ускладнюється тим, що самий короткий шлях не завжди найкращий. Часто критерієм при вибору маршруту є час передачі даних по цьому маршруту; він залежить від пропускної здатності каналів зв'язку та інтенсивності трафіку, що може змінюватися із часом. Деякі алгоритми маршрутизації намагаються пристосовуватися до змін навантаження, у той час як інші приймають рішення на основі середніх показників за тривалий час. Вибір маршруту може здійснюватися й за іншими критеріями, наприклад надійності передачі.

У загальному випадку функції мережевого рівня ширше, ніж функції передачі повідомлень по зв'язках з нестандартною структурою, які ми зараз розглянули на прикладі об'єднання декількох локальних мереж. Мережевий рівень вирішує також задачу узгодження різних технологій, спрощення адресації у великих мережах і створення надійних і гнучких бар'єрів на шляху небажаного трафіку між мережами.

Повідомлення мережевого рівня прийнято називати ***пакетами*** (packet). При організації доставки пакетів на мережевому рівні використовується поняття

«номер мережі». У цьому випадку адреса одержувача складається зі старшої частини – номера мережі й молодшої – номера вузла в цій мережі. Всі вузли однієї мережі повинні мати ту саму старшу частину адреси, тому терміну «мережа» на мережевому рівні можна дати й інше, більш формальне визначення: мережа – це сукупність вузлів, мережева адреса яких містить той самий номер мережі.

На мережевому рівні визначаються два види протоколів. Перший вид – *мережеві протоколи (routed protocols)* – реалізують просування пакетів через мережу. Саме ці протоколи звичайно мають на увазі, коли говорять про протоколи мережевого рівня. Однак часто до мережевого рівня відносять і інший вид протоколів, що називаються протоколами обміну маршрутною інформацією або просто *протоколами маршрутизації (routing protocols).* За допомогою цих протоколів маршрутизатори збирають інформацію про топологію міжмережевих з'єднань. Протоколи мережевого рівня реалізуються програмними модулями операційної системи, а також програмними й апаратними засобами маршрутизаторів.

На мережевому рівні працюють протоколи ще одного типу, які відповідають за відображення адреси вузла, використовуваного на мережевому рівні, у локальну адресу мережі. Такі протоколи часто називають *протоколами дозволу адрес (Address Resolution Protocol, ARP).* Іноді їх відносять не до мережевого рівня, а до канального, хоча тонкості класифікації не змінюють їхньої суті.

Транспортний рівень

На шляху від відправника до одержувача пакети можуть бути викривлені або загублені. Хоча деякі додатки мають власні засоби обробки помилок, існують і такі, які воліють відразу мати справу з надійним з'єднанням. Транспортний рівень забезпечує додаткам або верхнім рівням стека – прикладному й сеансовому – передачу даних з тим ступенем надійності, який їм потрібний. Модель OSI визначає п'ять класів сервісу, надаваних транспортним рівнем. Ці види сервісу відрізняються якістю надаваних послуг: терміновістю, можливістю відновлення перерваного зв'язку, наявністю засобів мультиплексування декількох з'єднань між різними прикладними протоколами через загальний транспортний протокол, а головне – здатністю до виявлення й виправлення помилок передачі, таких як викриволення, втрата й дублювання пакетів.

Вибір класу сервісу транспортного рівня визначається, з одного боку, тим, у якому ступені завдання забезпечення надійності вирішується самими додатками й протоколами більш високих, ніж транспортний, рівнів, а з іншого боку, цей вибір залежить від того, наскільки надійною є система транспортування даних у мережі, забезпечувана рівнями, розташованими нижче транспортного – мережевим, канальним і фізичним. Так, наприклад, якщо якість каналів передачі зв'язку дуже висока і ймовірність виникнення помилок, не виявлених протоколами більш низьких рівнів, невелика, то розумно скористатися одним з полегшених сервісів транспортного рівня, не обтяжених численними перевірками, квитуванням та іншими прийомами підвищення

надійності. Якщо ж транспортні засоби нижніх рівнів дуже ненадійні, то доцільно звернутися до найбільш розвиненого сервісу транспортного рівня, що працює, використовуючи максимум засобів для виявлення й усунення помилок, включаючи попереднє встановлення логічного з'єднання, контроль доставки повідомлень по контрольних сумах і циклічну нумерацію пакетів, встановлення тайм-аутів доставки й т.п.

Як правило, всі протоколи, починаючи із транспортного рівня й вище, реалізуються програмними засобами кінцевих вузлів мережі – компонентами їх мережевих операційних систем. Як приклад транспортних протоколів можна привести протоколи TCP і UDP стека TCP/IP і протокол SPX стека Novell.

Протоколи нижніх чотирьох рівнів узагальнено називають мережевим транспортом або транспортною підсистемою, тому що вони повністю вирішують завдання транспортування повідомлень із заданим рівнем якості в складених мережах з довільною топологією й різними технологіями. Три верхні рівні, що залишилися вирішують завдання надання прикладних сервісів на підставі наявної транспортної підсистеми.

Сеансовий рівень

Сеансовий рівень забезпечує керування взаємодією: фіксує, яка зі сторін є активною в даний момент, надає засоби синхронізації. Останні дозволяють вставляти контрольні точки в довгі передачі, щоб у випадку відмови можна було повернутися назад до останньої контрольної точки, а не починати все з початку. На практиці деякі додатки використовують сеансовий рівень, і він рідко реалізується у вигляді окремих протоколів, хоча функції цього рівня часто поєднують із функціями прикладного рівня й реалізують в одному протоколі.

Представницький рівень

Представницький рівень має справу з формою подання переданої по мережі інформації, не міняючи при цьому її змісту. За рахунок рівня подання інформація, передана прикладним рівнем однієї системи, завжди зрозуміла прикладному рівню іншої системи. За допомогою засобів даного рівня протоколи прикладних рівнів можуть перебороти синтаксичні розходження в поданні даних або ж розходження в кодах символів, наприклад кодів ASCII і EBCDIC. На цьому рівні може виконуватися шифрування й дешифрування даних, завдяки яким таємність обміну даними забезпечується відразу для всіх прикладних служб. Прикладом такого протоколу є протокол Secure SocketLayer (SSL), що забезпечує секретний обмін повідомленнями для протоколів прикладного рівня стека TCP/IP.

Прикладний рівень

Прикладний рівень – це в дійсності просто набір різноманітних протоколів, за допомогою яких користувачі мережі одержують доступ до поділюваних ресурсів, таких як файли, принтери або гіпертекстові веб-сторінки, а також організують свою спільну роботу, наприклад, по протоколу електронної пошти. Одиниця даних, якою оперує прикладний рівень, звичайно називається *повідомленням.*

Існує дуже велика розмаїтість служб прикладного рівня. Наведемо як приклад хоча б декілька найпоширеніших реалізацій файлових служб: NCP в

операційній системі Novell NetWare, SMB в Microsoft Windows NT, NFS, FTP і TFTP, що входять у стек TCP/IP.

Мережозалежні та мережонезалежні рівні **моделі OSI**

Функції всіх рівнів моделі OSI можуть бути віднесені до однієї із двох груп: або до функцій, що залежать від конкретної технічної реалізації мережі, або до функцій, орієнтованих на роботу з додатками.

Три нижніх рівні – фізичний, канальний і мережевий – є мережозалежними, тобто протоколи цих рівнів тісно пов'язані з технічною реалізацією мережі й використовуваним комунікаційним устаткуванням. Наприклад, перехід на устаткування FDDI означає повну зміну протоколів фізичного й канального рівнів у всіх вузлах мережі.

Три верхніх рівні – прикладний, представницький і сеансовий – орієнтовані на додатки й мало залежать від технічних особливостей побудови мережі. На протоколи цих рівнів не впливають які б то не було зміни в топології мережі, заміна устаткування або перехід на іншу мережеву технологію. Так, перехід від Ethernet на високошвидкісну технологію lOOVG-AnyLAN не вимагає ніяких змін у програмних засобах, що реалізують функції прикладного, представницького й сеансового рівнів.

Транспортний рівень є проміжним, він приховує деталі функціонування нижніх рівнів від верхніх. Це дозволяє розробляти додатки, що не залежать від технічних засобів безпосереднього транспортування повідомлень.

7.3. Віртуалізація (принципи, гіпервізори)

Віртуалізація – надання набору обчислювальних ресурсів або їхнього логічного об'єднання, абстраговане від апаратної реалізації і забезпечує при цьому логічну ізоляцію один від одного обчислювальних процесів, що виконуються на одному фізичному ресурсі.

Прикладом використання віртуалізації є можливість запуску кількох операційних систем на одному комп'ютері: при тому кожен з екземплярів таких гостьових операційних систем працює зі своїм набором логічних ресурсів (процесорних, оперативної пам'яті, пристроїв зберігання), надання яких із загального пулу, доступного на рівні обладнання, управляє хостова операційна система – гіпервізор. Також можуть бути піддані віртуалізації мережі передачі даних, мережі зберігання даних, платформне та прикладне програмне забезпечення (емуляція).

Види віртуалізації

Устаткування

– Емуляція – повна віртуалізація (віртуалізація всієї платформи); наприклад, QEMU або емулятори ігрових консолей.

Операційні системи

– Програмна віртуалізація:

- Динамічна трансляція; при динамічній (бінарній) трансляції проблемні команди гостьової операційної системи перехоплюються гіпервізором.
- Паравіртуалізація: операційна система взаємодіє з програмою гіпервізора, який надає їй гостьовий API замість використання безпосередньо таких ресурсів, як таблиця сторінок пам'яті.
- Вбудована віртуалізація.

– Апаратна віртуалізація – віртуалізація за допомогою спеціальної процесорної архітектури. На відміну від програмної віртуалізації, за допомогою цієї техніки можливе використання ізольованих гостьових систем, керованих гіпервізором безпосередньо.

– Віртуалізація на рівні операційної системи: робота кількох екземплярів простору користувача в рамках однієї ОС. Прикладами можуть бути Docker, LXC

Програмне забезпечення

– Віртуалізація додатків (також віртуалізація робочого оточення): робота окремих додатків у середовищі, відокремленому від основної ОС. Ця концепція тісно пов'язана з портативними програмами. Прикладами можуть бути: Citrix XenApp, Microsoft App-V.

– Віртуалізація сервісів: емуляція поведінки системних компонентів, необхідні запуску програми з метою налагодження та тестування (англ. *Application Under Test*). Замість віртуалізації компонентів повністю, ця технологія віртуалізує лише необхідні частини. Приклади: SoapUI, Parasoft Virtualize.

Пам'ять

– Віртуалізація пам'яті (англ. *memory virtualization*) – Об'єднанням оперативної пам'яті з різних ресурсів в єдиний масив. Реалізації: Oracle Coherence, GigaSpaces XAP.

– Віртуальна пам'ять – ізоляція адресного простору програми від адресного простору. Застосовується у всіх сучасних ОС.

Системи зберігання

– Віртуалізація зберігання даних, представлення набору фізичних носіїв як єдиного фізичного носія:

- Блокова віртуалізація.
- Файлова віртуалізація.

– Розподілена файлова система – будь-яка файлова система, яка дозволяє отримувати доступ до файлів з кількох пристроїв за допомогою комп'ютерної мережі.

– Віртуальна файлова система – рівень абстракції поверх конкретної реалізації файлової системи. Метою VFS є забезпечення однакового доступу клієнтських програм до різних типів файлових систем.

– Гіпервізор зберігання (англ. *storage hypervisor*) – програма, яка управляє віртуалізацією простору для зберігання даних і може об'єднувати різні фізичні простори в єдиний логічний масив [1].

– Віртуалізація пристроїв зберігання даних: віртуалізація жорсткого (логічний диск) або оптичного диска (наприклад, DAEMON Tools).

Бази даних

– Віртуалізація даних (англ. *data virtualization*) – подання даних в абстрактному вигляді, незалежно від систем управління та зберігання даних, що знаходяться нижче, а також їх структури. Це підхід до уніфікації даних з декількох джерел на одному рівні, щоб додатки, засоби звітності та кінцеві користувачі могли отримувати доступ до даних, не потребуючи докладних відомостей про вихідні джерела, місцезнаходження та структури даних. [2]

Мережа

– Віртуалізація мережі – процес об'єднання апаратних та програмних мережевих ресурсів у єдину віртуальну мережу:

- Зовнішня мережа, що з'єднує безліч в одну віртуальну.
- Внутрішня, що створює віртуальну мережу між програмними контейнерами всередині однієї системи.

– Віртуальна приватна мережа – забезпечення одного або кількох мережних з'єднань над іншою мережею.

Віртуалізація операційних систем

Для віртуалізації операційних систем застосовується серія підходів, які за типом реалізації поділяються на програмні та апаратні.

Програмна віртуалізація

Динамічна трансляція

При динамічній (*бінарній*) трансляції проблемні команди гостьової операційної системи перехоплюються гіпервізором. Після того, як ці команди замінюються на безпечні, відбувається повернення управління гостьовою системою.

Паравіртуалізація

Паравіртуалізація – техніка віртуалізації, коли гостьові операційні системи готуються до виконання у віртуалізованому середовищі, навіщо їх ядро трохи модифікується. Операційна система взаємодіє з програмою гіпервізора, який надає їй гостьовий API замість використання безпосередньо таких ресурсів, як таблиця сторінок пам'яті.

Метод паравіртуалізації дозволяє досягти більш високої продуктивності, ніж метод динамічної трансляції.

Метод паравіртуалізації застосовний лише в тому випадку, якщо гостьові операційні системи мають відкриті вихідні коди, які можна модифікувати відповідно до ліцензії, або гіпервізор і гостьова операційна система розроблені одним виробником з урахуванням можливості паравіртуалізації гостьової системи (хоча за умови, що під гіпервізором може бути запущений гіпервізор нижчого рівня, то й паравіртуалізації самого гіпервізора).

Вперше термін виник у проекті Denali.

Вбудована віртуалізація

Переваги:

– Спільне використання ресурсів декількома гостьовими операційними системами (каталоги, принтери тощо).

– Зручність інтерфейсу для вікон додатків з різних систем (вікна додатків, що перекриваються, однакова мінімізація вікон, як у хост-системі).

– При тонкому налаштуванні на апаратну платформу продуктивність мало відрізняється від оригінальної операційної системи. Швидке перемикання між системами (менше однієї секунди).

– Проста процедура поновлення гостьової операційної системи.

–Двостороння віртуалізація (додатки однієї системи запускаються до іншої та навпаки).

Реалізації: BlueStacks Multi-OS (MOS).

Апаратна віртуалізація

Переваги:

- Спрощення розробки програмних платформ віртуалізації за рахунок надання апаратних інтерфейсів керування та підтримки віртуальних гостьових систем. Це зменшує трудомісткість та час на розробку систем віртуалізації.

- Можливість збільшення швидкодії платформ віртуалізації. Управління віртуальними гостьовими системами здійснює безпосередньо невеликий проміжний шар програмного забезпечення, гіпервізор, що дає збільшення швидкодії.

- Покращується захищеність, з'являється можливість перемикання між кількома незалежними запущеними платформами віртуалізації на апаратному рівні. Кожна з віртуальних машин може працювати незалежно, у своєму просторі апаратних ресурсів повністю ізольовано один від одного. Це дозволяє усунути втрати швидкодії на підтримку хостової платформи та збільшити захищеність.

- Гостьова система стає не прив'язана до архітектури хостової платформи та реалізації платформи віртуалізації. Технологія апаратної віртуалізації уможливлює запуск 64-бітових гостьових систем на 32-бітових хостових системах (з 32-бітними середовищами віртуалізації на хостах).

Технології:

– Режим віртуального 8086 (застаріла).

– Intel VT (*VT-x, Intel Virtualization Technology for x86*).

– AMD-V.

Платформи, що використовують апаратну віртуалізацію:

– IBM LPAR.

– VMware.

– Hyper-V.

– Xen.

– KVM.

– Bhyve.

Контейнерна віртуалізація

Контейнерна віртуалізація – віртуалізація на рівні операційної системи – дозволяє запускати ізольовані віртуальні системи на одному фізичному вузлі, але не дозволяє запускати операційні системи з ядрами, відмінними від типу ядра базової операційної системи. При такому підході не існує окремого шару гіпервізора, натомість сама хостова операційна система відповідає за поділ апаратних ресурсів між кількома гостьовими системами (контейнерами) та забезпечує їхню незалежність. Деякі реалізації – FreeBSD Jail (2000), Virtuozzo Containers (2000), Solaris Containers (2005), Linux-VServer,
OpenVZ (2005), LXC (2008),iCore Virtual Accounts (2008), Docker (2013).

Області застосування віртуалізації

Віртуальні машини

Віртуальна машина – це оточення, яке представляється для "гостьової" операційної системи, як апаратне. Однак насправді це програмне оточення, яке емулює програмне забезпечення хостової системи. Ця емуляція має бути достатньо надійною, щоб драйвери гостьової системи могли працювати стабільно. При використанні паравіртуалізації, віртуальна машина не емулює апаратне забезпечення, а замість цього пропонує використовувати спеціальний API.

Приклади застосування:

– Тестові лабораторії та навчання: тестуванню у віртуальних машинах зручно піддавати програми, що впливають на налаштування операційних систем, наприклад інсталяційні програми. За рахунок простоти в розгортанні віртуальних машин вони часто використовуються для навчання новим продуктам і технологіям.

– Поширення попередньо встановленого програмного забезпечення: багато розробників програмних продуктів створюють готові образи віртуальних машин із попередньо встановленими продуктами і надають їх на безкоштовній або комерційній основі. Такі послуги надають Vmware VMTN або Parallels PTN.

Віртуалізація ресурсів

Віртуалізація ресурсів (або поділ ресурсів, англ. *partitioning*) може бути представлена як поділ одного фізичного вузла на кілька частин, кожна з яких видно для власника як окремий сервер. Не є технологією віртуальних машин, що здійснюється на рівні ядра операційної системи.

У системах з гіпервізором другого типу обидві операційні системи (гостьова та гіпервізора) забирають фізичні ресурси та вимагають окремого ліцензування. Віртуальні сервери, що працюють на рівні ядра ОС, майже не втрачають швидкодії, що дає можливість запускати на одному фізичному сервері сотні віртуальних, що не вимагають додаткових ліцензій.

Дисковий простір або пропускний канал мережі розділені на кілька менших складових, і тому легше використовуваних ресурсів того ж типу.

Наприклад, до реалізації поділу ресурсів можна віднести OpenSolaris Network Virtualization and Resource Control (Проект Crossbow), що дозволяє створювати кілька віртуальних мережевих інтерфейсів на основі одного фізичного.

Агрегація, розподіл чи додавання безлічі ресурсів у великі ресурси чи об'єднання ресурсів. Наприклад, симетричні мультипроцесорні системи поєднують безліч процесорів; RAID та дискові менеджери поєднують безліч дисків в один великий логічний диск; RAID та мережеве обладнання використовує безліч каналів, об'єднаних так, щоб вони представлялися як єдиний широкосмуговий канал. На мета-рівні комп'ютерні кластери роблять усе перераховане вище. Іноді сюди ж відносять мережеві файлові системи абстраговані від сховищ даних, на яких вони побудовані, наприклад, Vmware VMFS, Solaris / OpenSolaris ZFS, NetApp WAFL.

Віртуалізація додатків

Віртуалізація програм – процес використання програми, перетвореної з потребуючої установки в операційну систему на не вимагає (потрібно тільки запустити). Для віртуалізації програм програмне забезпечення віртуалізатора визначає при установці віртуалізованого додатка, які потрібні компоненти ОС, і емулює їх. Таким чином, створюється необхідне спеціалізоване середовище для цього віртуалізованого додатка і, тим самим, забезпечується ізольованість роботи цього додатка. Для створення віртуальної програми віртуалізоване поміщається в контейнер, оформлений, як правило, у вигляді папки. При запуску віртуального додатка запускається віртуалізоване додаток і контейнер, що є для нього робочим середовищем. Робоче середовище запускається і надає локальні раніше створені ресурси, що включає ключі реєстру, файли та інші компоненти, необхідні для запуску і роботи програми. Таке віртуальне середовище працює як прошарок між програмою та операційною системою, що дозволяє уникнути конфліктів між програмами. Віртуалізацію додатків забезпечують, наприклад, програми Citrix XenApp, SoftGrid та VMware ThinApp.

Переваги:

– ізольованість виконання додатків: відсутність несумісностей та конфліктів;

– щоразу в первозданному вигляді: не захаращується реєстр, немає конфігураційних файлів – необхідно для сервера;

– менші ресурсозатрати проти емуляцією всієї операційної системи.

Віртуалізація серверів

Віртуалізація серверів – це процес поділу фізичного сервера на кілька унікальних та ізольованих віртуальних машин (серверів) за допомогою програмного забезпечення (гіпервізора). На кожному віртуальному сервері можуть виконуватися власні операційні системи незалежно.

Віртуалізація серверів дозволяє:

– Оптимізувати витрати на придбання серверного обладнання. Під кожну задачу виділяється віртуальний сервер із необхідною кількістю ресурсів (ЦПУ, ОЗП та ін.), простої обладнання мінімізуються.

– Спростити супровід інфраструктури. Створення, видалення або обслуговування віртуальної машини зазвичай простіше і швидше, ніж аналогічні операції з фізичним сервером.

– Підвищити стійкість до відмови інфраструктури. Віртуальні машини ізольовані одна від одної, програмний збій однією них не призведе до втрати працездатності сервісів і додатків інших.

Гіпервізор забезпечує ізольоване середовище виконання для кожної віртуальної машини, а також керує доступом ВМ та гостьових ОС до апаратних ресурсів фізичного сервера. Говорячи простими словами, гіпервізор забезпечує паралельне та незалежне функціонування кількох операційних систем на одному комп'ютері.

У класичному підході гіпервізори групуються за двома типами:

– гіпервізори першого типу запускаються безпосередньо на апаратному забезпеченні комп'ютера (залізі),

– гіпервізори другого типу, яким для роботи необхідна наявність хостової операційної системи.

– в останні кілька років класична класифікація зазнає змін – додався гібридний тип гіпервізорів (тип 1.5), який поєднує характеристики першого та другого типів.

Гіпервізори першого типу (native, bare-metal)

Гіпервізор першого типу виконується як контрольна програма безпосередньо на апаратній частині комп'ютера і вимагає ОС загального призначення. У цій архітектурі гіпервізор керує розподілом обчислювальних ресурсів і сам контролює всі звернення віртуальних машин до пристроїв.

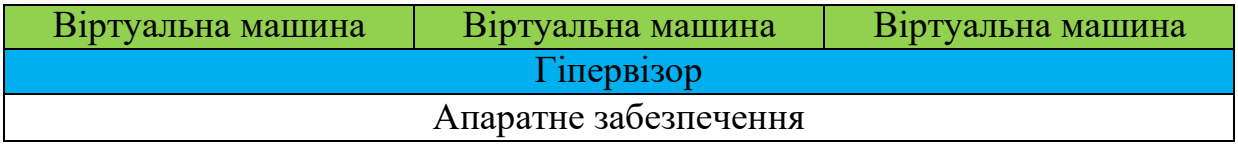

Рисунок 7.5 – Гіпервізор першого типу

Гіпервізори першого типу показують високу швидкодію, проте мають очевидний недолік – необхідність підтримувати драйвери пристроїв призводить до звуження списку сумісного апаратного забезпечення.

Приклади:

– VMWare ESXi;

– KVM (Proxmox VE) – може бути також віднесено до другого типу;

– Xen (Xenserver, Citrix Hypervisor), Hyper-V можуть бути також віднесені до гібридного типу.

Гіпервізори другого типу (hosted)

Гіпервізор другого типу виконується поверх хостової операційної системи (зазвичай Linux). Він управляє гостьовими операційними системами, тоді як емуляцією та управлінням фізичними ресурсами займається хостова ОС.

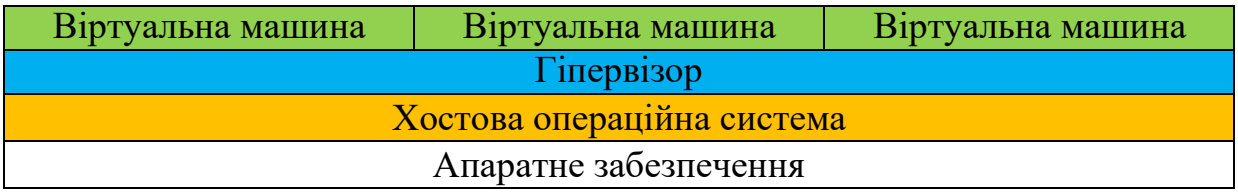

Рисунок 7.6 – Гіпервізор другого типу

Гіпервізори другого типу показують менше щодо гіпервізорів першого типу швидкодію і рідше використовуються в промисловій експлуатації, проте відмінно підходять для завдань навчання та розробки програмного забезпечення.

Приклади:

– Oracle VM VirtualBox, VMWare Workstation;
– KVM (Proxmox VE) – може бути віднесений до першого типу.

Гіпервізори гібридного типу (hybrid)

Гібридний гіпервізор поєднує характеристики гіпервізорів першого і другого типів – він виконується поверх спеціалізованої сервісної (або базової) операційної системи. Сервісна ОС називається батьківським розділом або доменом (parent partition у термінології Hyper-V або domain dom0 у термінології Xen). Після встановлення гіпервізора ядро ОС переходить у режим підтримки віртуалізації та передає керування ресурсами процесора та пам'яті гіпервізору. При цьому батьківський розділ бере на себе функцію обробки звернень до драйверів пристроїв та операцій вводу-виводу.

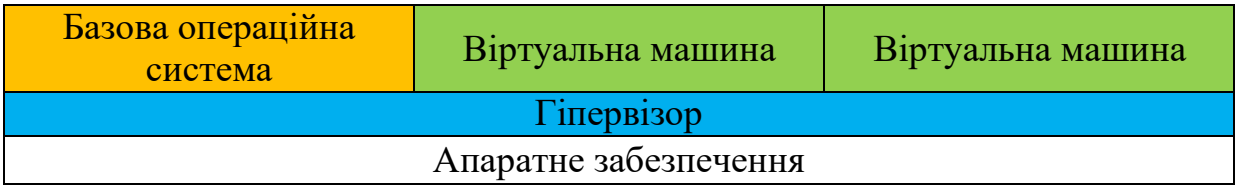

Рисунок 7.7 – Гібридний гіпервізор

Цей підхід є зручним з точки зору сумісності з обладнанням: не потрібно додавати в гіпервізор драйвери пристроїв (розширюється список сумісного апаратного забезпечення). Також гіпервізор звільняється від завдання обробки дзвінків до драйверів пристроїв – ці дзвінки обробляє сервісна ОС.

Приклади: Xen (XenServer, Citrix Hypervisor), Hyper-V – можуть бути віднесені до першого типу

7.4. Архітектура комп’ютерів

Архітектура комп'ютера – концептуальна модель комп'ютерної системи, втілена в її компонентах, їх взаємодії між собою і оточенням, що включає також принципи її проектування та розвитку. Аспекти реалізації (наприклад, технологія, що застосовується при реалізації пам'яті) не є частиною архітектури.

Рівні організації

Вирізняють кілька рівнів організації комп'ютера (комп'ютерної архітектури), від двох і більше:

– **Рівень 0.** Цифровий логічний рівень, це апаратне забезпечення машини, що складається з вентилів. також Логічні елементи (засувки), тригери, регістри.

– **Рівень 1.** Мікроархітектурний рівень, інтерпретація (мікропрограми) чи безпосереднє виконання. Електронні схеми виконують машинно-залежні програми. Сукупність регістрів процесора формує локальну пам'ять. також арифметико-логічний пристрій, пристрій керування. Його завдання – інтерпретація команд рівня 2 (рівня архітектури команд). В даний час на рівні архітектури команд зазвичай знаходяться прості команди, які виконуються за один цикл (такі, зокрема, RISC машини).

– **Рівень 2.** Рівень архітектури системи команд, трансляція (асемблер).

– **Рівень 3.** Рівень операційної системи, трансляція (асемблер). Це гібридний рівень: одна частина команд інтерпретується операційною системою, а інша мікропрограмою. також віртуальна пам'ять, файли.

– **Рівень 4.** Рівень мови асемблера, трансляція (компілятор). Четвертий рівень і вище використовується для написання прикладних програм, з першого до третього – системних програм. Програми у зручному для людини вигляді транслюються на мову рівнів 1-3.

– **Рівень 5.** Мова високого рівня. Програми мовами високого рівня транслюються зазвичай на рівні 3 та 4.

<table>
<tr><td colspan="2" rowspan="2"></td><td>Прикладні програми</td><td rowspan="5">Програмне забезпечення</td></tr>
<tr><td rowspan="4">Системне програмне забезпечення</td><td rowspan="3">Операційна система</td></tr>
<tr><td>Користувальницьке оточення</td></tr>
<tr><td>Ядро операційної системи (супервізор)</td></tr>
<tr><td></td><td>Вбудоване програмне забезпечення (firmware)</td></tr>
<tr><td colspan="2" rowspan="5"></td><td>Архітектура системи команд</td><td rowspan="4">Апаратна платформа (архітектура комп’ютера)</td></tr>
<tr><td>Мікропрограма (мікрокод)</td></tr>
<tr><td>Мікроархітектура</td></tr>
<tr><td>Цифровий логічний рівень</td></tr>
<tr><td>Фізичні пристрої</td></tr>
</table>

Рисунок 7.8 – Схема, що ілюструє багаторівневу структуру комп'ютера

Класифікація

За типом застосовуваного процесора

– CISC (англ. *complex instruction set computing)* – архітектура з повним набором команд. Такі процесори виконують усі команди, прості та складні, за велику кількість тактів. Команд у таких процесорах багато, і компілятори верхнього рівня рідко використовують усі команди.

– RISC (англ. *reduced instruction set computing*) – архітектура з скороченим набором команд. Такі процесори, загалом, працюють швидше, ніж із CISC- архитектурой, рахунок спрощення архітектури та скорочення кількості команд, але виконання складної команди вона складається з набору простих, що збільшує час виконання команди (за більшу кількість тактів). Варто відзначити, що сучасні процесори RISC по внутрішній складності наближаються, а то й перевершують класичні аналоги CISC.

– MISC (англ. *minimal instruction set computing*) – архітектура з мінімальним набором команд. Такі процесори мають мінімальну кількість команд, всі команди прості і вимагають невеликої кількості тактів на виконання, але якщо виконуються складні обчислення, наприклад, з числами з плаваючою комою, такі команди виконуються за істотно більшу кількість тактів, що перевищує CISC- і RISC-архітектури.

– VLIW (англ. *very long instruction word* – "дуже довга машинна команда") – архітектура з довгою машинною командою, в якій вказується паралельність виконання обчислень. Такі процесори отримали широке застосування у цифровій обробці сигналів.

За принципом поділу пам'яті

– Гарвардська архітектура – характерною рисою є поділ пам'яті програм та пам'яті даних.

– Фон Неймановська архітектура – характерною рисою є спільне зберігання програм та даних.