Зміст

1.Стек протоколів TCP/IP. Загальний опис

Стек протоколів TCP/IP, **TCP/IP-модель** — набір протоколів мережі Інтернет. Назва походить від назви стрижневих протоколів мережі Інтернет — **IP** (англ *Internet Protocol* — «міжмережевий протокол») і **TCP** (англ. *Transmission Control Protocol* — «протокол керування передаванням»). Фактично це систематизований стек протоколів, що поділяється на чотири рівні, які корелюються з еталонною моделлю OSI.

- TCP / IP - це набір протоколів, які дозволяють фізичним мережам об'єднуватися разом для утворення Internet. TCP / IP з'єднує індивідуальні мережі для утворення віртуальної обчислювальної мережі, в якій окремі головні комп'ютери ідентифікуються не фізичними адресами мереж, а IP-адресами.
- В TCP / IP використовується багаторівнева архітектура, яка чітко описує, за що відповідає кожен протокол. TCP і UDP забезпечують високо рівневі службові функції передачі даних для мережевих програм, і обидва спираються на IP при передачі пакетів даних. IP відповідає за маршрутизацію пакетів до їх пункту призначення.
- Дані, що переміщаються між двома прикладними програмами, що працюють на головних комп'ютерах Internet, "подорожують" вгору і вниз по стеку TCP / IP на цих комп'ютерах. Інформація, додана модулями TCP / IP на стороні відправника, "розрізається" відповідними TCP / IP-модулями на приймаючій кінці і використовується для відтворення вихідних даних.

Стек протоколів TCP / IP називають набір мережевих протоколів, що використовуються в Інтернет. У цьому стеку розрізняють кілька рівнів, і протоколи високого рівня завжди базуються на протоколах більш низьких рівнів.

1.У самому низу знаходяться фізичний рівень і канальний рівень. Приклад протоколу - Ethernet, що описує передачу даних по коаксіальному кабелю або кручений парі. Протоколи цих рівнів зазвичай реалізуються на рівні заліза, наприклад, в мережевої карти комп'ютера.

2.Вище йде мережевий рівень, де знаходиться протокол IP, що описує структуру мережі та доставку пакетів. Ще вище - транспортний рівень, де знаходиться протокол TCP, що використовується для передачі даних. Ці протоколи звичайно реалізуються на рівні Операційної Системи.

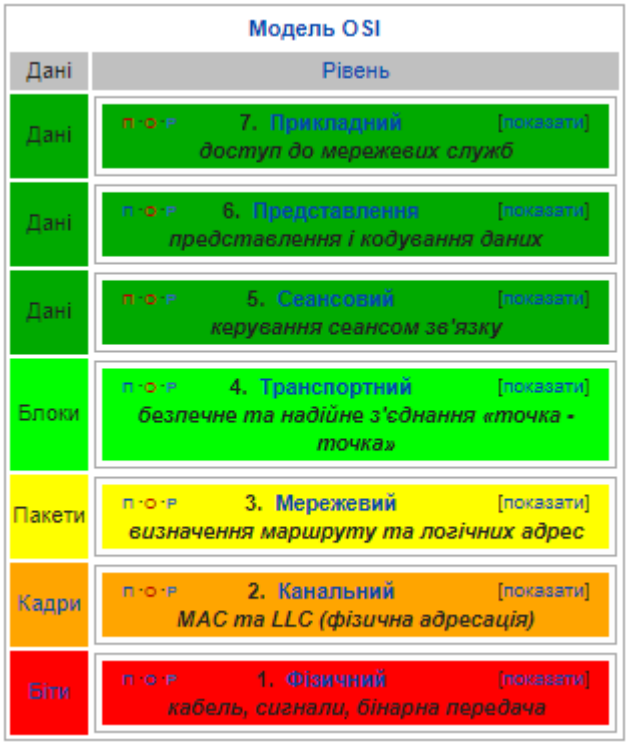
На самому верху знаходиться безліч протоколів прикладного рівня, що виконують конкретні прикладні завдання. Зазвичай вони програмуються в окремих додатках. IP - протокол, що лежить в основі Інтернету, його назва так і розшифровується: Internet Protocol. Згідно з протоколом, кожен вузол в мережі має свій IP адреса, що складається з 4х байт і зазвичай записується як nnnn Кожен вузол прямо «бачить» тільки вузли у своїй під мережі, з «схожими» адресами (докладніше див Маска під мережі). А іншим вузлам він передає пакети через проміжні вузли - Маршрутизатори. Подивитися, як виглядає маршрут пакета від вашого комп'ютера до інших вузлів, можна за допомогою команди traceroute.

CP протокол базується на IP для доставки пакетів, але додає дві важливі речі:

- з'єднання - це дозволяє йому, на відміну від IP, гарантувати доставку пакетів
- порти - для обміну пакетами між додатками, а не просто вузлами

2.Рівні моделі OSI.

Модель OSI — абстрактна мережева модель для комунікацій і розробки мережевих протоколів. Представляє рівневий підхід до мережі. Кожен рівень обслуговує свою частину процесу взаємодії. Завдяки такій структурі спільна робота мережевого обладнання й програмного забезпечення стає набагато простішою, прозорішою й зрозумілішою.



Рівні взаємодіють зверху вниз і знизу нагору за допомогою інтерфейсів і можуть ще взаємодіяти з таким же рівнем іншої системи за допомогою протоколів.

Прикладний рівень

Верхній (7-й) рівень моделі, забезпечує взаємодію мережі й користувача. Рівень дозволяє додаткам користувача доступ до мережних служб, таким як обробник запитів до баз даних, доступ до файлів, пересиланню електронної пошти. Також відповідає за передачу службової інформації, надає додаткам інформацію про помилки й формує запити до **рівня подання**.

Рівень представлення

Цей рівень відповідає за перетворення протоколів і кодування/декодування даних. Запити додатків, отримані з прикладного рівня, він перетворить у формат для передачі по мережі, а отримані з мережі дані перетворить у формат, зрозумілий додаткам. На цьому рівні може здійснюватися стиснення/розпакування або кодування/декодування даних, а також перенаправлення запитів іншому мережевому ресурсу, якщо вони не можуть бути оброблені локально.

Сеансовий рівень

Відповідає за підтримку сеансу зв'язку, дозволяючи додаткам взаємодіяти між собою тривалий час. Рівень керує створенням/завершенням сеансу, обміном інформацією, синхронізацією завдань, визначенням права на передачу даних і підтримкою сеансу в періоди неактивності додатків. Синхронізація передачі забезпечується розміщенням у потік даних контрольних точок, починаючи з яких відновлюється процес при порушенні взаємодії.

Транспортний рівень

Транспортний рівень (Transport layer) — 4-й рівень моделі OSI, призначений для доставлення даних без помилок, втрат і дублювання в тій послідовності, у якій вони були передані. При цьому немає значення, які дані передаються, звідки й куди, тобто він визначає сам механізм передачі. Блоки даних він розділяє на фрагменти, розмір яких залежить від протоколу, короткі об'єднує в один, довгі розбиває. Протоколи цього рівня призначені для взаємодії типу точка-точка.

Мережевий рівень

3-й рівень мережної моделі OSI, призначений для визначення шляху передачі даних. Відповідає за трансляцію логічних адрес й імен у фізичні, визначення найкоротших маршрутів, комутацію й маршрутизацію пакетів, відстеження неполадок і заторів у мережі. На цьому рівні працює такий мережний пристрій, як маршрутизатор.

Канальний рівень

Цей рівень призначений для забезпечення взаємодії мереж на фізичному рівні й контролю за помилками, які можуть виникнути. Отримані з фізичного рівня дані він упаковує в кадри даних[*джерело?*], перевіряє на цілісність, якщо потрібно — виправляє помилки й відправляє на мережний рівень. Канальний рівень може взаємодіяти з одним або декількома фізичними рівнями, контролюючи цю взаємодією й керуючи нею. Специфікація IEEE 802 поділяє цей рівень на 2 підрівня — MAC (Media Access Control) регулює доступ до поділюваного фізичного середовища, LLC (Logical Link Control) забезпечує обслуговування мережного рівня.

Фізичний рівень

Найнижчий рівень моделі, призначений безпосередньо для передачі потоку даних. Здійснює передачу електричних або оптичних сигналів у кабель і відповідно їхній прийом і перетворення в біти даних відповідно до методів кодування цифрових сигналів. Інакше кажучи, здійснює інтерфейс між мережним носієм і мережним пристроєм. На цьому рівні працюють концентратори й повторювачі (ретранслятори) сигналу. Фізичний рівень визначає електричні, процедурні і функціональні специфікації для середовища передачі даних, в тому числі роз'єми, розпаювання і призначення контактів, рівні напруги, синхронізацію зміни напруги, кодування сигналу.

3.IP протоколи. Фрагментація IP пакетів

Найбільш поширеними протоколами є протокол TCP/IP. Це протоколи нижнього рівня, власне є платформою зв'язку в Internet. TCP, або Transmission Control Protocol – розбиває передані дані на частини та нумерує їх. IP, або Internet Protocol передає всі частини одержувачу. Потім, за допомогою TCP, виконується перевірка, чи всі компоненти отримані. При отриманні всіх частин, протокол TCP розподіляє їх в необхідному порядку та монтує в єдине ціле.

Доцільно розрізняти фрагментацію повідомлень у вузлі-відправнику й динамічну фрагментацію повідомлень у 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, що розбиває потік байтів, переданий йому із прикладного рівня на повідомлення потрібного розміру (1460 байт для протоколу Ethernet). Тому протокол IP у вузлі-відправнику не використовує свої можливості по фрагментації пакетів.

А от при необхідності передати пакет у наступну мережу, для якої розмір пакета є занадто великим, IP-фрагментація стає необхідною. У функції рівня IP входить розбивка занадто довгого для конкретного типу складової мережі повідомлення на більше короткі пакети зі створенням відповідних службових полів, потрібних для наступного складання фрагментів у вихідне повідомлення.
IP-пакет може бути позначений як такий, що не фрагментується. Любою пакет, позначений таким чином, не може бути фрагментованим модулем IP ні при яких умовах. Якщо ж пакет, позначений як не фрагментований, не може досягти одержувача без фрагментації, то цей пакет просто знищується, а вузлу-відправникові посилає відповідне ICMP-повідомлення.
Протокол IP допускає можливість використання в межах окремої підмережі її власних засобів фрагментування, невидимих для протоколу IP. Наприклад, технологія ATM ділить поступаючи IP-пакети 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 виключенням, ніж правилом.
Процедури фрагментації й складання 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 повинне бути унікальним для даної пари відправник - одержувач, а також час, протягом якого пакет може бути активним у мережі.
Поле зсуву фрагмента повідомляє одержувачеві положення фрагмента у вихідному пакеті. Зсув фрагмента й довжина визначають частину вихідного пакета, принесену цим фрагментом.
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 зборки пакета.

4.Протоколи TCP І UDP – основні відмінності

Протоколи транспортного рівня, наступні в ієрархії за IP, використовуються для передачі даних між прикладними процесами, що реалізуються в мережевих вузлах. Пакет даних, що надійшов від одного комп'ютера іншому через Інтернет, має бути переданий процессу-оброблювачу, і саме по конкретному призначенню. Транспортний рівень приймає на себе відповідальність за це. На цьому рівні два основних протоколи – TCP і UDP.

TCP – транспортний протокол передачі даних в мережах TCP / IP, попередньо встановлює з'єднання з мережею.

UDP – транспортний протокол, що передає повідомлення-датаграми без необхідності установки з'єднання в IP-мережі.

Різниця між протоколами TCP і UDP – у так званій "гарантії доставки". TCP вимагає відгуку від клієнта, якому доставлений пакет даних, підтвердження доставки, і для цього йому необхідно встановлене заздалегідь з'єднання. Також протокол TCP вважається надійним, тоді як UDP отримав навіть іменування "протокол ненадійних датаграмм. TCP виключає втрати даних, дублювання і перемішування пакетів, затримки. UDP все це допускає, і з'єднання для роботи йому не потрібно. Процеси, яким дані передаються по UDP, повинні обходитися отриманим, навіть і з втратами. TCP контролює завантаженість з'єднання, UDP не контролює нічого, крім цілісності отриманих датаграмм.

З іншого боку, завдяки такій невибірковості і безконтрольності, UDP доставляє пакети даних (датаграми) набагато швидше, тому для додатків, які розраховані на широку пропускну здатність і швидкий обмін, UDP можна вважати оптимальним протоколом. До таких належать мережеві і браузерні ігри, а також програми перегляду потокового відео і додатки для відеозв'язку (або голосовий): від втрати пакета, повної або часткової, нічого не змінюється, повторювати запит не обов'язково, зате завантаження відбувається набагато швидше. Протокол TCP, як більш надійний, з успіхом застосовується навіть в поштових програмах, дозволяючи контролювати не тільки трафік, але і довжину повідомлення й швидкість обміну трафіком.

відмінність TCP від UDP полягає в наступному:
TCP гарантує доставку пакетів даних в незмінних вигляді, послідовності і без втрат, UDP нічого не гарантує.
TCP вимагає заздалегідь встановленого з'єднання, UDP з'єднання не вимагає.
UDP забезпечує більш високу швидкість передачі даних.
TCP надійніше і здійснює контроль над процесом обміну даними.
UDP переважніше для програм, що відтворюють потокове відео, відеофонії і телефонії, мережевих ігор.

5.Версії IP протоколу IP v4, IP v6

IPv4 (англ. ***Internet Protocol version 4***) — четверта версія мережевого протоколу IP. Перша версія протоколу, яка набула широко розповсюдження. Протокол IPv4, описаний у RFC 791 (вересень 1981 року), прийшов на заміну описаному у RFC 760 (січень 1980 року). Використовує 4 байтну форму запису адрес пристроїв в комп'ютерній мережі.

IPv6 (англ. *Internet Protocol version 6*) — нова версія IP-протоколу — IP версії 6. Розробка протоколу IPv6 почалася 1992 року, а з 2003 р. його підтримку забезпечують виробники більшості телекомунікаційного устаткування (корпоративного рівня). IPv6 — новий крок у розвитку Інтернету. Цей протокол розроблено з урахуванням вимог до Глобальної мережі, що постійно зростають. 3 лютого2011 року IANA виділила останні п'ять блоків IP-адрес /8 (IPv4).

Найбільш суттєва різниця між IPv4 та IPv6 полягає в тому, що раніше на інтернет-адресу виділяли 4 байти (32 біти), що відповідає стандартній на сьогодні чотириблоковій адресі IP, а протокол IPv6 виділяє на адресу 16 байтів (128 бітів). Це відповідає 340 секстильйонам адрес ($3{,}4x10^{38}$) або по $5x10^{28}$ адрес на кожну людину.

Протоколи ipv4 і ipv6 мають і інші відмінності. Наприклад, протокол IPv6 підтримує поліпшену многопоточную передачу, зате тут не підтримуються широкомовні пакети. IPv6 побудований на основі IPv4 з урахуванням всіх його помилок і недоробок. Але ці протоколи несумісні один з одним, тому всі пристрої повинні підтримувати ipv4 і ipv6, поки весь інтернет повністю не перейде на останній.

Протокол ipv4 старий і під час його створення не враховувалися багато аспектів його безпеки. Він передбачає, що про безпеку будуть піклується програми, які використовують мережу. Проте, IPv6 розроблений, щоб зробити передачу пакетів безпечнішою, тут з`явилися контрольні суми і шифрування пакетів. Протокол IPv6 призначений для забезпечення end-to-end шифрування для максимальної безпеки з`єднання. Розширення IPSec включає криптографічні протоколи для забезпечення захищеної передачі даних. Протоколи AH і ESP - це частина IPSec, які дозволяють перевірити цілісність і достовірність даних. ESP також забезпечує конфіденційність даних. Ще один протокол - IKE (Internet Key Exchange) який призначений для настройки і установки загальних атрибутів безпеки між двома пристроями.

6.Структура IP адреси для версії протоколу IP v4, IP v6

IP-пакет складається із заголовка й поля даних. Заголовок, що, як правило, має довжину 20 байт, має наступну структуру (мал. 5.12).

Мал. 5.12. Структура заголовка IP-пакета

Поле *Номер версії (Version)*, що займає 4 біти, вказує версію протоколу IP.

Поле *Довжина заголовка (IHL)* IP-пакета займає 4 біта і вказує значення довжини заголовка, вимірюване в 32-бітових словах. Звичайно заголовок має довжину в 20 байт (п'ять 32-бітових слів), але при збільшенні обсягу службової інформації ця довжина може бути збільшена за рахунок використання додаткових байт у полі *Опції (IP Options).*

Поле *Тип сервісу (Type of Service)* займає один байт і задає пріоритетність пакета й вид критерію вибору маршруту. Перші три біти цього поля утворять підполе *пріоритету* пакета *(Precedence)*.Пріоритет може мати значення від найнижчого — **0** (нормальний пакет) до найвищого — **7** (пакет керуючої інформації).
Поле *Тип сервісу* містить також три біти, що визначають критерій вибору маршруту. Реально вибір здійснюється між трьома альтернативами: малою затримкою, високою вірогідністю й високою пропускною здатністю.

Поле *Загальна довжина (Total Length)* займає 2 байти й означає загальну довжину пакета з урахуванням заголовка й поля даних. Максимальна довжина пакета обмежена розрядністю поля, що визначає цю величину, і становить 65 535 байт. У стандарті передбачається, що всі хости повинні бути готові приймати пакети аж до 576 байт

Поле *Ідентифікатор пакета (Identification)* займає 2 байти й використовується для розпізнавання пакетів, що утворилися шляхом фрагментації вихідного пакета. Всі фрагменти повинні мати однакове значення цього поля.

Поле *Прапори (Flags)* займає 3 біти й містить ознаки, пов'язані із 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. Біт, що залишився, зарезервований.

Поле *Зсув фрагмента (Fragment Offset)* займає 13 біт і задає зсув у байтах поля даних цього пакета від початку загального поля даних вихідного пакета, підданого фрагментації. Використовується при зборці/розбиранні фрагментів пакетів при передачах їх між мережами з різними величинами MTU. Зсув повинен бути кратним 8 байт.

Поле *Час життя (Time to Live)* займає один байт й означає граничний строк, протягом якого пакет може переміщатися по мережі. Час життя даного пакета виміряється в секундах і задається джерелом передачі.

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

Контрольна сума (Header Checksum) займає 2 байти й розраховується тільки по заголовку. Оскільки деякі поля заголовка міняють своє значення в процесі передачі пакета по мережі (наприклад, час життя), контрольна сума перевіряється й повторно розраховується при кожній обробці IP-заголовка. Контрольна сума – 16 біт - підраховується як доповнення до суми всіх 16-бітових слів заголовка.

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

Поле *Опції (IP Options)* є необов'язковим і використається звичайно тільки при налагодженні мережі. Механізм опцій надає функції керування, які необхідні або просто корисні при певних ситуаціях, однак він не потрібний при звичайних комунікаціях.

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

Нижче наведена роздруківка значень полів заголовка одного з реальних IP-пакетів, захоплених у мережі Ethernet засобами аналізатора протоколів Microsoft Network Monitor.

7. Структура пакету даних UDP протоколу.

UDP - мінімальний орієнтований на обробку повідомлень протокол транспортного рівня, задокументований у RFC 768.

UDP не надає жодних гарантій доставки повідомлення для вищого протоколу і не зберігає стану відправлених повідомлень. З цієї причини UDP іноді називають Unreliable Datagram Protocol (англ. - ненадійний протокол датаграм).

UDP забезпечує багатоканальну передачу (за допомогою номерів портів) і перевірку цілісності (за допомогою контрольних сум) заголовка і істотних даних. Надійна передача в разі необхідності повинна реалізовуватися призначеним для користувача додатком.

Біти	0 - 15	16 - 31
0-31	Порт відправника (Source port)	Порт отримувача (Destination port)
32-63	Довжина датаграми (Length)	Контрольна сума (Checksum)
64-...	Дані (Data)	

Заголовок UDP складається з чотирьох полів, кожне по 2 байта (16 біт). Два з них необов'язкові до використання в IPv4 (рожеві осередки в таблиці), в той час як в IPv6 необов'язковий тільки порт відправника.

Порт відправника. У цьому полі вказується номер порту відправника. Передбачається, що це значення задає порт, на який при необхідності буде надсилатися відповідь. В іншому ж випадку, значення повинно бути рівним 0. Якщо хостом-джерелом є клієнт, то номер порту буде, швидше за все, динамічним. Якщо джерелом є сервер, то його порт буде одним з «добре відомих».

Порт одержувача. Це поле є обов`язковим і містить порт одержувача. Аналогічно порту відправника, якщо хостом-одержувачем є клієнт, то номер порту динамічний, якщо одержувач - сервер, то це буде «добре відомий» порт.

Довжина датаграми. Поле, що задає довжину всієї датаграми (заголовка і даних) в байтах. Мінімальна довжина дорівнює довжині заголовка - 8 байт. Теоретично, максимальний розмір поля - 65535 байт для UDP-датаграми (8 байт на заголовок і 65527 на дані). Фактичний межа для довжини даних при використанні IPv4 - 65507 (крім 8 байт на UDP-заголовок потрібно ще 20 на IP-заголовок).

На практиці також слід враховувати, що якщо довжина IPv4 пакета з UDP буде перевищувати MTU (для Ethernet за замовчуванням 1500 байт), то відправка такого пакета може викликати його фрагментацію, що може привести до того, що він взагалі не зможе бути доставлений, якщо проміжні маршрутизатори або кінцевий хост не підтримуватимуть фрагментовані IP пакети. Також в RFC 791 вказується мінімальна довжина IP пакета 576 байт, яку повинні підтримувати всі учасники IPv4, і рекомендується відправляти IP пакети більшого розміру тільки в тому випадку якщо ви впевнені, що приймаюча сторона може прийняти пакети такого розміру. Отже, щоб уникнути фрагментації UDP пакетів (і можливої їх втрати), розмір даних в UDP не повинен перевищувати: MTU - (Max IP Header Size) - (UDP Header Size) = 1500 - 60 - 8 = тисячі чотириста тридцять-два байт. Для того щоб бути впевненим, що пакет буде прийнятий будь-яким хостом, розмір даних в UDP не повинен перевищувати: (мінімальна довжина IP пакета) - (Max IP Header Size) - (UDP Header Size) = 576 - 60 - 8 = 508 байт.

У Jumbogram'max IPv6 пакети UDP можуть мати більший розмір. Максимальне значення становить 4 294 967 295 байт (232 - 1), з яких 8 байт відповідають заголовку, а решта 4 294 967 287 байт - даними.

Слід зауважити, що більшість сучасних мережевих пристроїв відправляють і приймають пакети IPv4 довжиною до 10000 байт без їх поділу на окремі пакети. Неофіційно такі пакети називають «Jumbo-пакетами», хоча поняття Jumbo офіційно відноситься до IPv6. Проте, «Jumbo-пакети» підтримують не всі пристрої і перед організацією зв'язку за допомогою UDP / IP IPv4 посилок з довжиною, що перевищує 1500 байт, потрібно перевіряти можливість такого зв'язку дослідним шляхом на конкретному обладнанні.

Контрольна сума. Поле контрольної суми використовується для перевірки заголовка і даних на помилки. Якщо сума не згенерована передавачем, то поле заповнюється нулями. Поле не є обов'язковим для IPv4.

8. Presentation layer моделі OSI.

Представницький рівень, Рівень подання, Рівень представлення (англ. **Presentation layer**) — шостий рівень мережевої моделі OSI.

Цей рівень відповідає за перетворення протоколів і кодування / декодування даних. Запити додатків, отримані з рівня додатків, він перетворить у формат для передачі по мережі, а отримані з мережі дані перетворить у формат, зрозумілий додаткам. На цьому рівні може здійснюватися стиснення / розпакування або кодування / декодування даних, а також перенаправлення запитів іншому мережному ресурсу, якщо вони не можуть бути оброблені локально.

На представницькому рівні передана по мережі інформація не змінює змісту. За допомогою засобів, реалізованих на даному рівні, протоколи прикладних програм долають синтаксичні відмінності в експонованих даних або ж відмінності в кодах символів, наприклад погоджуючи представлення даних розширений двійковий код обміну інформацією **EBCDIC** використовуваного мейнфреймів компанії IBM з одного боку і американський стандартний код обміну інформацією **ASCII** з інший.

Іншою функцією, виконуваною на рівні уявлень, є шифрування і дешифрування даних, що забезпечує таємність переданих даних відразу для всіх прикладних служб. Щоб вирішити це завдання, процеси та коди, що знаходяться на рівні уявлень, повинні виконати перетворення даних. Прикладом протоколу, що забезпечує секретний обмін по мережі, є рівень захищених сокетів (англ. **Secure Sockets Layer** - SSL).

9. Session layer моделі OSI.

Сеа́нсовий рі́вень (англ. **Session layer**) - 5-й рівень мережевої моделі **OSI**, відповідає за підтримання сеансу зв'язку, дозволяючи програмам взаємодіяти між собою тривалий час. Рівень управляє створенням / завершенням сеансу, обміном інформацією, синхронізацією завдань, визначенням права на передачу даних і підтримкою сеансу в періоди неактивності програм. Синхронізація передачі забезпечується включенням у потік даних контрольних точок, починаючи з яких поновлюється процес при порушенні взаємодії.

Сеанси передачі складаються із запитів і відповідей, які здійснюються між програмами. Служби сеансового рівня зазвичай використовуються в середовищах програм, у яких потрібно використання віддаленого виклику процедур.

Прикладом протоколів сеансового рівня є протокол сеансового рівня стека протоколів OSI, який відомий як X.235 або ISO 8327. У разі втрати з'єднання цей протокол може спробувати його відновити. Якщо з'єднання не використовується тривалий час, то протокол сеансового рівня може його закрити і відкрити заново. Він дозволяє проводити передачу в **дуплексному** або в **напівдуплексному** режимі і забезпечує наявність контрольних точок в потоці обміну повідомленнями.

Іншими прикладами реалізації сеансового рівня є Zone Information Protocol (ZIP) - протокол **AppleTalk**, що забезпечує узгодженість процесу зв'язування по імені, а також протокол управління сеансом (англ. Session Control Protocol (SCP)) - протокол рівня сеансу IV стадії проекту розробки стека протоколів DECnet.

У рамках семантичних конструкцій сеансового рівня мережевої архітектури OSI цей рівень відповідає на службові запити з представницького рівня і здійснює службові запити до транспортному рівню.

Служби

- Аутентифікація
- Права доступу
- Відновлення сеансу (встановлення контрольних точок та відновлення)

Сеансовий рівень моделі OSI відповідає за встановлення контрольних точок та відновлення. Він дозволяє відповідним чином поєднувати і синхронізувати інформацію декількох потоків, можливо від різних джерел.

Прикладом застосування є організація відеоконференцій в мережі, коли звуковий і відео потоки повинні бути синхронізовані для уникнення проблем із синхронізацією руху губ з промовою. Управління правами на участь у розмові гарантує, що той, хто показується на екрані, дійсно є співрозмовником, який в даний момент говорить.

10. Transport layer моделі OSI

Модель OSI — абстрактна мережева модель для комунікацій і розробки мережевих протоколів. Представляє рівневий підхід до мережі. Кожен рівень обслуговує свою частину процесу взаємодії. Завдяки такій структурі спільна робота мережевого обладнання й програмного забезпечення стає набагато простішою, прозорішою й зрозумілішою.

Транспортний рівень (*Transport layer*) призначений для управління наскрізним транспортуванням повідомлень від вузла-відправника до вузла-одержувача з метою оптимізації використання засобів зв'язку, вибору виду і якості обслуговування процесу, а також забезпечення цілісності інформації, якщо її не забезпечують нижні рівні EMBBC. Інакше кажучи, на цьому рівні створюється віртуальний канал між двома точками мережі з вибором режиму передачі — комутація каналів, пакетів, повідомлень, а також формується стандартне транспортне повідомлення, що складається з інформації, яка передається, і сформованих ідентифікаторів початку та кінця повідомлення, що забезпечують передачу інформації від системи-відправника до системи-одержувача.

З метою вибору оптимального набору транспортних послуг стандартним протоколом визначено три типи (за частотою помилок і припустимої інтенсивності збоїв) мережних з'єднань і п'ять класів (за кількістю і якістю запитуваної послуги) транспортного протоколу, причому для підвищення надійності мережного з'єднання в цих класах протоколів передбачені п'ять груп спеціальних керуючих процедур, які називаються *примітивами*. Ці види сервісу відрізняються якістю послуг, що надаються: 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 та виправлення помилок передачі, таких як спотворення, втрата і дублювання пакетів.

Транспортний рівень відповідає за забезпечення доставки інформації з необхідною якістю між будь-якими вузлами мережі і реалізує такі функції:

- розбивка повідомлення сеансового рівня на пакети, їхня нумерація;
- буферизація прийнятих пакетів;
- впорядковування пакетів, що прибувають;
- адресація прикладних процесів;
- управління потоком.

Як правило, всі протоколи, починаючи з транспортного рівня і вище, реалізуються програмними засобами кінцевих вузлів мережі — компонентами їх мережних операційних систем. Найпоширенішими протоколами транспортного рівня є протоколи TCP і UDP (управління передачею) стека TCP/IP, NCP (Netware Core Protocol), SPX (упорядкований обмін пакетами) стека Novell, TP4 (протокол передачі класу 4) тощо.

11. Application layer моделі OSI

Модель OSI — абстрактна мережева модель для комунікацій і розробки мережевих протоколів. Представляє рівневий підхід до мережі. Кожен рівень обслуговує свою частину процесу взаємодії. Завдяки такій структурі спільна робота мережевого обладнання й програмного забезпечення стає набагато простішою, прозорішою й зрозумілішою.

Прикладний рівень (*Application layer*) є найвищим рівнем моделі, що безпосередньо пов'язаний із прикладними процесами і надає різні послуги, у тому числі й залежно від виду використовуваного обладнання.

Засобами прикладного рівня є набір різноманітних протоколів, за допомогою яких користувачі мережі одержують доступ до ресурсів, що розподіляються, таких як файли, принтери або гіпертекстові веб-сторінки, а також організують спільну роботу, наприклад за допомогою протоколу електронної пошти. Одиниця даних, якою оперує прикладний рівень, зазвичай називається повідомленням (*message*). Прикладний рівень надає набір усіх мережних сервісів, які забезпечує система кінцевому користувачеві, і відповідає за розв'язання таких завдань:

- ідентифікація, перевірка прав доступу;
- принт- і файл-сервіс, пошта, віддалений доступ.

На прикладному рівні працюють такі протоколи:

- *FTP* (*File Transfer Protocol*) — використовується для передачі файлів між комп'ютерами, на яких можуть бути встановлені різні операційні системи або платформи. Програмне забезпечення FTP-сервера виконується на комп'ютері, що передає файли, а клієнтська програма FTP використовується для організації з'єднання і завантаження файлів із сервера. Клієнтська програма FTP, що викликається з командного рядка, включена майже в усі реалізації пакета протоколів TCP/IP.
- *Telnet* використовується для емуляції терміналу і для надання доступу до аплікацій і файлів на іншому комп'ютері. На відміну від FTP, протокол Telnet не можна використовувати для копіювання файлів з одного комп'ютера на інший. Його можна використовувати тільки для читання й для виконання аплікацій на віддаленому вузлі.
- *SMTP* (*Simple Mail Transfer Protocol*) є простим протоколом ASCII, не орієнтованим на конкретного постачальника і використовується для передачі електронних повідомлень за допомогою Інтернету.
- *SNMP* (*Simple Network Management Protocol*) використовується для отримання інформації про мережу. Його можна застосовувати з різними платформами й операційними системами.

Окрім названих, існує досить багато інших протоколів прикладного рівня. Наприклад, такі протоколи, як NCP в операційній системі Novell NetWare, SMB у Microsoft Windows NT, NFS і TFTP, що входять у стек TCP/IP, HTTP (Hypertext Transfer Protocol) і NNTP (Network News Transfer Protocol).

12. Network layer моделі OSI

Модель OSI — абстрактна мережева модель для комунікацій і розробки мережевих протоколів. Представляє рівневий підхід до мережі. Кожен рівень обслуговує свою частину процесу взаємодії. Завдяки такій структурі спільна робота мережевого обладнання й програмного забезпечення стає набагато простішою, прозорішою й зрозумілішою.

Network layer — 3-й рівень моделі OSI, призначений для визначення шляху передачі даних. Відповідає за трансляцію логічних адрес й імен у фізичні, визначення найкоротших маршрутів, комутацію й маршрутизацію пакетів, відстеження неполадок і заторів у мережі. На цьому рівні працює такий мережний пристрій, як маршрутизатор.

Функції Мережного рівня

- Мережний рівень моделі OSI може бути як з установкою з'єднання, так і без нього. В стеці протоколів TCP/IP підтримує тільки протокол IP, який є протоколом без встановлення з'єднання; протоколи з установкою з'єднання знаходяться на наступних рівнях цієї моделі.
- Кожен хост в мережі повинен мати унікальну адресу, який визначає, де він знаходиться. Ця адреса зазвичай призначається з ієрархічної системи. В Інтернеті адреси відомі як адреси протоколу IP.
- Просування даних.

Так як багато мереж розділені на підмережі і з'єднуються з іншими розгалуженими мережами, мережі використовують спеціальні хости, які називаються шлюзами або роутерами (маршрутизаторами) для доставляння пакетів між мережами. Це також використовується в інтересах мобільних додатків, коли користувач рухається від однієї програми до іншої, в цьому випадку пакети (повідомлення) повинні слідувати за ним. У протоколі IPv4 така ідея описана, але практично не застосовується. IPv6 містить більш раціональне рішення.

13. Data link layer моделі OSI

Open System Interconnection (*OSI*) - набір протоколів, що описують процес взаємодії по мережі двох розподілених систем, незалежно від їх архітектури.

Модель OSI не є протоколом сам по собі, а описує структуру для побудови гнучкої, відмовостійкої і функціонально сумісної мережевої архітектури.

OSI складається з семи окремих, але пов'язаних один з одним рівнів, де кожен рівень має уявлення і може взаємодіяти тільки з двома прилеглими рівнями - «вище» і «нижче», і де кожен описує частину процесу отримання і передачі даних по мережі.

Канальний рівень (Data Link layer)

Цей рівень призначений для забезпечення взаємодії мереж на фізичному рівні й контролю за помилками, які можуть виникнути. Отримані з фізичного рівня дані він упаковує в кадри даних, перевіряє на цілісність, якщо потрібно виправляє помилки й відправляє на мережний рівень. Канальний рівень може взаємодіяти з одним або декількома фізичними рівнями, контролюючи й управляючи цією взаємодією. Специфікація IEEE 802 розділяє цей рівень на 2 підрівня — MAC (Media Access Control) регулює доступ до поділюваного фізичного середовища, LLC (Logical Link Control) забезпечує обслуговування мережного рівня. На цьому рівні працюють комутатори, мости й мережні адаптери.

MAC-підрівень забезпечує коректне спільне використання загального середовища, надаючи його в розпорядження тієї або іншої станції мережі. Також додає адресну інформацію до фрейму, позначає початок і кінець фрейму.

Рівень LLC відповідає за достовірну передачу кадрів даних між вузлами, а також реалізовує функції інтерфейсу з мережевим рівнем за допомогою фреймування кадрів. Також здійснює ідентифікування протоколу мережевого рівня.

У програмуванні цей рівень представляє драйвер мережної карти, в операційних системах є програмний інтерфейс взаємодії канального й мережного рівня між собою, це не новий рівень, а просто реалізація моделі для конкретної ОС. Приклади таких інтерфейсів: NDIS, ODI.

14. Phisical layer моделі OSI

Модель OSI - абстрактна мережева модель для комунікацій і розробки мережевих протоколів. Представляє рівневий підхід до мережі. Кожен рівень обслуговує свою частину процесу взаємодії. Завдяки такій структурі спільна робота мережевого обладнання й програмного забезпечення стає набагато простішою, прозорішою й зрозумілішою.

Фізичний рівень (Physical layer)

Найнижчий рівень моделі, призначений безпосередньо для передачі потоку даних. Здійснює передачу електричних або оптичних сигналів у кабель і відповідно їхній прийом і перетворення в біти даних відповідно до методів кодування цифрових сигналів. Інакше кажучи, здійснює інтерфейс між мережним носієм і мережним пристроєм. На цьому рівні працюють концентратори й повторювачі (ретранслятори) сигналу. Фізичний рівень визначає електричні, процедурні і функціональні специфікації для середовища передачі даних, в тому числі роз'єми, розпаювання і призначення контактів, рівні напруги, синхронізацію зміни напруги, кодування сигналу.

Цей рівень приймає кадр даних від канального рівня, кодує його в послідовність сигналів, які потім передаються у лінію зв'язку.

Фізичним рівнем в лінію зв'язку кадр даних (фрейм) не передається як єдине ціле. Кадр представляється як послідовність сигналів, що передаються один за одним. Сигнали, в свою чергу, представляють біти даних кадру.

В сучасних мережах використовуються 3 основних типи середовища передавання: мідний кабель (copper), оптичне волокно (fiber) та бездротове середовище передавання (wireless). Тип сигналу, за допомогою якого здійснюється передача даних, залежить від типу середовища передавання. Для мідного кабелю сигнали, що представляють біти даних є електричними імпульсами, для оптичного волокна — імпульсами світла. У випадку використання бездротових з'єднань сигнали є радіохвилями (електромагнітними хвилями).

Коли пристрій, що працює на фізичному рівні кодує біти кадру в сигнали для конкретного середовища передавання, він має розрізняти кадри. Тобто позначати, де закінчується один кадр і починається іншій. Інакше мережеві пристрої, що здійснюють прийом сигналів, не зможуть визначити, коли кадр буде отриманий повністю. Відомо, що початок і кінець кадру позначається на канальному рівні, але в багатьох технологіях фізичний рівень також може додати спеціальні сигнали, що використовуються тільки для позначення початку і кінця кадру даних.

Основними функціями фізичного рівня є: фізичні компоненти, кодування даних, передача даних. Фізичні компоненти — електронне обладнання, середовище передавання і конектори, через які передаються сигнали, що представляють біти даних.

15. Протоколи IPX/SPX. Загальний опис

Стек IPX/SPX є оригінальним стеком протоколів фірми Novell, розробленим для мережевої операційної системи NetWare ще на початку 80-х років. Протоколи мережевого і сеансового рівнів Internetwork Packet Exchange (IPX) і Sequenced Packet Exchange (SPX), які й дали назву стеку, є прямою адаптацією протоколів XNS фірми Xerox, поширених набагато менше, ніж стек IPX/SPX. Багато особливостей стека IPX/SPX обумовлені орієнтацією ранніх версій ОС NetWare (до версії 4.0) на роботу в локальних мережах невеликих розмірів, що складаються з персональних комп'ютерів із скромними ресурсами. Зрозуміло, що для таких комп'ютерів компанії Novell потрібні були протоколи, для реалізації яких потрібна була б мінімальна кількість оперативної пам'яті і які б швидко працювали на процесорах невеликої обчислювальної потужності. У результаті, протоколи стека IPX/SPX донедавна добре працювали в локальних мережах, і не дуже – у великих корпоративних мережах, тому що вони занадто перевантажували повільні глобальні зв'язки широкомовними пакетами, що інтенсивно використовуються декількома протоколами цього стека. З моменту випуску версії NetWare 4.0 Novell внесла і продовжує вносити у свої протоколи серйозні зміни, спрямовані на їхню адаптацію для роботи в корпоративних мережах. Зараз стек IPX/ SPX реалізований не тільки в NetWare, але й у декількох інших популярних мережевих ОС, наприклад SCO UNIX, Sun Solaris, Microsoft Windows NT.

На фізичному і канальном рівнях в мережах Novell використовуються всі популярні протоколи цих рівнів (Ethernet, Token Ring, FDDI та інші).

На мережевому рівні в стеку Novell працює протокол IPX, а також протоколи обміну маршрутною інформацією RIP і NLSP (аналог протоколу OSPF стека TCP / IP). IPX є протоколом, який займається питаннями адресації і маршрутизації пакетів в мережах Novell. Маршрутні рішення IPX засновані на адресних полях в заголовку його пакету, а також на інформації, що надходить від протоколів обміну маршрутною інформацієюПротокол IPX підтримує тільки дейтаграммний спосіб обміну повідомленнями, за рахунок чого економно споживає обчислювальні ресурси. Отже, протокол IPX забезпечує виконання трьох функцій: завдання адреси, встановлення маршруту та розсилку дейтаграмм.

Транспортному рівню моделі OSI в стеку Novell відповідає протокол SPX, який здійснює передачу повідомлень з встановленням сполук.

SPX (Sequence Packet eXchange) і його вдосконалена модифікація SPX II є транспортні протоколи 7-рівневої моделі ISO. Це протокол гарантує доставку пакета і використовує техніку ковзаючого вікна (віддалений аналог протоколу TCP). У разі втрати або помилки пакет пересилається повторно, число повторень задається програмно. У протоколі SPX не передбачена широкомовна або мультікастінг-адресація. У SPX індукується ситуація, коли партнер несподівано перериває з'єднання, наприклад через обрив зв'язку. Пакети SPX вкладаються в пакети IPX. При цьому у полі тип пакета IPX записується код 5. Заголовок пакета SPX завжди містить 42 байта, включаючи 30 байт заголовка IPX-пакета, куди він вкладений.

16. Команда Netstat. Опис та основні атрибути

netstat (англ. ***net**work **stat**istics*) — службова комп'ютерна програма, призначена для відображення поточного статусу підключень (вхідних та вихідних) по TCP/IP чи UDP, таблиць маршрутизації, кількості мережевих адаптерів та статистики протоколів. Програмне забезпечення доступне на UNIX-подібних та на системах, базованих на Windows.

Використовується для пошуку проблем в мережі та для визначення кількості трафіку як засобу виміру продуктивності.

Параметр	Платформа	Значення
-a	Усі доступні	Відображення всіх підключеннь і портів, на які комп'ютер очікує з'єднання. (Підключення з боку сервера звичайно не відображаються)
-b	Windows XP та новіші	Відображає назву програми, що створила з'єднання чи прослуховуючий порт
-b	OS X та NetBSD	Забезпечує відображення загальної кількості байт трафіку
-e	Усі доступні	Відображення статистики Ethernet. Параметр можна використовувати разом з -s
-h	Unix	Відображення всіх доступних ключів при роботі
-i	Unix	Відображає статистику мережевого інтерфейсу
-n	Усі доступні	Відображення адрес і номерів портів в числовому форматі, без спроб визначення імен
-p протокол	Windows та BSD	Відображення підключень для протоколу, заданому в параметрі. Доступні значення «tcp», «udp» та «ip». Використовується з ключем -s для відображення статистики
-r	Усі доступні	Відображення вмісту таблиці маршрутизації
-s	Усі доступні	Відображення детальної статистики по протоколах. За замовчуванням виводяться лише дані для TCP
-t	Linux	Відображає лише TCP підключення
-W	FreeBSD	Відображення широкого виводу — не обрізати назви хостів чи адреси IPv6
/?	Windows	Відображення всіх доступних параметрів при роботі

17. Основні можливості бібліотеки System.net.networkinformation

Простір імен System.Net.NetworkInformation надає доступ до даних мережного трафіку, інформації про мережеві адреси та повідомлення про зміни адреси для локального комп'ютера. Простір імен також містить класи, які реалізують утиліту Ping. Ви можете використовувати Ping і пов'язані з ним класи, щоб перевірити, чи доступний комп'ютер через мережу.

GatewayIPAddressInformation - Представляє IP-адресу мережевого шлюзу. Цей клас не може бути інстанційним.
GatewayIPAddressInformationCollection - Зберігає набір типів GatewayIPAddressInformation.
IcmpV4Statistics - Забезпечує протокол Internet Control Message Protocol для IPv4 (ICMPv4) статистичних даних для локального комп'ютера.
IcmpV6Statistics - Забезпечує Інтернет-протокол керування повідомленнями для Інтернет-протоколу версії 6 (ICMPv6) статистичних даних для локального комп'ютера.
IPAddressCollection - Зберігає набір типів IP-адреси.
IPAddressInformation - Надає інформацію про адресу мережевого інтерфейсу.
IPAddressInformationCollection - Зберігає набір типів IPAddressInformation.
IPGlobalProperties - Надає інформацію про мережеве підключення локального комп'ютера.
IPGlobalStatistics - Забезпечує статистичні дані Інтернет-протоколу (IP).
IPInterfaceProperties - Надає інформацію про мережеві інтерфейси, які підтримують Інтернет-протокол версії 4 (IPv4) або Інтернет-протокол версії 6 (IPv6).
IPInterfaceStatistics - Забезпечує статистичні дані Інтернет-протоколу для мережевого інтерфейсу на локальному комп'ютері.
IPv4InterfaceProperties - Надає інформацію про мережеві інтерфейси, які підтримують Інтернет-протокол версії 4 (IPv4).
IPv4InterfaceStatistics - Надає статистичні дані для мережного інтерфейсу на локальному комп'ютері.
IPv6InterfaceProperties - Надає інформацію про мережеві інтерфейси, які підтримують Інтернет-протокол версії 6 (IPv6).
MulticastIPAddressInformation - Надає інформацію про групову адресу мережевого інтерфейсу.
MulticastIPAddressInformationCollection - Зберігає набір типів MulticastIPAddressInformation.
NetworkAvailabilityEventArgs - Надає дані для події NetworkAvailabilityChanged.
NetworkChange - Дозволяє програмам отримувати сповіщення, коли змінюється адреса Інтернет-протоколу (IP) мережевого інтерфейсу, який також називається мережевою картою або адаптером.
NetworkInformationException - Виняток, який виникає при виникненні помилки під час отримання інформації про мережу.
NetworkInformationPermission - Контролює доступ до мережної інформації та статистики трафіку для локального комп'ютера. Цей клас не може бути успадкований.
NetworkInformationPermissionAttribute - Дозволяє діям безпеки для NetworkInformationPermission застосовуватися до коду з використанням декларативної безпеки.
NetworkInterface - Забезпечує конфігурацію та статистичну інформацію для мережевого інтерфейсу.
PhysicalAddress - Забезпечує MAC-адресу для мережного інтерфейсу (адаптера).
Ping - Дозволяє програмі визначати, чи доступний віддалений комп'ютер через мережу.
PingCompletedEventArgs - Надає дані для події PingCompleted.
PingException - Виняток, що виникає, коли метод Send або SendAsync викликає метод, який викидає виняток.
PingOptions - Використовується для керування передачею пакетів даних Ping.

PingReply - Надає інформацію про стан і дані, отримані в результаті надсилання або SendAsync.
TcpConnectionInformation - Надає інформацію про з'єднання протоколу керування передачею (TCP) на локальному комп'ютері.
TcpStatistics - Забезпечує статистичні дані протоколу керування передачею (TCP).
UdpStatistics - Забезпечує статистичними даними протокол UDP.
UnicastIPAddressInformation - Надає інформацію про одноадресний адресу мережевого інтерфейсу.
UnicastIPAddressInformationCollection - Зберігає набір типів UnicastIPAddressInformation.

18. Опис класу IP Global Properties

IPGlobalProperties Class - Надає інформацію про мережеве підключення локального комп'ютера.

Цей клас надає конфігурацію та статистичну інформацію про мережеві інтерфейси локального комп'ютера та мережні підключення.

Properties:
DhcpScopeName - Отримує ім'я області протоколу конфігурації динамічного хоста (DHCP).
DomainName - Отримує домен, у якому зареєстровано локальний комп'ютер.
HostName - Отримує ім'я хоста для локального комп'ютера.
IsWinsProxy - Отримує логічне значення, яке вказує, чи локальний комп'ютер діє як проксі-сервер служби імен Інтернету Windows (WINS).
NodeType - Отримує тип вузла базового вводу / виводу мережі (NetBIOS) локального комп'ютера.

Methods:
BeginGetUnicastAddresses (AsyncCallback, Object) - Починається асинхронний запит на отримання стабільної таблиці одноадресних IP-адрес на локальному комп'ютері.
EndGetUnicastAddresses (IAsyncResult) - Завершує очікуваний асинхронний запит на отримання стабільної таблиці IP-адрес на локальному комп'ютері.
GetActiveTcpConnections () - Повертає інформацію про підключення протоколу Інтернету 4 (IPv4) та протокол управління передачею IPv6 (TCP) на локальному комп'ютері.
GetActiveTcpListeners () - Повертає інформацію про кінцеву точку про Інтернет-протокол версії 4 (IPv4) і IPv6 протоколу керування передачею (TCP) слухачами на локальному комп'ютері.
GetActiveUdpListeners () - Повертає інформацію про слухачів протоколу Інтернет-протоколу 4 (IPv4) та протоколу протоколу протоколу IPv6 (UDP) на локальному комп'ютері.
GetIcmpV4Statistics () - Забезпечує статистичними даними для локального комп'ютера протокол ICMP.
GetIcmpV6Statistics () - Забезпечує статистичними даними для локального комп'ютера протокол ICMP.
GetIPGlobalProperties () - Отримує об'єкт, який надає інформацію про підключення до локальної мережі та статистику трафіку.
GetIPv4GlobalStatistics () - Забезпечує статистичні дані для локальних комп'ютерів версії 4 протоколу Internet (IPv4).
GetIPv6GlobalStatistics () - Забезпечує статистичні дані для локального комп'ютера версії 6 протоколу Internet (IPv6).
GetTcpIPv4Statistics () - Забезпечує статистичні дані протоколу керування передачею / Internet Protocol version 4 (TCP / IPv4) для локального комп'ютера.
GetTcpIPv6Statistics () - Забезпечує статистичні дані протоколу керування передачею / Internet Protocol version 6 (TCP / IPv6) для локального комп'ютера.
GetUdpIPv4Statistics () - Забезпечує статистичними даними для локального комп'ютера статистичні дані протоколу користувача / протоколу Інтернету 4 (UDP / IPv4).
GetUdpIPv6Statistics () - Забезпечує статистичними даними для локального комп'ютера статистику протоколу користувача / протоколу Інтернету версії 6 (UDP / IPv6).
GetUnicastAddresses () - Отримує стабільну таблицю адрес IP на локальному комп'ютері.

GetUnicastAddressesAsync () - Отримує стабільну таблицю одноадресних IP-адрес на локальному комп'ютері як асинхронну операцію.

19. Опис класу NetworkInformation

Простір імен System.Net.NetworkInformation забезпечує доступ до даних про трафік мережі, мережевих адрес і повідомленнями про зміну адрес локального комп'ютера. Це простір імен також містить класи, що реалізують функціональність програми Ping. Клас Ping і інші пов'язані з ним класи можуть використовуватися для перевірки доступності комп'ютера по мережі.

Основні класи

- GatewayIPAddressInformation - Являє IP-адреса шлюзу. Для цього класу неможливе створення екземплярів.
- IPAddressCollection - Зберігає набір типів IPAddress.
- IcmpV6Statistics - Надає статистику протоколу ICMPv6 для локального комп'ютера.
- IPAddressCollection - Зберігає набір типів IPAddress.
- NetworkChange - Дозволяє додаткам отримувати повідомлення при зміні IP-адреси мережевого інтерфейсу, званого також мережевою платою або адаптером.
- NetworkInformationException - Це виняток відбувається при помилку під час отримання відомостей про мережі.
- NetworkInformationPermission - Управляє доступом до відомостей про мережі і статистиці трафіку для локального комп'ютера. Цей клас не успадковується.

Делегати

- NetworkAddressChangedEventHandler - Посилається на один або кілька методів, що викликаються при зміні адреси мережевого інтерфейсу.
- NetworkAvailabilityChangedEventHandler - Посилається на один або кілька методів, що викликаються при зміні доступності мережі.
- PingCompletedEventHandler - Надає метод, що обробляє подію PingCompleted об'єкта Ping.

Перерахування

- DuplicateAddressDetectionState - Показує поточний стан IP-адреси.
- IPStatus - Повідомляє про стан відправки повідомлення перевірки зв'язку ICMP на комп'ютер.
- NetBiosNodeType - Вказує тип вузла NetBIOS.
- NetworkInformationAccess - Вказує дозвіл на доступ до відомостей про мережеві інтерфейси і статистиці трафіку.
- NetworkInterfaceComponent - Вказує версії протоколу IP, підтримувані мережевим інтерфейсом.

20. Опис класу ManagmentClass.

Являє клас управління CIM. Клас управління - це клас WMI, наприклад, такий як Win32_LogicalDisk, який може представляти дисковий накопичувач, або Win32_Process, який може представляти процес, наприклад Notepad.exe. Елементи цього класу дозволяють здійснювати доступ до даних WMI за допомогою певного шляху WMI.

Основні конструктори

- ManagementClass () - Ініціалізує новий екземпляр класу ManagementClass. Це конструктор за замовчуванням.

- ManagementClass (ManagementPath, ObjectGetOptions) - Виконує ініціалізацію нового екземпляра класу ManagementClass, ініціалізіруемих по шляху даного класу WMI з використанням заданих параметрів. Цей клас клас управління моделі CIM - один з таких класів WMI, як Win32_LogicalDisk, який може представляти дисковий накопичувач, або Win32_Process, який може представляти процес, наприклад Notepad.exe.

- ManagementClass (String) - Виконує ініціалізацію нового екземпляра класу ManagementClass, ініціалізіруемих по даному шляху. Цей клас клас управління моделі CIM - один з таких класів WMI, як Win32_LogicalDisk, який може представляти дисковий накопичувач, або Win32_Process, який може представляти процес, наприклад Notepad.exe.

- ManagementClass (String, String, ObjectGetOptions) - Виконує ініціалізацію нового екземпляра класу ManagementClass для заданого класу WMI в заданій області та з заданими параметрами. Цей клас клас управління моделі CIM - один з таких класів WMI, як Win32_LogicalDisk, який може представляти дисковий накопичувач, або Win32_Process, який може представляти процес, наприклад Notepad.exe

Основні властивості

- CanRaiseEvents - Повертає значення, яке показує, чи може компонент викликати подія. (Inherited from Component)
- ClassPath - Повертає або задає шлях до класу об'єкта. (Inherited from ManagementObject)
- Container - Повертає контейнер IContainer, що містить компонент Component. (Inherited from Component)
- Derivation - Отримує масив, що містить всі класи WMI в ієрархії успадкування, починаючи з цього класу до самого верхнього класу ієрархії.
- DesignMode - Повертає значення, яке вказує, чи знаходиться даний компонент Component в режимі конструктора в даний час. (Inherited from Component)
- Events - Повертає список обробників подій, які прикріплені до цього об'єкта Component. (Inherited from Component)

Основні методи

- Clone () - Повертає копію об'єкта.
- CompareTo (ManagementBaseObject, ComparisonSettings) - Порівнює даний об'єкт з іншим на підставі заданих параметрів. (Inherited from ManagementBaseObject)
- CopyTo (ManagementOperationObserver, ManagementPath) - Копіює об'єкт в інше місце розташування в асинхронному режимі. (Inherited from ManagementObject)
- CreateInstance () - Виконує ініціалізацію нового екземпляра класу WMI.
- Delete () - Видаляє об'єкт. (Inherited from ManagementObject)
- Dispose (Boolean) - Звільняє некеровані ресурси, використовувані об'єктом Component, а при необхідності звільняє також керовані ресурси. (Inherited from Component)
- Equals (Object) - Порівнює два керуючих об'єкта. (Inherited from ManagementBaseObject)
- Get () - Прив'язує відомості клас WMI до керуючого об'єкту. (Inherited from ManagementObject)

21. Опис класу NetworkInterface.

Надає конфігурацію і статистику мережевого інтерфейсу. Цей клас інкапсулює дані для мережевих інтерфейсів, також відомий як адаптери, на локальному комп'ютері. Вам не слід створювати екземпляри цього класу; GetAllNetworkInterfaces метод повертає масив, що містить один екземпляр цього класу для кожного мережевого інтерфейсу на локальному комп'ютері.

Конструктори

- NetworkInterface () - Ініціалізує новий екземпляр класу NetworkInterface.

Властивості

- Description - Повертає опис інтерфейсу.
- Id - Повертає ідентифікатор мережевого адаптера.
- IPv6LoopbackInterfaceIndex - Отримує індекс інтерфейсу замикання на себе IPv6.
- IsReceiveOnly - Повертає значення типу Boolean, яке вказує, чи налаштований мережевий інтерфейс тільки на прийом пакетів даних.
- LoopbackInterfaceIndex - Повертає індекс інтерфейсу замикання на себе IPv4.
- Name - Повертає ім'я мережевого адаптера.
- NetworkInterfaceType - Повертає тип інтерфейсу.
- OperationalStatus - Повертає поточний операційний стан мережевого підключення.
- Speed - Повертає швидкість мережевого інтерфейсу.
- SupportsMulticast - Повертає значення типу Boolean, яке вказує, чи дозволений мережному інтерфейсу прийом пакетів під LGPL.

Методи

- Equals (Object) - Визначає, чи рівний заданий об'єкт поточного об'єкту. (Inherited from Object)
- GetAllNetworkInterfaces ()- Повертає об'єкти, що описують мережеві інтерфейси локального комп'ютера.
- GetHashCode ()- Служить хеш-функцією за замовчуванням. (Inherited from Object)
- GetIPProperties ()- Повертає об'єкт, що описує конфігурацію мережевого інтерфейсу.
- GetIPStatistics ()- Отримує статистику IP для цього примірника NetworkInterface.
- GetIPv4Statistics ()- Отримує статистику IPv4 для цього примірника NetworkInterface.
- GetIsNetworkAvailable ()- Вказує, чи доступне підключення до мережі.
- GetPhysicalAddress ()- Повертає MAC-адреса (фізична адреса) даного адаптера.
- GetType ()- Повертає об'єкт Type для поточного екземпляра. (Inherited from Object)
- MemberwiseClone ()- Створює неповну копію поточного об'єкта Object. (Inherited from Object)
- Supports (NetworkInterfaceComponent) - Повертає значення типу Boolean, яке вказує, чи підтримує інтерфейс заданий протокол.
- ToString ()- Повертає рядок, що представляє поточний об'єкт. (Inherited from Object)

22. Опис класу IPGlobalStatistics.

```
public static void ShowIPStatistics(NetworkInterfaceComponent version)
{
   IPGlobalProperties properties = IPGlobalProperties.GetIPGlobalProperties();
   IPGlobalStatistics ipstat = null;
   switch (version)
   {
      case NetworkInterfaceComponent.IPv4:
          ipstat = properties.GetIPv4GlobalStatistics();
        Console.WriteLine("{0}IPv4 Statistics ",Environment.NewLine);
          break;
      case NetworkInterfaceComponent.IPv6:
          ipstat = properties.GetIPv4GlobalStatistics();
          Console.WriteLine("{0}IPv6 Statistics ",Environment.NewLine);
          break;
      default:
          throw new ArgumentException("version");
      //    break;
   }
   Console.WriteLine("  Forwarding enabled ..... : {0}",
      ipstat.ForwardingEnabled);
   Console.WriteLine("  Interfaces ..... : {0}",
      ipstat.NumberOfInterfaces);
   Console.WriteLine("  IP addresses ..... : {0}",
      ipstat.NumberOfIPAddresses);
   Console.WriteLine("  Routes ..... : {0}",
      ipstat.NumberOfRoutes);
   Console.WriteLine("  Default TTL ..... : {0}",
      ipstat.DefaultTtl);
   Console.WriteLine("");
   Console.WriteLine("  Inbound Packet Data:");
   Console.WriteLine("      Received ..... : {0}",
      ipstat.ReceivedPackets);
   Console.WriteLine("      Forwarded ..... : {0}",
      ipstat.ReceivedPacketsForwarded);
   Console.WriteLine("      Delivered ..... : {0}",
      ipstat.ReceivedPacketsDelivered);
   Console.WriteLine("      Discarded ..... : {0}",
      ipstat.ReceivedPacketsDiscarded);
   Console.WriteLine("      Header Errors ..... : {0}",
      ipstat.ReceivedPacketsWithHeadersErrors);
   Console.WriteLine("      Address Errors ..... : {0}",
      ipstat.ReceivedPacketsWithAddressErrors);
   Console.WriteLine("      Unknown Protocol Errors ..... : {0}",
      ipstat.ReceivedPacketsWithUnknownProtocol);
   Console.WriteLine("");
   Console.WriteLine("  Outbound Packet Data:");
   Console.WriteLine("      Requested ..... : {0}",
       ipstat.OutputPacketRequests);
   Console.WriteLine("      Discarded ..... : {0}",
      ipstat.OutputPacketsDiscarded);
   Console.WriteLine("      No Routing Discards ..... : {0}",
      ipstat.OutputPacketsWithNoRoute);
```

```
    Console.WriteLine("      Routing Entry Discards ..... : {0}",
        ipstat.OutputPacketRoutingDiscards);
    Console.WriteLine("");
    Console.WriteLine("  Reassembly Data:");
    Console.WriteLine("      Reassembly Timeout ..... : {0}",
        ipstat.PacketReassemblyTimeout);
    Console.WriteLine("      Reassemblies Required ..... : {0}",
        ipstat.PacketReassembliesRequired);
    Console.WriteLine("      Packets Reassembled ..... : {0}",
        ipstat.PacketsReassembled);
    Console.WriteLine("      Packets Fragmented ..... : {0}",
        ipstat.PacketsFragmented);
    Console.WriteLine("      Fragment Failures ..... : {0}",
        ipstat.PacketFragmentFailures);
    Console.WriteLine("");
}
```

Цей клас використовується GetIPv4GlobalStatistics і GetIPv6GlobalStatistics для збору даних про трафік методів для отримання IP-адреса. Протокол використовується для передачі IP-пакетів з вихідного комп'ютера на кінцевий комп'ютер. IP-адреса також обробляє пакет поділу, який не вміщується на кілька пакетів, які досить малі для транспорту, в рамках процесу під назвою фрагментація. Властивості цього класу відповідають об'єктам бази MIB для IP-адреси, визначеного в стандарті IETF RFC 2011 року.

23. Описати функцію, яка виводить інформацію про кількість мережевих адаптерів.

```
public static void DisplayDnsConfiguration()
{
   NetworkInterface[] adapters = NetworkInterface.GetAllNetworkInterfaces();
   foreach (NetworkInterface adapter in adapters)
   {
      IPInterfaceProperties properties = adapter.GetIPProperties();
      Console.WriteLine(adapter.Description);
      Console.WriteLine("  DNS suffix ..... : {0}",
         properties.DnsSuffix);
      Console.WriteLine("  DNS enabled ..... : {0}",
         properties.IsDnsEnabled);
      Console.WriteLine("  Dynamically configured DNS ..... : {0}",
         properties.IsDynamicDnsEnabled);
   }
   Console.WriteLine();
}
```

Масив NetworkInterface містить об'єкти, що описує доступні мережеві інтерфейси, або порожній масив, якщо інтерфейси не виявлені.

24. Описати функцію, яка визначає тип мережевих адаптерів встановлених на локальному ПК.

```
public static void ShowNetworkInterfaces()
{
   IPGlobalProperties computerProperties = IPGlobalProperties.GetIPGlobalProperties();
   NetworkInterface[] nics = NetworkInterface.GetAllNetworkInterfaces();
   Console.WriteLine("Interface information for {0}.{1}     ",
         computerProperties.HostName, computerProperties.DomainName);
   if (nics == null || nics.Length < 1)
   {
      Console.WriteLine("  No network interfaces found.");
      return;
   }

   Console.WriteLine("  Number of interfaces ..... : {0}", nics.Length);
   foreach (NetworkInterface adapter in nics)
   {
      IPInterfaceProperties properties = adapter.GetIPProperties();
      Console.WriteLine();
      Console.WriteLine(adapter.Description);
      Console.WriteLine(String.Empty.PadLeft(adapter.Description.Length,'='));
      Console.WriteLine("  Interface type ..... : {0}", adapter.NetworkInterfaceType);
      Console.WriteLine("  Physical Address ..... : {0}",
               adapter.GetPhysicalAddress().ToString());
      Console.WriteLine("  Operational status ..... : {0}",
         adapter.OperationalStatus);
      string versions ="";

      // Create a display string for the supported IP versions.
      if (adapter.Supports(NetworkInterfaceComponent.IPv4))
      {
          versions = "IPv4";
       }
      if (adapter.Supports(NetworkInterfaceComponent.IPv6))
      {
         if (versions.Length > 0)
         {
            versions += " ";
          }
         versions += "IPv6";
      }
      Console.WriteLine("  IP version ..... : {0}", versions);
      ShowIPAddresses(properties);

      // The following information is not useful for loopback adapters.
      if (adapter.NetworkInterfaceType == NetworkInterfaceType.Loopback)
      {
         continue;
      }
      Console.WriteLine("  DNS suffix ..... : {0}",
         properties.DnsSuffix);
```

```
        string label;
        if (adapter.Supports(NetworkInterfaceComponent.IPv4))
        {
            IPv4InterfaceProperties ipv4 = properties.GetIPv4Properties();
            Console.WriteLine("  MTU.................................. : {0}", ipv4.Mtu);
            if (ipv4.UsesWins)
            {

                IPAddressCollection winsServers = properties.WinsServersAddresses;
                if (winsServers.Count > 0)
                {
                    label = "  WINS Servers ........................... :";
                    ShowIPAddresses(label, winsServers);
                }
            }
        }

        Console.WriteLine("  DNS enabled ............................. : {0}",
            properties.IsDnsEnabled);
        Console.WriteLine("  Dynamically configured DNS .............. : {0}",
            properties.IsDynamicDnsEnabled);
        Console.WriteLine("  Receive Only ............................ : {0}",
            adapter.IsReceiveOnly);
        Console.WriteLine("  Multicast ............................... : {0}",
            adapter.SupportsMulticast);
        ShowInterfaceStatistics(adapter);

        Console.WriteLine();
    }
```

Цей клас інкапсулює дані для мережевих інтерфейсів, також відомий як адаптери, на локальному комп'ютері. Нам не слід створювати екземпляри цього класу; GetAllNetworkInterfaces метод повертає масив, що містить один екземпляр цього класу для кожного мережевого інтерфейсу на локальному комп'ютері.

25. Описати функцію, яка виводить інформацію про Mac адресу даного ПК.

MAC-адреса (від англ. *Media Access Control* — управління доступом до носія) — це унікальний ідентифікатор, що зіставляється з різними типами устаткування для комп'ютерних мереж. Більшість мережевих протоколів канального рівня використовують один з трьох просторів MAC-адрес, керованих IEEE: MAC-48, EUI-48 і EUI-64. Адреси в кожному з просторів теоретично мають бути глобально унікальними. Не всі протоколи використовують MAC-адреси, і не всі протоколи, що використовують MAC-адреси, потребують подібної унікальності цих адрес.

```
public static string GetMacAddress()
{
   string macAddresses = "";
   foreach (NetworkInterface nic in NetworkInterface.GetAllNetworkInterfaces())
   {
      if (nic.OperationalStatus == OperationalStatus.Up)
      {
         macAddresses += nic.GetPhysicalAddress().ToString();
         break;
      }
   }
   return macAddresses;
}
```

26. Описати функцію, яка визначає IP адресу даного ПК.

IP-адреса, **адреса Ай-Пі** (від англ. *Internet Protocol address*) — це ідентифікатор (унікальний числовий номер) мережевого рівня, який використовується для адресації комп'ютерів чи пристроїв у мережах, які побудовані з використанням протоколу TCP/IP (н-д Інтернет). У мережі Інтернет потрібна глобальна унікальність адрес, у разі роботи в локальній мережі — у межах мережі.

У версії протоколу IPv4 IP-адреса має довжину 4 байта, а у версії IPv6 — 16 байт.

Прикладом IP-адреси може бути адреса 127.0.0.1 (локальна IP-адреса, змінити її неможливо, і вона на кожній ОС лише одна — localhost).

Процес перетворення доменного імені на адресу IP виконується DNS-сервером.

```
public static string GetIpAddress()
{
   String host = System.Net.Dns.GetHostName();
   System.Net.IPAddress ip = System.Net.Dns.GetHostByName(host).AddressList[0];

      return ip.ToString();
}
```

27. Описати функцію, яка дозволяє отримати інформацію про зайняті TCP/IP порти на даному ПК.

TCP / IP - це набір протоколів, який задає стандарти зв'язку між комп'ютерами і містить докладні угоди про маршрутизації і межсетевом взаємодії. TCP / IP широко застосовується в Internet, тому з його допомогою можуть спілкуватися користувачі з дослідних інститутів, шкіл, університетів, урядових установ і промислових підприємств.

TCP / IP забезпечує зв'язок підключених до мережі комп'ютерів, зазвичай званих хостами. Будь-яку мережу можна підключити до іншої мережі і організувати зв'язок з її хостами. Незважаючи на те, що існують різні мережеві технології, багато з яких засновані на комутації пакетів і потоковому режимі передачі, набір протокол TCP / IP має однією істотною перевагою: він забезпечує апаратну незалежність.

```
private void WriteToGridTcpInfo()
{
   List<Connection> ConnectionTCP = new List<Connection>();
   ConnectionTCP.AddRange(NetworkInformation.GetTcpV4Connections());
   for (int index = 0; index < ConnectionTCP.Count; index++)
   {
      Connection info = ConnectionTCP[index];
      if (info.ConnectType == ConnectType.Tcp)
      {
         dataGridView1.Rows.Add(info.OwningProcess, info.ConnectType,
                                info.LocalEndPoint, info.RemoteEndPoint, info.State );
      }
   }

    }
```

28. Отримати назву процесу по його ID ідентифікатору.

.NET Core — це модульна платформа для розробки програмного забезпечення, з відкритим вихідним кодом. Сумісна з такими операційними системами як Windows, Linux та macOS. Була випущена компанією Microsoft. Підтримує наступні мови програмування: C#, Visual Basic .NET (частково) та F#.

За допомогою засобів платформи .NET Core та середовища розробки Visual Studio 2017 Community можна створити просту консольну програму, що визначатиме ім'я запущеного процесу в системі по його ідентифікатору і це працюватиме на різних операційних системах.

```
1   using System;
2   using System.Diagnostics;
3
4   namespace ProcessNameById
5   {
        0 references
6       public class Program
7       {
            0 references
8           public static void Main(string[] args)
9           {
10              int processId;
11
12              Console.Write("Type process ID: ");
13
14              while (!int.TryParse(Console.ReadLine(), out processId))
15              {
16                  Console.WriteLine("Error: Invalid integer value.");
17                  Console.Write("Type process ID: ");
18              }
19
20              Console.WriteLine("[{0}]Process name: {1}", processId, GetProcessName(processId));
21          }
22
            1 reference
23          public static string GetProcessName(int processId)
24          {
25              return Process.GetProcessById(processId).ProcessName;
26          }
27      }
28  }
```