ЛЕКЦІЯ №2 ФАЙЛОВІ СИСТЕМИ

Таблиця розміщення файлів (FAT).

Весь логічний диск розбивається операційною системою на частини однакового розміру, які називаються кластерами.

Кластер може утримувати декілька секторів. Для кожного кластеру FAT має свій індивідуальний елемент, в якому зберігається інформація про використання даного кластеру.

Розмір FAT визначається загальною кількістю кластерів на логічному диску. Якщо кількість кластерів на диску < 4085, то використовується FAT12, якщо 4085<кластерів<655525-FAT16, далі FAT32.

Назви типів FAT походять від розміру елементу: для FAT12 кожний елемент має розмір 12 бітів (1,5 байта), FAT16 – 16 бітів (2б), FAT32 – 32біти (4б). У FAT32 чотири старших двійкових розряди зарезервовані та ігноруються в процесі роботи операційної системи. Значущими є тільки сім молодших шістнадцяткових розрядів елементу. Кожному файлу, який міститься в області даних диску, відповідає ланцюжок елементів FAT – впорядкований однонаправлений список. Він забезпечує пересування тільки вперед. Якщо потрібно повернутися до попереднього кластеру, то треба знов провести пошук з самого початку списку.

Перші 2 елементи FAT – резервні. В першому резервному елементі FAT зберігається сигнатура. Для FAT12 це FF8h, FAT16 – FFF8h, FAT32 – FFFFFF8h.

У другому резервному елементі при форматуванні диску записується ознака кінця файлу (FFFh, FFFFh, FFFFFFFh для Microsoft). Інші фірми можуть використовувати інші значення. Крім того, системи FAT16 та FAT32 можуть використовувати 2 старших розряди цього елементу в якості прапорів. Прапор ClnShutBitMask займає в системі FAT16 двійковий розряд 15, в FAT32 – розряд 27. Якщо він дорівнює одиниці, то логічний диск чистий (clear), якщо нулю, то “брудний”(dirty).

Термін “брудний” означає, що робота з диском не була закінчена правильно і при завантаженні системи повинна бути виконана процедура відновлення диску. Інший прапор HrdErrBitMask – служить ознакою присутності збоїв при виконанні

операції введення/виведення. В системі FAT16 він міститься в двійковому розряді 14, в FAT32 – в розряді 26.

При завантаженні ОС він встановлюється в 1, а якщо виник збій при запису або читанні інформації, то він скидається в 0.

Табл. 1.8 Значення елементів таблиці розміщення файлів FAT

FAT 12	FAT 16	FAT 32	Зміст
0	0	0	Вільний кластер
FF7 h	FFF7 h	FFFFFF7 h	Дефектний кластер
003	0003	0000003	№ наступного кластеру диску

Загальна схема використання FAT.

1. Читаємо в пам’ять FAT повністю.

Зазвичай FAT розташовується відразу після BOOT-сектору (логічний сектор з №1). Для точного визначення початкового сектору FAT потрібно прочитати в пам’ять BOOT-сектор та проаналізувати утримуваний BPB. В ньому є поля, де записано кількість зарезервованих секторів, котрі розташовуються перед FAT та розмір FAT в секторах. Крім того, на диску може знаходитися декілька копій FAT. ОС використовує лише першу копію, інші потрібні лише для роботи утиліт відновлення диску.

2. Визначаємо номер 1-го кластеру файлу, для якого необхідно визначити його розташування на диску.

3. Використовуємо номер першого кластеру як індекс в FAT для визначення номера наступного кластеру.

4. Повторюємо попередню процедуру доти, доки вилучене з FAT значення не буде відповідати кінцю файлу.

Процедура визначення номера кластеру з FAT залежить від формату таблиці розміщення файлів.

16-бітову FAT можемо представити як масив 16-бітових чисел. Для визначення номера наступного кластеру потрібно вилучити 16-бітове значення з FAT, використовуючи в якості індексу номер попереднього кластеру. Для 12-бітової FAT процедура визначення наступного кластеру виглядає так:

1. Помножити номер початкового кластеру на 3.

2. Поділити результат на 2, тому що кожен елемент таблиці має довжину 1,5 байт.

3. Прочитати 16-бітове слово з FAT, використовуючи в якості зсуву значення, отримане після ділення на 2.

4. Якщо номер початкового кластеру парний, то на вибране з FAT слово треба накласти маску 0fffh, залишивши молодші 12 бітів. Якщо номер початкового кластера непарний, то вибране з FAT значення потрібно зсунути праворуч на 4 біти, залишивши 12 старших бітів.

5. Отриманий результат є номером наступного кластеру в ланцюжку.

Каталоги файлів.

Каталог файлів – це масив 32 байтних елементів описувачів файлів. З точки зору ОС всі каталоги – це файли і можуть містити довільну кількість записів. Виняток – кореневий каталог в системах FAT12 та FAT16. Кореневий каталог (ROOT DIRECTORY) – це головний каталог диску, з якого починається дерево підкаталогів.

Для кореневого каталогу в FAT12 та FAT16 виділено в системній області логічного диску спеціальне місце фіксованого розміру 18 Кб, яке розраховане на зберігання 512 елементів. В FAT32 кореневий каталог – довільного розміру.

Табл. 1.9 Структура кореневого каталогу

Зсув	Розмір	Опис
0	8	Коротке ім’я файлу чи каталогу, вирівняне ліворуч і доповнене проміжками.
8	3	Розширення імені файлу, вирівняне ліворуч і доповнене проміжками
0Bh	1	Атрибути файлу
1Ch	1	Зарезервовано для Windows NT (=0)
1Dh	1	Уточнення часу створення файлу (містить десятки мілісекунд 0..199)
0Eh	2	Час створення файлу чи час його останньої модифікації.
10h	2	Дата створення файлу чи час його останньої модифікації.
12h	2	Дата останнього звертання до файлу для запису або читання

Зсув	Розмір	Опис
14h	2	Старше слово номеру першого кластеру, розподіленого файлу.
16h	2	Час виконання останньої операції запису в файл.
18h	2	Дата виконання останньої операції запису в файл.
1Ah	2	Молодше слово номеру першого кластеру файлу.
1Ch	4	Розмір файлу в байтах.

Байт атрибутів є приналежністю кожного файлу. Поля від 0Dh до 14h обробляються тільки в FAT32. Біти цього байту мають наступне значення:

- 00h файл призначений лише для читання;

- 01h прихований файл (його немає в списку);

- 02h системний файл. Цей біт зазвичай встановлений в файлах, які виступають частиною ОС;

- 07h позначка диску. Для цього значення поля імені файлу та розширення повинні розглядатися як одне поле довжиною 11 байт. Це поле й містить позначку диску;

- 14h підкаталог даного каталогу;

- 05h архівний файл.

Формат поля часу створення файлу.

Біти 04…00 - число двосекундних інтервалів в двійковій формі (від 0 до 29, тобто від 0 до 58 сек.)

Біти 10…05 - число хвилин в двійковій формі (0-59).

1. Перший байт короткого імені виконує функції ознаки зайнятості елементу каталогу. Якщо він дорівнює E5h, то елемент каталогу вільний; 00h – то елемент каталогу вільний і є початком чистої області каталогу; 05h, то в цьому байті міститься ASCII символ з кодом 0E5h. В короткому імені не можна використовувати символи з кодами <20h (виняток - 05h).

2. Не можна використовувати символи з кодами 22h, 2Ah, 2Bh, 2Ch, 2Eh, 2Fh, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh ,5Bh, 5Ch, 5Dh, 7Ch.

3. Не можна використовувати символ проміжку (20h) в першому байті.

Ознака того, що вільний елемент каталогу використовується для зберігання ділянки довгого імені файлу - заповнення одиницями розрядів з 0 по 3 байту атрибутів.

Довге ім’я можна створювати починаючи з ОС Windows 95. Для його зберігання використовують вільні елементи каталогу, суміжні з основним елементом - описувачем файлу.

Воно записується в Unicode, де кожній національній абетці відповідає свій набір кодів. До вільних елементів каталогу довге ім’я записується в розрізаному на шматочки вигляді. В одному елементі каталогу можна зберігати фрагмент до 13 символів Unicode. Невикористана частина останнього фрагменту заповнюється кодами FFFFh.

Табл. 1.10 Структура елементу каталогу з довгим ім’ям файлу

Зсув	Розмір	Опис
0h	1	Номер фрагменту
01h	10	Перша частина фрагменту імені файлу
0Bh	1	Атрибути файлу
0Ch	1	Байт прапорів
0Dh	1	Контрольна сума короткого імені
0Eh	12	Друга частина фрагменту імені
1Ah	2	Номер першого кластеру (повинен бути 0)
1Ch	4	Третя частина фрагменту імені

Довге ім’я записується до каталогу першим. Фрагменти розташовані в зворотньому порядку. Далі міститься описувач файлу, який зберігає скорочений варіант цього імені.

Початок каталогу
Остання частина довгого імені
……
Перша частина довгого імені
Описувач файлу
……   Кінець каталогу

Рис. 1.3 Описувач файлу

Файлова система NTFS.

Для NTFS вся інформація на томі є файлом або частиною файлу. Кожний розподілений на томі NTFS сектор належить деякому файлу.

Диск NTFS умовно ділиться на дві частини. Розглянемо загальну структуру NTFS. Перші 12% диска відводяться під так звану MFT зону - простір, в який росте метафайл MFT.

Запис яких-небудь даних в цю область неможливий. MFT-зона завжди тримається порожньою - це робиться для того, щоб найголовніший, службовий файл (MFT) не фрагментувався при своєму зростанні. Інші 88% диска є звичайним простором для зберігання файлів.

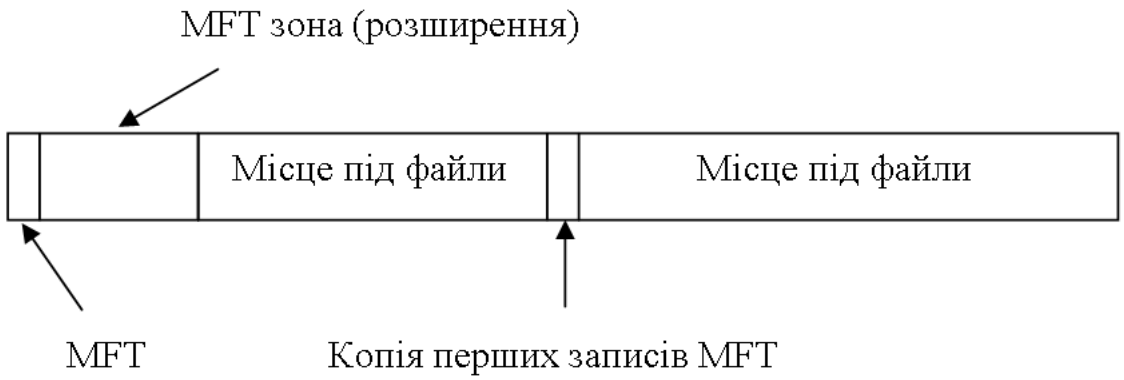

Рис. 1.4 Загальна структура NTFS

Вільне місце диска, включає все фізично вільне місце - незаповнені проміжки MFT-зони туди теж включаються. Механізм використовування MFT-зони такий: коли файли вже не можна записувати в звичайний простір, MFT-зона просто скорочується (в поточних версіях операційних систем рівно в два рази), звільняючи таким чином місце для запису файлів.

При звільненні місця в звичайній області MFT зона може знову розширитися. При цьому не виключена ситуація, коли в цій зоні залишилися і звичайні файли: ніякої аномалії тут немає. Метафайл MFT все-таки може фрагментуватися, хоч це і було б небажано.

Ця заснована на атрибутах файлова система підтримує об'єктно-орієнтовані додатки, обробляючи всі файли як об'єкти, які мають атрибути, що визначаються користувачем і системою.

Функціональний склад файлової системи NTFS.

Головна файлова таблиця. Кожний файл на томі NTFS представлений записом в спеціальному файлі, який називається головною файловою таблицею (MFA — master file table). NTFS резервує перші 16 записів таблиці для спеціальної інформації. Перший запис цієї таблиці описує безпосередньо головну файлову таблицю; за нею слідує дзеркальний запис (mirror record) MFT. Якщо перший запис MFT зруйнований, то NTFS читає другий запис для відшукання дзеркального файлу MFT, перший запис якого ідентичний першому запису MFT.

Місцеположення сегментів даних MFT і дзеркального файлу MFT записані в секторі початкового завантаження. Дублікат сектора початкового завантаження знаходиться в логічному центрі диска.

Третій запис MFT — файл реєстрації (log file); використовується для відновлення файлів. Сімнадцята і подальші записи головної файлової таблиці використовуються власне файлами і каталогами (також розглядаються як файли NTFS) на томі. На рисунку 1.5 показана спрощена структура MFT.

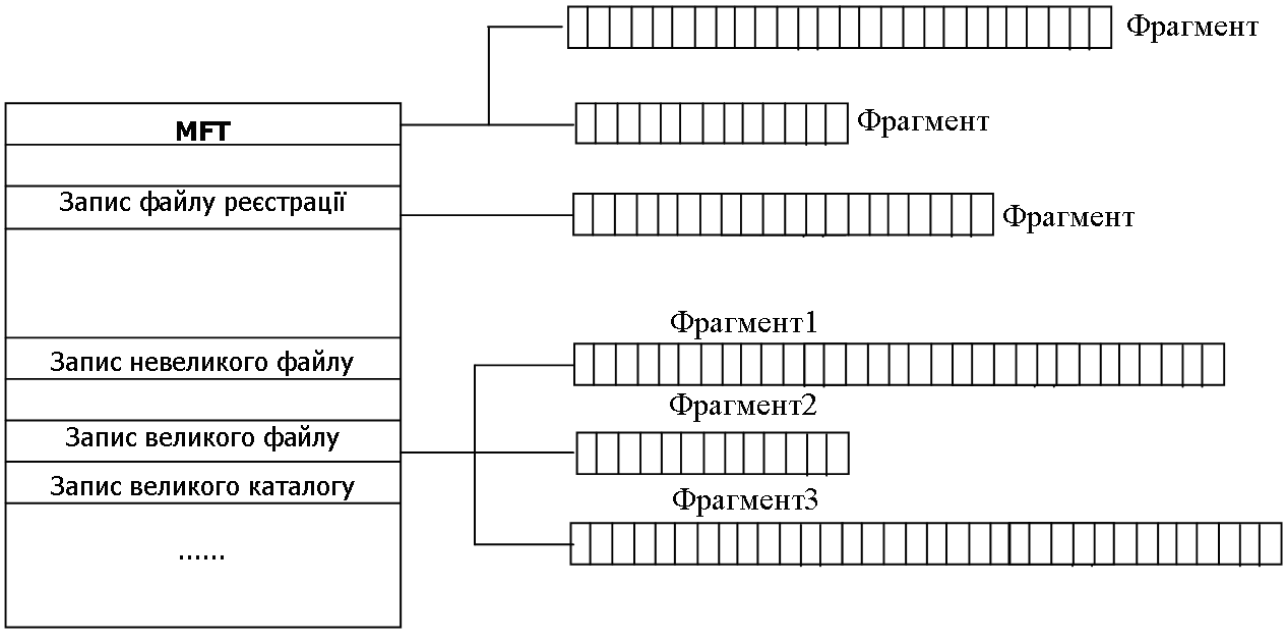

Рис. 1.5 Організація головної файлової таблиці

Головна файлова таблиця відводить певну кількість простору для кожного запису файлу. Атрибути файлу записуються в розподілений простір MFT.

Невеликі файли і каталоги (зазвичай до 1500 байт або менше) можуть повністю міститись усередині запису головної файлової таблиці. Такий підхід забезпечує дуже швидкий доступ до файлів.

Стандартна інформація	Ім'я файлу або каталогу	Дескриптор безпеки	Дані або покажчик	….

Рис. 1.6 Розподілений простір MFT

Запис MFT для невеликого файлу або каталогу.

В NTFS пошук файлу проводиться тільки для безпосереднього його використовування. Записи каталогу поміщені усередині головної файлової таблиці так само, як записи файлу. Замість даних, каталоги містять індексну інформацію. Невеликі записи каталогів знаходяться повністю усередині структури MFT. Великі каталоги організовані в B-tree, маючи записи з покажчиками на зовнішні кластери.

Перші 16 файлів NTFS (метафайли) носять службовий характер. Кожний з них відповідає за який-небудь аспект роботи системи. Перевага модульного підходу полягає у вражаючій гнучкості - наприклад, на fat-і фізичне пошкодження в самій області fat фатально для функціонування всього диска, а NTFS може змістити, навіть фрагментувати по диску, всі свої службові області, обійшовши будь-які несправності поверхні - окрім перші 16 елементів MFT.

Метафайли знаходяться у кореневому каталозі NTFS диска - вони починаються з символу імені "$", хоча отримати яку-небудь інформацію про них стандартними засобами складно. Для цих файлів вказаний цілком реальний розмір - можна довідатись, наприклад, скільки операційна система витрачає на каталогізацію всього диска, подивившись розмір файлу $MFT. В наступній таблиці наведені ті метафайли, що використовуються на даний момент метафайли і їх призначення.

Табл. 1.11 Метафайли

Назва	Опис
$MFT	відповідно сам MFT
$MFTmirr	копія перші 16 записів MFT, розміщена посередині диска
$logfile	файл підтримки журналювання (див. нижче)
$volume	службова інформація - мітка тому, версія файлової системи, т.д.
$attrdef	список стандартних атрибутів файлів на томі
$.	кореневий каталог
$bitmap	карта вільного місця тому
$boot	завантажувальний сектор (якщо розділ завантажувальний)
$quota	файл, в якому записані права користувачів на використовування дискового простору (почав працювати лише в nt5)
$upcase	файл - таблиця відповідності заголовних і прописних букв в іменах файлів на поточному томі. Потрібен в основному тому, що в NTFS імена файлів записуються в unicode, що складає 65 тисяч різних символів, шукати великі і малі еквіваленти яких дуже нетривіально.

Атрибути файлу NTFS. NTFS проглядає кожний файл (або каталог), як набір атрибутів файлу. Такі елементи, як ім'я файлу, зашита інформація про файл і навіть дані — все це атрибути файлу. Кожний атрибут ідентифікований кодом типа атрибута, але необов'язково ім'ям атрибута.

Якщо атрибути файлу можуть знаходиться всередині запису файлу MFT, вони називаються резидентними (resident) атрибутами. Інформація типу імені файлу і відмітки часу завжди включається в запис файлу MFT. Якщо файл дуже великий, щоб містити всі атрибути в записі файлу MFT, частина атрибутів є нерезидентною (nonresident). Нерезидентні атрибути займають один або декілька пробігів (run) дискового простору у іншому місці тому (пробіг дискового простору — безперервна лінійна область на диску). Взагалі, всі атрибути можуть бути викликані як потік байтів незалежно від того, чи є вони резидентними або нерезидентними.

В таблиці представлений список всіх атрибутів файлу, на даний час визначених для NTFS. Цей список можна розширити, тобто інші атрибути файлу в майбутньому можуть бути визначені у разі потреби.

Табл. 1.12 Атрибути файлу NTFS

Тип атрибута	Опис
Attribute List (список атрибутів)	Перераховує всі інші атрибути (тільки у великих файлах)
Filename (ім'я файлу)	Атрибут, що повторюється для довгих і для коротких імен файлів. Довге ім'я файлу може містити до 255 символів Unicode. Коротке ім'я — доступно, вісім плюс три символи, без урахування регістра. Додаткові імена, або жорсткі зв'язки (hard links), використовуються POSIX і можуть бути також включені як додаткові атрибути імені файлу
Security Descriptor (дескриптор безпеки)	Фіксує інформацію про те, хто може звертатися до файлу, хто є його власником і так далі
Data (дані)	Містить дані файлу
Index Root (корінь індексів)	Використовується при роботі з каталогами
Index Allocation (індексне розміщення)	Використовується при роботі з каталогами
Volume Information (інформація тому)	Використовується тільки в системному файлі тому і включає зокрема версію і ім'я тому
Bitmap (бітовий масив)	Надає інформацію про використовування записів в MFT або каталозі
Extended Attribute Information (інформація розширеного атрибута)	Використовується файловими серверами, які пов'язані з системами OS/2 Цей тип атрибута не використовується Windows NT
Extended Attributes (розширені атрибути)	Використовується файловими серверами, які пов'язані з системами OS/2 Цей тип атрибута не використовується Windows NT

NTFS підтримує імена файлів до 255 символів. Імена файлів NTFS використовують набір символів Unicode з 16 бітами; проте питання доступу вирішено. NTFS автоматично генерує підтримуване MS-DOS ім'я (вісім плюс три символи) для кожного файлу.

Таким чином, файли NTFS можуть використовуватися через мережу. Це особливо важливо для файлових серверів організації, яка використовує персональні комп'ютери з двома або всіма трьома даними операційними системами.

Як і будь-яка інша система, NTFS ділить все корисне місце на кластери - блоки даних, що використовуються одноразово. NTFS підтримує майже будь-які розміри

кластерів - від 512 байт до 64 Кбайт, деяким стандартом же вважається кластер розміром 4 Кбайт.

Ніяких аномалій кластерної структури NTFS не має. Пошук вільного місця в NTFS проводиться в тому випадку, якщо файл потрібно створити з нуля або скопіювати на диск. Пошук місця під фізичні дані файлу залежить від того, як зберігається інформація про зайняті ділянки диска. NTFS має бітову карту вільного місця, одному кластеру відповідає 1 біт.

Для пошуку файлу з даним ім'ям в лінійному каталозі, такому, наприклад, як у FAT, операційній системі доводиться проглядати всі елементи каталогу, поки вона не знайде потрібний. Оскільки, внутрішня структура каталогу є бінарним деревом, то імена файлів розташовуються так, щоб пошук файлу здійснювався більш швидким способом - за допомогою отримання двозначних відповідей на питання про положення файлу.

Питання, на яке бінарне дерево здатне дати відповідь, таке: в якій групі, відносно даного елемента, знаходиться потрібне ім'я - вище або нижче?

Ми починаємо з такого питання до середнього елемента, і кожна відповідь звужує зону пошуку в середньому в два рази, як показано на рисунку 1.7. Файли, наприклад, просто відсортовані за абеткою, і відповідь на питання здійснюється очевидним способом - порівнянням початкових літер.

Область пошуку, звужена в два рази, починає досліджуватися аналогічним чином, починаючи знову ж таки з середнього елемента.

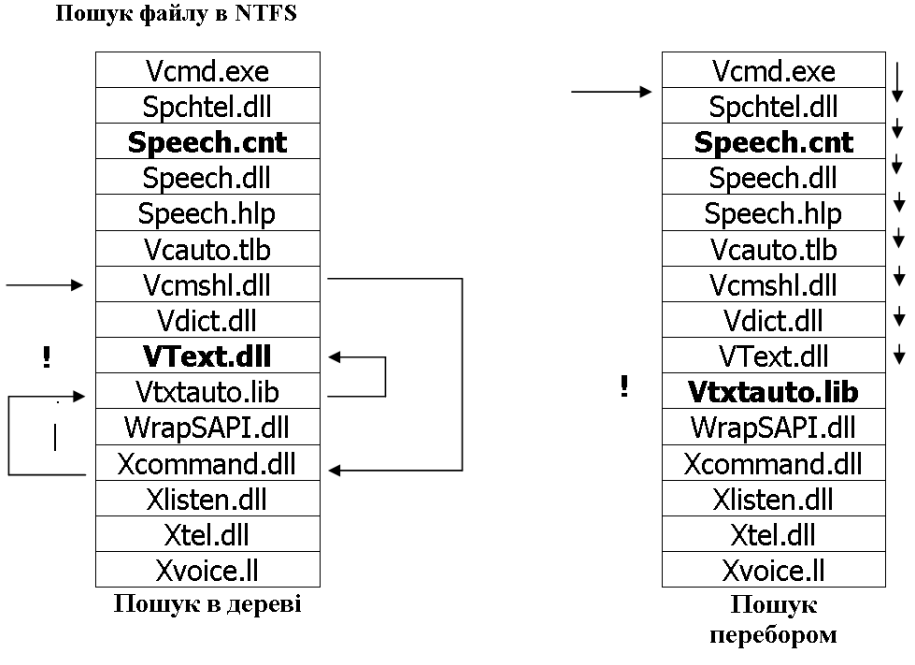

Рис. 1.7 Пошук файлу в NTFS

NTFS - відмовостійка система, яка цілком може привести себе в коректний стан при практично будь-яких реальних збоях. Будь-яка з сучасних файлових систем заснована на такому понятті, як транзакція - дія, яка виконується цілком і коректно або не виконується взагалі. У NTFS просто не буває проміжних (помилкових або некоректних) станів - квант зміни даних не може бути поділений на “до” і “після” збою, приносячи руйнування і незрозумілість - він або є і як наслідок певний результат, або його відмінено.

Файли NTFS мають один досить корисний атрибут - "стислий". Річ у тому, що NTFS має вбудовану підтримку стиснення дисків - те, для чого раніше доводилося використовувати stacker або doublespace. Будь-який файл або каталог в індивідуальному порядку може зберігається на диску в стислому вигляді - цей процес абсолютно прозорий для додатків.

Стиснення файлів має дуже високу швидкість і лише одна велика негативна властивість - величезна віртуальна фрагментація стислих файлів, яка, правда, нікому особливо не заважає. Стиснення здійснюється блоками по 16 кластерів і використовує так звані "віртуальні кластери" - знову ж таки гранично гнучке рішення, що дозволяє добитися цікавих ефектів - наприклад, половина файлу може бути стиснена, а половина - ні. Це досягається завдяки тому, що зберігання інформації про компресування певних фрагментів дуже схоже на звичайну фрагментацію файлів: наприклад, типовий запис фізичної розкладки для реального, нестислого, файлу:

- кластери файлу з 1 по 43-й зберігаються в кластерах диска починаючи з 400-м;

- кластери файлу з 44 по 52-й зберігаються в кластерах диска починаючи з 8530-м.

Фізична розкладка типового стислого файлу:

- кластери файлу з 1 по 9-й зберігаються в кластерах диска починаючи з 400-м;

- кластери файлу з 10 по 16-й ніде не зберігаються;

- кластери файлу з 17 по 18-й зберігаються в кластерах диска починаючи з 409-м;

- кластери файлу з 19 по 36-й ніде не зберігаються.

Стислий файл має "віртуальні кластери", як показано на рисунку, реальної інформації в яких немає. Як тільки система бачить такі віртуальні кластери, вона відразу розуміє, що дані попереднього блоку, кратного 16-ти, повинні бути не стиснені, а дані, що вийшли, якраз заповнять віртуальні кластери - ось, по суті, і весь алгоритм.

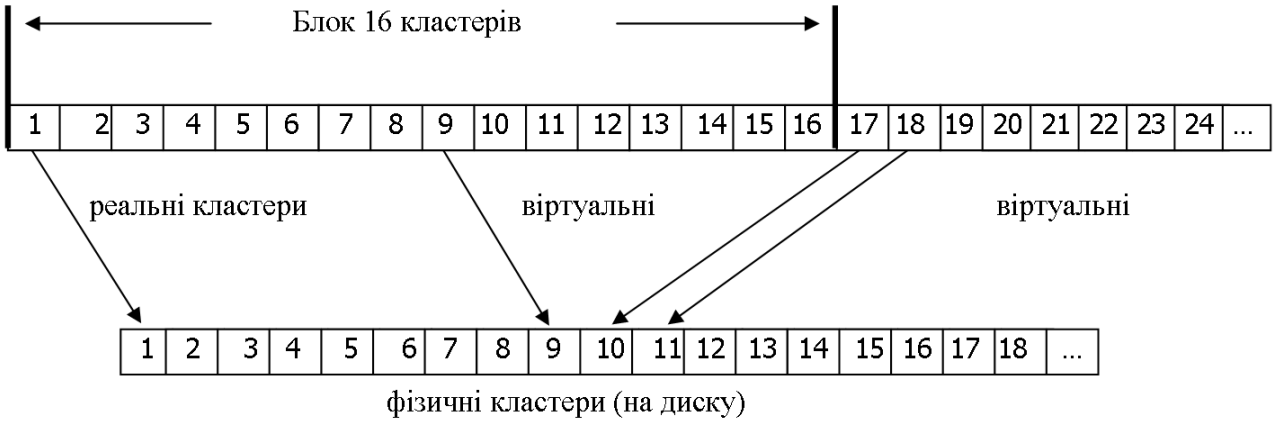

Рис. 1.8 Стиснення файлу в NTFS

NTFS містить безліч засобів розмежування прав об'єктів - є думка, що це найдосконаліша файлова система із всіх нині існуючих.

Недоліки та переваги FAT та NTFS.

FAT - переваги:

- Для ефективної роботи вимагається трохи оперативної пам'яті;
- Швидка робота з малими і середніми каталогами;
- Диск виконує в середньому меншу кількість рухів головок (порівняно з NTFS);
- Ефективна робота на повільних дисках.

FAT - недоліки:

- Катастрофічна втрата швидкодії із збільшенням фрагментації, особливо для великих дисків (тільки FAT32);
- Складнощі з довільним доступом до великих (скажімо, 10% і більше від розміру диска) файлів;
- Дуже повільна робота з каталогами, що містять велику кількість файлів.

NTFS - переваги:

- Фрагментація файлів не має практично ніяких наслідків для самої файлової системи;

- Робота фрагментованої системи погіршується тільки з погляду доступу до самих даних файлів;

- Складність структури каталогів і число файлів в одному каталозі не чинить особливих перешкод швидкодії;

- Швидкий доступ до довільного фрагмента файлу (наприклад, редагування великих wav файлів);

- Дуже швидкий доступ до маленьких файлів (декілька сотень байт) - весь файл знаходиться в тому ж місці, де і системні дані (запис MFT).

NTFS - недоліки:

- Істотні вимоги до пам'яті системи (64 Мбайт - абсолютний мінімум);

- Повільні диски і контроллери без Bus Mastering сильно знижують швидкодію NTFS;

- Робота з каталогами середніх розмірів ускладнена тим, що вони майже завжди фрагментовані;

- Диск, що довго працює в заповненому на 80% - 90% стані, показуватиме дуже низьку швидкодію.

Організація даних на лазерному диску.

Сектори тома організовані у вигляді логічних секторів. Зазвичай розмір логічного сектору складає 2048 байтів, однак стандарт допускає також використання секторів, що мають розмір 512 чи 1024 байта. Логічний сектор може складатися з декількох розташованих послідовно фізичних секторів. Логічні сектори не повинні перекриватися. Ідентифікувати логічний сектор можна за унікальним номером (Logical Sector Number). Нумерація логічних секторів починається з нуля. Інформація в кожному томі диска представлена у вигляді набору логічних секторів. Такий набір секторів називається простором тому (Volume Space). Простір тому організований у виді одномірного (лінійного) масиву байтів. Байти в просторі тому нумеруються послідовно, починаючи з одиниці. Перший байт простору тому відповідає першому байту першого вхідного до складу тому логічного сектора.