Systemy plików

Maciek Kowalczyk
Mateusz Patelak

12 stycznia 2006

Pożądane cechy współczesnych dyskowych systemów plików

1. Odporność struktury systemu plików na utraty zasilania
2. Odporność danych na utraty zasilania
3. Szybkość podnoszenia po nieprawidłowym zamknięciu
4. Bezpieczeństwo danych:
   - rozszerzone uprawnienia użytkowników (ACL)
   - szyfrowanie
5. Pliki rozrzedzone oraz kompresja
6. Odporność na fragmentację

Problemy ze złożonymi operacjami

- Operacje na systemie plików nie są atomowe.

Problemy ze złożonymi operacjami

- Operacje na systemie plików nie są atomowe.
- Utrata zasilania w trakcie modyfikacji systemu plików, w najlepszym przypadku, oznacza zapisanie w pliku niechcianych danych!

Problemy ze złożonymi operacjami

- Operacje na systemie plików nie są atomowe.
- Utrata zasilania w trakcie modyfikacji systemu plików, w najlepszym przypadku, oznacza zapisanie w pliku niechcianych danych!
- Może być jednak dużo gorzej, zaburzenie spójności systemu plików ma tendencję do rozszerzania się.

Problemy ze złożonymi operacjami

- Operacje na systemie plików nie są atomowe.
- Utrata zasilania w trakcie modyfikacji systemu plików, w najlepszym przypadku, oznacza zapisanie w pliku niechcianych danych!
- Może być jednak dużo gorzej, zaburzenie spójności systemu plików ma tendencję do rozszerzania się.
- Sprawdzanie systemu plików trwa długo.

Metoda pierwsza:

- Zrezygnujmy z tradycyjnej struktury na rzecz dziennika.
- Zapisujemy w nim po kolei wszystkie operacje do wykonywania.

Metoda pierwsza:

- Zrezygnujmy z tradycyjnej struktury na rzecz dziennika.
- Zapisujemy w nim po kolei wszystkie operacje do wykonywania.

Otrzymaliśmy tzw. dziennikowy system plików (log-structured FS)

Metoda pierwsza:

- Zrezygnujmy z tradycyjnej struktury na rzecz dziennika.
- Zapisujemy w nim po kolei wszystkie operacje do wykonywania.

Otrzymaliśmy tzw. dziennikowy system plików (log-structured FS)

- Naprawa polega na cofnięciu się na początek ostatniej transakcji.
- Mamy bardzo szybki zapis.
- Mamy dostęp do starych wersji plików (snapshot)

Metoda pierwsza:

- Zrezygnujmy z tradycyjnej struktury na rzecz dziennika.
- Zapisujemy w nim po kolei wszystkie operacje do wykonywania.

Otrzymaliśmy tzw. dziennikowy system plików (log-structured FS)

- Naprawa polega na cofnięciu się na początek ostatniej transakcji.
- Mamy bardzo szybki zapis.
- Mamy dostęp do starych wersji plików (snapshot)

Na podobne zasadzie działają systemy plików: UDF, JFFS2

Metoda druga:

- Do tradycyjnej struktury dodajemy cykliczny dziennik.

Metoda druga:

- Do tradycyjnej struktury dodajemy cykliczny dziennik.
- Operacje będące częścią transakcji zapisujemy do dziennika.

Metoda druga:

- Do tradycyjnej struktury dodajemy cykliczny dziennik.
- Operacje będące częścią transakcji zapisujemy do dziennika.
- Transakcję kończymy wstawiając zapis "commit".

Metoda druga:

- Do tradycyjnej struktury dodajemy cykliczny dziennik.
- Operacje będące częścią transakcji zapisujemy do dziennika.
- Transakcję kończymy wstawiając zapis "commit".
- Co jakiś czas, lub gdy dziennik zacznie się zbytnio zapełniać, wykonujemy zlecone operacje (flush).

Metoda druga:

- Do tradycyjnej struktury dodajemy cykliczny dziennik.
- Operacje będące częścią transakcji zapisujemy do dziennika.
- Transakcję kończymy wstawiając zapis "commit".
- Co jakiś czas, lub gdy dziennik zacznie się zbytnio zapełniać, wykonujemy zlecone operacje (flush).

Metoda druga:

- Do tradycyjnej struktury dodajemy cykliczny dziennik.
- Operacje będące częścią transakcji zapisujemy do dziennika.
- Transakcję kończymy wstawiając zapis "commit".
- Co jakiś czas, lub gdy dziennik zacznie się zbytnio zapełniać, wykonujemy zlecone operacje (flush).

Technika ta nazywana jest journalingiem (kronikowaniem,księgowaniem). Korzysta z niej większość nowoczesnych systemów plików.

Metoda druga:

- Do tradycyjnej struktury dodajemy cykliczny dziennik.
- Operacje będące częścią transakcji zapisujemy do dziennika.
- Transakcję kończymy wstawiając zapis "commit".
- Co jakiś czas, lub gdy dziennik zacznie się zbytnio zapełniać, wykonujemy zlecone operacje (flush).

Technika ta nazywana jest journalingiem (kronikowaniem,księgowaniem).
Korzysta z niej większość nowoczesnych systemów plików.
Na czym teraz polega sprawdzenie?

Metoda druga:

- Do tradycyjnej struktury dodajemy cykliczny dziennik.
- Operacje będące częścią transakcji zapisujemy do dziennika.
- Transakcję kończymy wstawiając zapis "commit".
- Co jakiś czas, lub gdy dziennik zacznie się zbytnio zapełniać, wykonujemy zlecone operacje (flush).

Technika ta nazywana jest journalingiem (kronikowaniem,księgowaniem).
Korzysta z niej większość nowoczesnych systemów plików.
Na czym teraz polega sprawdzenie?

- Odrzucamy operacje z dziennika po ostatnim commicie.

Metoda druga:

- Do tradycyjnej struktury dodajemy cykliczny dziennik.
- Operacje będące częścią transakcji zapisujemy do dziennika.
- Transakcję kończymy wstawiając zapis "commit".
- Co jakiś czas, lub gdy dziennik zacznie się zbytnio zapełniać, wykonujemy zlecone operacje (flush).

Technika ta nazywana jest journalingiem (kronikowaniem,księgowaniem).
Korzysta z niej większość nowoczesnych systemów plików.
Na czym teraz polega sprawdzenie?

- Odrzucamy operacje z dziennika po ostatnim commicie.
- Wykonujemy zaległe operacje.

Pożądane cechy współczesnych dyskowych systemów plików

1. Odporność struktury systemu plików na utraty zasilania
2. Odporność danych na utraty zasilania
3. Szybkość podnoszenia po nieprawidłowym zamknięciu
4. Bezpieczeństwo danych:
   - rozszerzone uprawnienia użytkowników (ACL)
   - szyfrowanie
5. Pliki rozrzedzone oraz kompresja
6. Odporność na fragmentację

Główne cechy systemu ext3

Obustronna kompatybilność z ext2

Jedyną różnicą pomiędzy tymi systemami jest obecność dziennika.

Konwersja z ext2 do ext3

Dodanie dziennika:

```
tune2fs -j /dev/hdaX
```

Może on być również utworzony na innym dysku

Główne cechy systemu ext3

Obustronna kompatybilność z ext2

Jedyną różnicą pomiędzy tymi systemami jest obecność dziennika.

Konwersja z ext2 do ext3

Dodanie dziennika:

```
tune2fs -j /dev/hdaX
```

Może on być również utworzony na innym dysku

Ponadto partycję ext3 można zamontować jako ext2 i odwrotnie.

Główne cechy systemu ext3

Journaling

Trzy tryby journalingu:

1. tylko metadanych, z opóźnionym zapisem (writeback)
   najszybszy, ale nie daje gwarancji zapisania danych na dysk.

Główne cechy systemu ext3

Journaling

Trzy tryby journalingu:

1. tylko metadanych, z opóźnionym zapisem (writeback)
   najszybszy, ale nie daje gwarancji zapisania danych na dysk.
2. tylko metadanych (ordered), domyślny Dane są zapisywane przed commitem.

Główne cechy systemu ext3

Journaling

Trzy tryby journalingu:

1. tylko metadanych, z opóźnionym zapisem (writeback)
   najszybszy, ale nie daje gwarancji zapisania danych na dysk.
2. tylko metadanych (ordered), domyślny Dane są zapisywane przed commitem.
3. pełny (journal) W logu są zapisywane także zmiany metadanych.

Główne cechy systemu ext3

Journaling

Physical journaling

Charakterystyczne dla systemu ext3 jest zapisywanie w logu całych bloków z metadanymi, nie zaś tylko samych modyfikacji.

Główne cechy systemu ext3

Indeksowanie katalogów

Hash Trees

Zmodyfikowane B-drzewa, które jako klucza używają wartości funkcji mieszającej nazwy plików.

Wraz z ext3 zostało wprowadzone indeksowanie większych katalogów przy pomocy H-trees.

Główne cechy systemu ext3

Indeksowanie katalogów

Hash Trees

Zmodyfikowane B-drzewa, które jako klucza używają wartości funkcji mieszającej nazwy plików.

Wraz z ext3 zostało wprowadzone indeksowanie większych katalogów przy pomocy H-trees.
Obecnie ta funkcja jest również dostępna w systemie ext2.

Główne cechy systemu ext3

Ext3 oficjalnie nie umożliwia kompresji ani szyfrowania plików. Rozszerzone atrybuty (przy pomocy nich zaimplementowano ACLe) mają ograniczoną wielkość.

Pochodzenie systemu reiser4

System został stworzony w 2004 roku przez firmę Namesys.

ReiserFS

Poprzednik był dołączany do jądra od wczesnych wersji 2.4.
Jego główną zaletą była duża szybkość pracy z małymi plikami.

Pochodzenie systemu reiser4

System został stworzony w 2004 roku przez firmę Namesys.

ReiserFS

Poprzednik był dołączany do jądra od wczesnych wersji 2.4. Jego główną zaletą była duża szybkość pracy z małymi plikami.

Nowa wersja została napisana od podstaw, ale do tej pory nie weszła w skład głównej wersji jądra.

Główne cechy systemu reiser4

Budowa

Szkieletem systemu plików jest tu "tańczące drzewo".

Dancing Tree

Jest to zmodyfikowane B*-drzewo, które nie jest balansowane przy każdej operacji, a tylko po:

- wykonaniu flush
- upływie określonego odstępu czasu

Główne cechy systemu reiser4

Journaling

Reiser4 wykorzystuje pełne kronikowanie, razem z danymi. Zapewnia więc, że dane są zapisywane atomowo.

Główne cechy systemu reiser4

Journaling

Reiser4 wykorzystuje pełne kronikowanie, razem z danymi.
Zapewnia więc, że dane są zapisywane atomowo.
Jednak w tym przypadku dane nie są zapisywane dwukrotnie.
Użyto triku polegającego na zamianie wskaźników do bloku docelowego i części logu.

Główne cechy systemu reiser4

Journaling

Reiser4 wykorzystuje pełne kronikowanie, razem z danymi.
Zapewnia więc, że dane są zapisywane atomowo.
Jednak w tym przypadku dane nie są zapisywane dwukrotnie.
Użyto triku polegającego na zamianie wskaźników do bloku docelowego i części logu.
Oczywiście taka zamiana powoduje konieczność dalszych zapisów, jednak ze względu na strukturę drzewiastą systemu plików, ich ilość będzie proporcjonalna do wysokości drzewa.

Główne cechy systemu reiser4

Journaling

Reiser4 wykorzystuje pełne kronikowanie, razem z danymi.
Zapewnia więc, że dane są zapisywane atomowo.
Jednak w tym przypadku dane nie są zapisywane dwukrotnie.
Użyto triku polegającego na zamianie wskaźników do bloku docelowego i części logu.
Oczywiście taka zamiana powoduje konieczność dalszych zapisów, jednak ze względu na strukturę drzewiastą systemu plików, ich ilość będzie proporcjonalna do wysokości drzewa. System plików jest więc w pełni transakcyjny.
Autor twierdzi, że strata wydajności z tym związana jest z nawiązką pokrywana w innych obszarach.

Główne cechy systemu reiser4

Opóźniona alokacja

Jest to technika pochodząca z systemu plików XFS.
Polega na odroczeniu alokacji bloków do czasu zapisu na dysk.
Zmniejsza to fragmentację dysku, gdyż:

- Pozwala uniknąć alokacji miejsca dla krótko istniejących plików.
- Jeśli kilka nowoutworzonych plików rozrasta się jednocześnie to można je tak zaalokować, że nie będą się niepotrzebnie przeplatać.

Główne cechy systemu reiser4

Małe pliki i duże katalogi

Podobnie jak we wcześniejszej wersji reiser umożliwia upychanie małych plików i końcówek do jednego bloku.

Główne cechy systemu reiser4

Małe pliki i duże katalogi

Podobnie jak we wcześniejszej wersji reiser umożliwia upychanie małych plików i końcówek do jednego bloku.
Podobnie jak w ext3, katalogi są drzewami indeksowanymi hash'em nazwy pliku.

Główne cechy systemu reiser4

Wtyczki

Mechanizm plug-in'ów pozwala na implementację takich funkcji jak:

- Kompresja
- Szyfrowanie
- ACL

A także wielu innych. Daje to nadzieję na szybki rozwój tego systemu plików.

Najważniejsze z pozostałych systemów plików obsługiwanych przez Linux

- JFS - kronikowany system plików, "podarowany" przez IBM'a, przeniesiony na Linux'a w 2001 roku, w pełni 64-bitowy
- XFS - także kronikowany, stworzony przez SGI, był źródłem inspiracji dla twórców reiser'a
- FAT/VFAT - obsługiwany w pełni
- NTFS - bardzo ograniczone możliwości zapisu, brak obsługi zaawansowanych funkcji jak szyfrowanie, kompresja

Preferowany system plików dla rodziny systemów Windows NT. Systemy powstałe na bazie MS-DOS nie potrafią czytać NTFS, ale umożliwiają im to zewnętrzne aplikacje.

Główne cechy NTFS

Odporna na awarie struktura plików

- operacje wejścia-wyjścia jako transakcje

Główne cechy NTFS

Odporna na awarie struktura plików

- operacje wejścia-wyjścia jako transakcje
- dotyczy tylko metadanych (struktur systemu plików)

Główne cechy NTFS

Odporna na awarie struktura plików

- operacje wejścia-wyjścia jako transakcje
- dotyczy tylko metadanych (struktur systemu plików)
- logowanie samych zmian w danych

Główne cechy NTFS

Odporna na awarie struktura plików

- zapisanie podoperacji transakcji w logu
- wykonanie podoperacji transakcji w cache woluminu
- zaznaczenie w logu, że transakcja została wykonana

Główne cechy NTFS

Odporna na awarie struktura plików

- zapisanie podoperacji transakcji w logu
- wykonanie podoperacji transakcji w cache woluminu
- zaznaczenie w logu, że transakcja została wykonana

Główne cechy NTFS

Odporna na awarie struktura plików

- zapisanie podoperacji transakcji w logu
- wykonanie podoperacji transakcji w cache woluminu
- zaznaczenie w logu, że transakcja została wykonana

Główne cechy NTFS

Odporna na awarie struktura plików

- zapisanie podoperacji transakcji w logu
- wykonanie podoperacji transakcji w cache woluminu
- zaznaczenie w logu, że transakcja została wykonana

Nagły brak prądu?

Po włączeniu NTFS ponawia zatwierdzone transakcje znajdujące się w logu i cofa podoperacje niezatwierdzonych transakcji

Główne cechy NTFS

Bezpieczeństwo danych

- szybkie szyfrowanie plików i katalogów

Główne cechy NTFS

Bezpieczeństwo danych

- szybkie szyfrowanie plików i katalogów
- złożone uprawnienia użytkowników w dostępie do plików

Główne cechy NTFS

Bezpieczeństwo danych

Szyfrowanie - jak to działa?

- klucz symetryczny (File Encryption Key, FEK, algorytm DESX 128 bit, ale można też 3DES 168 bit)

Główne cechy NTFS

Bezpieczeństwo danych

Szyfrowanie - jak to działa?

- klucz symetryczny (File Encryption Key, FEK, algorytm DESX 128 bit, ale można też 3DES 168 bit)
- FEK zaszyfrowany kluczem publicznym związanym z użytkownikiem, trzymany jako alternatywny strumień w pliku (algorytm RSA)

Główne cechy NTFS

Bezpieczeństwo danych

Szyfrowanie - jak to działa?

- klucz symetryczny (File Encryption Key, FEK, algorytm DESX 128 bit, ale można też 3DES 168 bit)
- FEK zaszyfrowany kluczem publicznym związanym z użytkownikiem, trzymany jako alternatywny strumień w pliku (algorytm RSA)
- aby odszyfrować plik, używany jest prywatny klucz użytkownika do klucza symetrycznego trzymanego w nagłówku zaszyfrowanego pliku

Główne cechy NTFS

Bezpieczeństwo danych

Szyfrowanie - jak to działa?

- klucz symetryczny (File Encryption Key, FEK, algorytm DESX 128 bit, ale można też 3DES 168 bit)
- FEK zaszyfrowany kluczem publicznym związanym z użytkownikiem, trzymany jako alternatywny strumień w pliku (algorytm RSA)
- aby odszyfrować plik, używany jest prywatny klucz użytkownika do klucza symetrycznego trzymanego w nagłówku zaszyfrowanego pliku
- za pomocą odszyfrowanego klucza, odszyfrowywany jest sam plik

Główne cechy NTFS

Bezpieczeństwo danych

Szyfrowanie - jak to działa?

- klucz symetryczny (File Encryption Key, FEK, algorytm DESX 128 bit, ale można też 3DES 168 bit)
- FEK zaszyfrowany kluczem publicznym związanym z użytkownikiem, trzymany jako alternatywny strumień w pliku (algorytm RSA)
- aby odszyfrować plik, używany jest prywatny klucz użytkownika do klucza symetrycznego trzymanego w nagłówku zaszyfrowanego pliku
- za pomocą odszyfrowanego klucza, odszyfrowywany jest sam plik
- dwa różne systemy, bo symetrycznym ok. 1000 razy szybciej

Główne cechy NTFS

Wady szyfrowania NTFS

- przy szyfrowaniu tworzone są tymczasowe pliki z niezaszyfrowaną zawartością szyfrowanego pliku (pliki te są potem tylko zaznaczone do usunięcia - lepiej szyfrować katalogi)

Główne cechy NTFS

Wady szyfrowania NTFS

- przy szyfrowaniu tworzone są tymczasowe pliki z niezaszyfrowaną zawartością szyfrowanego pliku (pliki te są potem tylko zaznaczone do usunięcia - lepiej szyfrować katalogi)
- nazwy plików i folderów nie są szyfrowane (rozwiązanie: zzipować)

Główne cechy NTFS

Bezpieczeństwo danych

Gdzie jest klucz prywatny?

Windows crhoni wszystkie prywatne klucze szyfrując je za pomocą Protected Storage service. Protected Storage używa do szyfrowania Session Key otrzymanego z 512-bitowym Master Key. Master Key jest zaszyfrowany przez Master Key Encryption Key, który jest otrzymany z hasła użytkownika, przy użyciu funkcji Password Based Key Derivation Function.

Główne cechy NTFS

Alternatywne strumienie danych

- pliki mogą się składać z jednego lub więcej strumieni (ADS - Alternate data streams)

Główne cechy NTFS

Alternatywne strumienie danych

- pliki mogą się składać z jednego lub więcej strumieni (ADS - Alternate data streams)
- jeden strumień nienazwany i dowolna ilość nazwanych

Główne cechy NTFS

Alternatywne strumienie danych

- pliki mogą się składać z jednego lub więcej strumieni (ADS - Alternate data streams)
- jeden strumień nienazwany i dowolna ilość nazwanych
- dodatkowe strumienie mogą zawierać dowolne dane

Główne cechy NTFS

Alternatywne strumienie danych

- pliki mogą się składać z jednego lub więcej strumieni (ADS - Alternate data streams)
- jeden strumień nienazwany i dowolna ilość nazwanych
- dodatkowe strumienie mogą zawierać dowolne dane
- zazwyczaj są to dane opisujące plik lub metadane

Przykład

```
plik.txt:sekret.txt
```

Główne cechy NTFS

Alternatywne strumienie danych

- pliki mogą się składać z jednego lub więcej strumieni (ADS - Alternate data streams)
- jeden strumień nienazwany i dowolna ilość nazwanych
- dodatkowe strumienie mogą zawierać dowolne dane
- zazwyczaj są to dane opisujące plik lub metadane

Przykład

`plik.txt:sekret.txt`

Nie można wykryć `sekret.txt` poprzez rozmiar pliku `plik.txt`!

Główne cechy NTFS

Oszczędność miejsca

- kompresja plików i katalogów

Główne cechy NTFS

Oszczędność miejsca

- kompresja plików i katalogów
  - algorytm LZ77

Główne cechy NTFS

Oszczędność miejsca

- kompresja plików i katalogów
  - algorytm LZ77
  - niezależna kompresja bloków X klastrów

Główne cechy NTFS

Oszczędność miejsca

- kompresja plików i katalogów
  - algorytm LZ77
  - niezależna kompresja bloków X klastrów
- Sparse Files - obsługa rzadkich plików (mających klastry wypełnione zerami)

Główne cechy NTFS

Fragmentacja

Defragmentation API

Pozwala aplikacjom na defragmentację plików poprzez API (mapa klastrów, które są używane i nie, mapa klastrów używanych przez plik)

Główne cechy NTFS

Fragmentacja

Reparse points

Jeśli plik lub katalog ma znacznik reparse, dane zostają przekazane dane do sterowników filtrów, które są załadowane w systemie operacyjnym. Sterowniki sprawdzają, czy dane pasują i, jeśli tak, wykonują swoją specjalną funkcjonalność na pliku/katalogu. Używane m.in. w Volume Mount Points, Directory Junctions.

Główne cechy NTFS

Inne cechy

1. nazwy plików Unicode (2 bajty na znak), poza NUL

Główne cechy NTFS

Inne cechy

1. nazwy plików Unicode (2 bajty na znak), poza NUL
2. indeksowanie atrybutów (Collations) - B*drzewa

Główne cechy NTFS

Inne cechy

1. nazwy plików Unicode (2 bajty na znak), poza NUL
2. indeksowanie atrybutów (Collations) - B*drzewa
3. directory junctions - referencje do innych katalogów

Główne cechy NTFS

Inne cechy

1. nazwy plików Unicode (2 bajty na znak), poza NUL
2. indeksowanie atrybutów (Collations) - B*drzewa
3. directory junctions - referencje do innych katalogów
4. (POSIX style) hard links - twarde dowiązania do plików na tym samym woluminie (drugi wpis w MFT)

Główne cechy NTFS

Inne cechy

1. distributed link tracking

Główne cechy NTFS

Inne cechy

1. distributed link tracking
2. dynamiczne remapowanie złych klastrów

Główne cechy NTFS

Inne cechy

1. distributed link tracking
2. dynamiczne remapowanie złych klastrów
3. stemple czasowe w formacie GMT/UTC

Główne cechy NTFS

Inne cechy

1. distributed link tracking
2. dynamiczne remapowanie złych klastrów
3. stemple czasowe w formacie GMT/UTC
4. Quota - nie bierze pod uwagę kompresji NTFS, chyba że zostanie to ustawione

Główne cechy NTFS

Inne cechy

1. distributed link tracking
2. dynamiczne remapowanie złych klastrów
3. stemple czasowe w formacie GMT/UTC
4. Quota - nie bierze pod uwagę kompresji NTFS, chyba że zostanie to ustawione
5. Hierarchical Storage Management - transfer nieużywanych plików do innego medium (używa reparse point)

Główne cechy NTFS

Ograniczenia

- długość ścieżki do 32000 Unicode, pliku do 255 znaków

Główne cechy NTFS

Ograniczenia

- długość ścieżki do 32000 Unicode, pliku do 255 znaków
- zarezerwowane nazwy plików - metadane są trzymane w zwykłych, ale ukrytych i w większości niedostępnych plikach w katalogu głównym

Główne cechy NTFS

Ograniczenia

- długość ścieżki do 32000 Unicode, pliku do 255 znaków
- zarezerwowane nazwy plików - metadane są trzymane w zwykłych, ale ukrytych i w większości niedostępnych plikach w katalogu głównym
- maksymalna wielkość woluminu - 2^{64-1} klastrów (adresowanie 64-bitowe), ale w Windows XP tylko 2^{32-1} klastrów, co przy klastrze domyślnej wielkości (4 KB) daje prawie 16 TB

Główne cechy NTFS

Ograniczenia

- długość ścieżki do 32000 Unicode, pliku do 255 znaków
- zarezerwowane nazwy plików - metadane są trzymane w zwykłych, ale ukrytych i w większości niedostępnych plikach w katalogu głównym
- maksymalna wielkość woluminu - 2^{64-1} klastrów (adresowanie 64-bitowe), ale w Windows XP tylko 2^{32-1} klastrów, co przy klastrze domyślnej wielkości (4 KB) daje prawie 16 TB
- maksymalna wielkość plików - teoretycznie do 16 EB, ale w Windows XP tylko 16 TB

Struktura NTFS

Po sformatowaniu

- BOOT

Struktura NTFS

Po sformatowaniu

- BOOT
- MFT

Struktura NTFS

Po sformatowaniu

- BOOT
- MFT
- wolna przestrzeń

Struktura NTFS

Po sformatowaniu

- BOOT
- MFT
- wolna przestrzeń
- metadane

Struktura NTFS

Po sformatowaniu

- BOOT
- MFT
- wolna przestrzeń
- metadane
- wolna przestrzeń

Struktura NTFS

- wszystko jest plikiem

Struktura NTFS

- wszystko jest plikiem
- Master File Table jest indeksem plików (oraz plikiem)

Struktura NTFS

- wszystko jest plikiem
- Master File Table jest indeksem plików (oraz plikiem)
- $Boot jest plikiem zlokalizowanym na początku dysku, pokazuje gdzie znajduje się MFT

Struktura NTFS

- wszystko jest plikiem
- Master File Table jest indeksem plików (oraz plikiem)
- $Boot jest plikiem zlokalizowanym na początku dysku, pokazuje gdzie znajduje się MFT
- w centrum znajdują się takie pliki jak $MFTMirr i $LogFile

Struktura NTFS

- wszystko jest plikiem
- Master File Table jest indeksem plików (oraz plikiem)
- $Boot jest plikiem zlokalizowanym na początku dysku, pokazuje gdzie znajduje się MFT
- w centrum znajdują się takie pliki jak $MFTMirr i $LogFile
- na końcu woluminu znajduje się kopia boot sektora (klaster 0)

Struktura NTFS

- wszystko jest plikiem
- Master File Table jest indeksem plików (oraz plikiem)
- $Boot jest plikiem zlokalizowanym na początku dysku, pokazuje gdzie znajduje się MFT
- w centrum znajdują się takie pliki jak $MFTMirr i $LogFile
- na końcu woluminu znajduje się kopia boot sektora (klaster 0)
- wszystkie pliki poza $Boot mogą zostać przeniesione

Struktura NTFS

MFT Region

Aby zapobiec fragmentacji MFT Windows utrzymuje wokół niego bufor, gdzie żadne nowe pliki nie zostaną utworzone do czasu zapełnienia reszty woluminu.

Struktura NTFS

FILE Record

Każdy rekord MFT zbudowany jest z atrybutów. Lista tych atrybutów jest zdefiniowana w pliku $AttrDef.
Każdy atrybut rozpoczyna się standardowym nagłówkiem (Standard Header). Atrybuty mogą być rezydentne lub nie.

Struktura NTFS

Atrybuty

$STANDARD_INFORMATION

Zawiera takie informacje, jak stemple czasowe, identyfikator właściciela, informację o wielkości pliku (ze wszystkimi strumieniami, 0 jeśli quota wyłączona), DOS File Permissions, Security Id, Update Sequence Number (indeks w pliku $UsnJrnl).

Struktura NTFS

Atrybuty

$ATTRIBUTE_LIST

Potrzebny, gdy mało miejsca w rekordzie MFT. Zawiera informacje o typie i rozmieszczeniu w MFT wszystkich innych atrybutów należących do pliku.
Np. gdy plik ma dużo twardych dowiązań, staje się pofragmentowany, ma wiele nazwanych strumieni.

Struktura NTFS

Atrybuty

$FILE_NAME

Oprócz nazwy pliku (w Unicode) zawiera referencję do katalogu, w którym znajduje się plik a także flagi.
Każde twarde dowiązanie ma swój atrybut $FILE_NAME (aktualizowany tylko przy zmianie nazwy pliku).

Struktura NTFS

Atrybuty

$FILE_NAME

Oprócz nazwy pliku (w Unicode) zawiera referencję do katalogu, w którym znajduje się plik a także flagi.
Każde twarde dowiązanie ma swój atrybut $FILE_NAME (aktualizowany tylko przy zmianie nazwy pliku).

Read-Only, Hidden, System, Archive, Device, Normal, Temporary, Sparse File, Reparse Point, Compressed, Offline, Not Content Indexed, Encrypted, Directory, Index View

Struktura NTFS

Atrybuty

$OBJECT_ID

Unikalny identyfikator pliku

Struktura NTFS

Atrybuty

$SECURITY_DESCRIPTOR

Generalnie mówi o tym, kto ma jakie prawa do pliku, kto jest właścicielem. W nowych NTFS wszystkie te atrybuty są trzymane w pliku $Secure. Wskazuje tam Security Id. ACL (Access Control List) zawiera jeden lub więcej ACEs. Każdy ACE zawiera SID.

Struktura NTFS

ACE access mask

Bit	Znaczenie	Opis
0 - 15	Object Specific	Read, Execute, Append data
16 - 22	Standard	Delete, Write ACL, Write Owner
23	Can access security ACL	
24 - 27	Reserved	
28	Generic ALL	Everything below
29	Generic Execute	All necessary to execute
30	Generic Write	All necessary to write
31	Generic Read	All things to read

Struktura NTFS

Atrybuty

$VOLUME_NAME

Zawiera nazwę woluminu.

$VOLUME_INFORMATION

Informacja, która wersja NTFS, czy dysk wymaga sprawdzenia.

Struktura NTFS

Atrybuty

$DATA

Właśnie tutaj znajduje się zawartość pliku (może być kilka $Data - strumienie alternatywne, in. forki).

Struktura NTFS

Atrybuty

$INDEX_ROOT

Wierzchołek B*drzewa dla indeksów (np. dla katalogu). Zawiera listę wierzchołków, gdzie znajduje się plik. Jeśli katalog jest wystarczająco mały, mieści się właśnie tutaj.

Struktura NTFS

Atrybuty

$INDEX_ROOT

Wierzchołek B*drzewa dla indeksów (np. dla katalogu). Zawiera listę wierzchołków, gdzie znajduje się plik. Jeśli katalog jest wystarczająco mały, mieści się właśnie tutaj.

$INDEX_ALLOCATION

Miejsce dla wierzchołków B*drzewa opisywanego przez $INDEX_ROOT. Dla katalogu są to nazwy plików.

Struktura NTFS

Atrybuty

$BITMAP

1. W $MFT pokazuje, które jego rekordy są w użyciu.
2. W indeksach (np. katalogach) - która pozycja indeksu jest w użyciu (każdy bit, to jedna Virtual Cluster Number).

Struktura NTFS

Atrybuty

$REPARSE_POINT

Używane m.in. gdy plik jest łączem symbolicznym lub woluminem. Plik $Extend/$Reparse jest indeksem wszystkich reparse points na woluminie.

Struktura NTFS

Atrybuty

$EA_INFORMATION

Używane do implementacji rozszerzonych atrybutów.

Struktura NTFS

Atrybuty

$LOGGED_UTILITY_STREAM

Działa jak ograniczony nazwany strumień, ale operacje na nim zapisywane są do dziennika, jak zmiany normalnych metadanych. Używane np. przez EFS (Encrypting File System) - wszystkie zaszyfrowane pliki mają ten atrybut nazwany $EFS.

Struktura NTFS

Pliki systemu plików

Są dwa rodzaje plików: metadane i normalne. Metadane zawierają informacje o woluminie.

Struktura NTFS

Pliki systemu plików

$LogFile

- cykliczny bufor, w którym znajdują się zapisy transakcji

Struktura NTFS

Pliki systemu plików

$LogFile

- cykliczny bufor, w którym znajdują się zapisy transakcji
- na początku pliku znajdują się dwie kopie restart area

Struktura NTFS

Pliki systemu plików

$LogFile

- cykliczny bufor, w którym znajdują się zapisy transakcji
- na początku pliku znajdują się dwie kopie restart area
  - gdy wolumin jest odmontowany, powinny być identyczne

Struktura NTFS

Pliki systemu plików

$LogFile

- cykliczny bufor, w którym znajdują się zapisy transakcji
- na początku pliku znajdują się dwie kopie restart area
  - gdy wolumin jest odmontowany, powinny być identyczne
  - prawdopodobnie zawierają wskaźnik do pierwszej i ostatniej transakcji oraz do ostatniej wykonanej transakcji

Struktura NTFS

Pliki systemu plików

$LogFile

- cykliczny bufor, w którym znajdują się zapisy transakcji
- na początku pliku znajdują się dwie kopie restart area
  - gdy wolumin jest odmontowany, powinny być identyczne
  - prawdopodobnie zawierają wskaźnik do pierwszej i ostatniej transakcji oraz do ostatniej wykonanej transakcji
- przy pierwszym dostępie do dysku po awarii systemu, system czyta log i wykonuje rollback wszystkich operacji do ostatniej zakonczonej transakcji

Struktura NTFS

Pliki systemu plików

$Volume

Informacje o woluminie.

Struktura NTFS

Pliki systemu plików

$AttrDef

Zawiera definicje atrybutów, np. etykietę, typ, które z nich są indeksowane, które zawsze rezydentne, a które nie.

Struktura NTFS

Pliki systemu plików

(.)

Katalog główny. Jeśli wolumin ma reparse points, to (.) ma jako atrybut Named Data Stream $MountMgrDatabase.

Struktura NTFS

Pliki systemu plików

$Bitmap

Każdy bit reprezentuje jeden LCN.

Struktura NTFS

Pliki systemu plików

$Boot

Wskazuje na boot sector woluminu. Zawiera informacje o rozmiarze woluminu, klastrach i MFT. Jako jedyny, ten plik nie może być przeniesiony. Zawiera także LCN zerowego VCN $MFT i $MFTMirr. Plik ten zaczyna się w fizycznym klastrze 0. $Data tego pliku nigdy nie jest rezydentny.

Struktura NTFS

Pliki systemu plików

$BadClus

Zawiera listę wszystkich złych klastrów na woluminie. Jego strumień nienazwany jest pusty, ale strumień $Bad jest wielkości całego woluminu. Dobre klastry reprezentowane są jako luźne (sparse, zerowe) klastry. Złe klastry wskazują na te klastry na dysku. Przy dynamicznym remapowaniu złych klastrów, nowy zły klaster zostanie po cichu dodany do tego pliku.

Struktura NTFS

Pliki systemu plików

$Secure

Pole Security Id w $STANDARD_INFORMATION jest indeksem w $Secure. W pliku tym jest strumień $SDS (Security Descriptor Stream) i dwa indeksy: $SII (Security Id Index) i $SDH (Security Descriptor Hash) do odszukiwania.

Struktura NTFS

Pliki systemu plików

$UpCase

Zawiera wielkie litery (dla każdego znaku w Unicode). Używany do porównywania nazw plików niezależnie od strony kodowej.

Struktura NTFS

Pliki systemu plików

$Extend

Od Windows 2000. Katalog zawierający pliki $ObjId, $Quota, $Reparse, $UsnJrnl.

Struktura NTFS

Pliki systemu plików

$ObjId

Indeks atrybutów $OBJECT_ID.

Struktura NTFS

Pliki systemu plików

$Quota

Pojawiła się w Windows NT, ale nie była używana. Quoty są trzymane dla każdego użytkownika i dla każdego woluminu.

Struktura NTFS

Koncepcje i pojęcia

Klastry

- liczba sektorów tworzących klaster jest potęgą 2

Struktura NTFS

Koncepcje i pojęcia

Klastry

- liczba sektorów tworzących klaster jest potęgą 2
- Logical Cluster Number (LCN) - kolejny numer klastra na woluminie (od 0)

Struktura NTFS

Koncepcje i pojęcia

Klastry

- liczba sektorów tworzących klaster jest potęgą 2
- Logical Cluster Number (LCN) - kolejny numer klastra na woluminie (od 0)
- Virtual Cluster Number (VCN) - każdy klaster nierezydentnego strumienia dostaje kolenjny numer (od 0)

Struktura NTFS

Koncepcje i pojęcia

Klastry

- liczba sektorów tworzących klaster jest potęgą 2
- Logical Cluster Number (LCN) - kolejny numer klastra na woluminie (od 0)
- Virtual Cluster Number (VCN) - każdy klaster nierezydentnego strumienia dostaje kolenjny numer (od 0)
- Data Runs - każdy spójny blok LCN'ów dostaje Data Run, który zawiera VCN, LCN i długość

Struktura NTFS

Koncepcje i pojęcia

Klastry

- liczba sektorów tworzących klaster jest potęgą 2
- Logical Cluster Number (LCN) - kolejny numer klastra na woluminie (od 0)
- Virtual Cluster Number (VCN) - każdy klaster nierezydentnego strumienia dostaje kolenjny numer (od 0)
- Data Runs - każdy spójny blok LCN'ów dostaje Data Run, który zawiera VCN, LCN i długość

Wielkość klastra jest trzymana w $Boot. Także tam jest zdefiniowana wielkość MFT File Record (w klastrach) i Index Record. Dzięki klastrom można adresować większe dyski niż za pomocą sektorów.

Struktura NTFS

Koncepcje i pojęcia

Standardowy nagłówek atrybutu

- każdy atrybut każdego rekortu MFT ma standardowy nagłówek

Struktura NTFS

Koncepcje i pojęcia

Standardowy nagłówek atrybutu

- każdy atrybut każdego rekortu MFT ma standardowy nagłówek
- typ, rozmiar, nazwa atrybutu, oraz czy jest on rezydentny, indeksowany, luźny (sparse), zaszyfrowany czy skompresowany (gdy atrybut jest strumieniem)

Struktura NTFS

Koncepcje i pojęcia

Standardowy nagłówek atrybutu

- każdy atrybut każdego rekortu MFT ma standardowy nagłówek
- typ, rozmiar, nazwa atrybutu, oraz czy jest on rezydentny, indeksowany, luźny (sparse), zaszyfrowany czy skompresowany (gdy atrybut jest strumieniem)
- rozmiar zależny od tego, czy ma nazwę i od tego czy jest rezydentny.

Struktura NTFS

Koncepcje i pojęcia

Standardowy nagłówek atrybutu

- każdy atrybut każdego rekortu MFT ma standardowy nagłówek
- typ, rozmiar, nazwa atrybutu, oraz czy jest on rezydentny, indeksowany, luźny (sparse), zaszyfrowany czy skompresowany (gdy atrybut jest strumieniem)
- rozmiar zależny od tego, czy ma nazwę i od tego czy jest rezydentny.

Przykładowo domyślny, nienazwany strumień $Data jeśli nie jest rezydentny, jego nagłówek zawiera informację, gdzie znajduje się pierwszy i ostatni VCN.

Struktura NTFS

Koncepcje i pojęcia

Collation

Używane do sortowania i wyszukiwania obiektów w NTFS. Można porównywać nazwy plików, SID, unsigned longs...

Struktura NTFS

Koncepcje i pojęcia

B*drzewa

- zbalansowane

Struktura NTFS

Koncepcje i pojęcia

B*drzewa

- zbalansowane
- klucze drzewa w NTFS zawierają dane i wskaźnik do dzieci

Struktura NTFS

Koncepcje i pojęcia

B*drzewa

- zbalansowane
- klucze drzewa w NTFS zawierają dane i wskaźnik do dzieci
- nie ma maksymalnej liczby kluczy w węźle

Struktura NTFS

Koncepcje i pojęcia

B*drzewa

- zbalansowane
- klucze drzewa w NTFS zawierają dane i wskaźnik do dzieci
- nie ma maksymalnej liczby kluczy w węźle
- ma węzły co najmniej 2/3 pełne (oprócz korzenia)

Struktura NTFS

Koncepcje i pojęcia

B*drzewa

- zbalansowane
- klucze drzewa w NTFS zawierają dane i wskaźnik do dzieci
- nie ma maksymalnej liczby kluczy w węźle
- ma węzły co najmniej 2/3 pełne (oprócz korzenia)
- korzeń znajduje się w $MFT.

Struktura NTFS

Koncepcje i pojęcia

B*drzewa

- zbalansowane
- klucze drzewa w NTFS zawierają dane i wskaźnik do dzieci
- nie ma maksymalnej liczby kluczy w węźle
- ma węzły co najmniej 2/3 pełne (oprócz korzenia)
- korzeń znajduje się w $MFT.
- na nim zbudowane są wszystkie indeksy

Struktura NTFS

Koncepcje i pojęcia

Katalogi

- jest indeksem nazw plików (sam katalog, jak wszystko inne, jest plikiem)

Struktura NTFS

Koncepcje i pojęcia

Katalogi

- jest indeksem nazw plików (sam katalog, jak wszystko inne, jest plikiem)
- pozycje indeksu zawierają nazwę pliku, informacje standardowe i wskaźnik do informacji o zabezpieczeniach

Struktura NTFS

Koncepcje i pojęcia

Katalogi

- jest indeksem nazw plików (sam katalog, jak wszystko inne, jest plikiem)
- pozycje indeksu zawierają nazwę pliku, informacje standardowe i wskaźnik do informacji o zabezpieczeniach
- korzeń (zawsze rezydentny) zawiera kilka pozycji indeksu

Struktura NTFS

Koncepcje i pojęcia

Katalogi

- jest indeksem nazw plików (sam katalog, jak wszystko inne, jest plikiem)
- pozycje indeksu zawierają nazwę pliku, informacje standardowe i wskaźnik do informacji o zabezpieczeniach
- korzeń (zawsze rezydentny) zawiera kilka pozycji indeksu
- $INDEX_ALLOCATION to zestaw przebiegów mówiący gdzie szukać dalszych części indeksu

Struktura NTFS

Koncepcje i pojęcia

Katalogi

- jest indeksem nazw plików (sam katalog, jak wszystko inne, jest plikiem)
- pozycje indeksu zawierają nazwę pliku, informacje standardowe i wskaźnik do informacji o zabezpieczeniach
- korzeń (zawsze rezydentny) zawiera kilka pozycji indeksu
- $INDEX_ALLOCATION to zestaw przebiegów mówiący gdzie szukać dalszych części indeksu
- jest także bitmapa mówiąca, które klastry z indeksami są używane

Struktura NTFS

Koncepcje i pojęcia

Katalogi

Indeks jest trzymany w wierzchołkach B*drzewa:

- każdy wierzchołek zawiera jedną lub kilka pozycji indeksu; pozycje posortowane są rosnąco

Struktura NTFS

Koncepcje i pojęcia

Katalogi

Indeks jest trzymany w wierzchołkach B*drzewa:

- każdy wierzchołek zawiera jedną lub kilka pozycji indeksu; pozycje posortowane są rosnąco
- każda pozycja indeksu może wskazywać na podwierzchołek zawierający tylko mniejsze pozycje

Struktura NTFS

Koncepcje i pojęcia

Katalogi

Indeks jest trzymany w wierzchołkach B*drzewa:

- każdy wierzchołek zawiera jedną lub kilka pozycji indeksu; pozycje posortowane są rosnąco
- każda pozycja indeksu może wskazywać na podwierzchołek zawierający tylko mniejsze pozycje
- korzeń jest w $INDEX_ROOT

Struktura NTFS

Koncepcje i pojęcia

FILE records

- to takie inodes
- nagłówek - atrybuty - 0xFFFFFFFF
- LSN ($LogFile Sequence Number)

Struktura NTFS

Koncepcje i pojęcia

Fixups

Aby móc wykrywać błędy w klastrach, sektory mają Fixups trzymane w Update Sequence Array, gdzie są skopiowane 2 ostatnie bajty każdego sektora w klastrze. Na ich miejsce jest kopiowany Update Sequence Number.

Struktura NTFS

Koncepcje i pojęcia

Data Runs

- nierezydentne atrybuty są trzymane w ciągach klastrów (przebiegach, runs)

Struktura NTFS

Koncepcje i pojęcia

Data Runs

- nierezydentne atrybuty są trzymane w ciągach klastrów (przebiegach, runs)
- każdy przebieg jest reprezentowany przez klaster początkowy i długość

Struktura NTFS

Koncepcje i pojęcia

Data Runs

- nierezydentne atrybuty są trzymane w ciągach klastrów (przebiegach, runs)
- każdy przebieg jest reprezentowany przez klaster początkowy i długość
- startowy klaster kolejnego przebiegu jest kodowany jako ofset do początkowego klastra poprzedniego przebiegu

Struktura NTFS

Koncepcje i pojęcia

Data Runs

- nierezydentne atrybuty są trzymane w ciągach klastrów (przebiegach, runs)
- każdy przebieg jest reprezentowany przez klaster początkowy i długość
- startowy klaster kolejnego przebiegu jest kodowany jako ofset do początkowego klastra poprzedniego przebiegu
- normalne, skompresowane i rzadkie pliki są definiowane przez przebiegi

Struktura NTFS

Koncepcje i pojęcia

Data Runs

- nierezydentne atrybuty są trzymane w ciągach klastrów (przebiegach, runs)
- każdy przebieg jest reprezentowany przez klaster początkowy i długość
- startowy klaster kolejnego przebiegu jest kodowany jako ofset do początkowego klastra poprzedniego przebiegu
- normalne, skompresowane i rzadkie pliki są definiowane przez przebiegi
- Runlist to sekwencja elementów - każdy element trzyma ofset do początkowego LCN poprzedniego elementu i długości przebiegu w klastrach a na początku - długości tych liczb

Data Runs

Przykład 1 - zwykły, niepofragmentowany plik

Przebiegi: 21 18 34 56 00

1. 21 18 34 56
   - rozmiar długości przebiegu - 1 (little endian)
   - rozmiar offsetu - 2
   - długość przebiegu - 0x18
   - offset - 0x5634
2. 00 - koniec

Data Runs

Przykład 1 - zwykły, niepofragmentowany plik

Przebiegi: 21 18 34 56 00

1. 21 18 34 56
   - rozmiar długości przebiegu - 1 (little endian)
   - rozmiar offsetu - 2
   - długość przebiegu - 0x18
   - offset - 0x5634
2. 00 - koniec

Czyli 0x18 klastrów na LCN 0x5634

Data Runs

Przykład 2 - pofragmentowany, zwykły plik

Przebiegi: 31 38 73 25 34 32 14 01 E5 11 02 31 42 AA 00 03 00

1. 31 38 73 25 34
   - długość - 0x38 (1B)
   - ofset - 0x342573
2. 32 14 01 E5 11 02
   - długość - 0x114
   - ofset - 0x363758 (0x211E5 w stosunku do 0x342573)
3. ...

Data Runs

Przykład 3 - rzadki, niepofragmentowany plik

Przebiegi: 11 30 20 01 60 11 10 30 00

1. 11 30 20 - 0x30 @ 0x20

Data Runs

Przykład 3 - rzadki, niepofragmentowany plik

Przebiegi: 11 30 20 01 60 11 10 30 00

1. 11 30 20 - 0x30 @ 0x20
2. 01 60 - 0x60 @ nigdzie (zera)

Data Runs

Przykład 3 - rzadki, niepofragmentowany plik

Przebiegi: 11 30 20 01 60 11 10 30 00

1. 11 30 20 - 0x30 @ 0x20
2. 01 60 - 0x60 @ nigdzie (zera)
3. 11 10 30 - 0xx10 @ 0x20 + 0x30 = 50

Data Runs

Przykład 3 - rzadki, niepofragmentowany plik

Przebiegi: 11 30 20 01 60 11 10 30 00

1. 11 30 20 - 0x30 @ 0x20
2. 01 60 - 0x60 @ nigdzie (zera)
3. 11 10 30 - 0xx10 @ 0x20 + 0x30 = 50
4. 00 - koniec

- nazwa kodowa nowego systemu plików Windows (Windows Future System)

- nazwa kodowa nowego systemu plików Windows (Windows Future System)
- relacyjna baza danych; połączenie tradycyjnych relacyjnych baz danych, obiektów, XML i systemu plików

- nazwa kodowa nowego systemu plików Windows (Windows Future System)
- relacyjna baza danych; połączenie tradycyjnych relacyjnych baz danych, obiektów, XML i systemu plików
- reprezentacja pliku nie poprzez ścieżkę, ale poprzez jego właściwości

- nazwa kodowa nowego systemu plików Windows (Windows Future System)
- relacyjna baza danych; połączenie tradycyjnych relacyjnych baz danych, obiektów, XML i systemu plików
- reprezentacja pliku nie poprzez ścieżkę, ale poprzez jego właściwości
- spłaszczenie hierarchii, wyszukiwanie pliku poprzez jego atrybuty

- nazwa kodowa nowego systemu plików Windows (Windows Future System)
- relacyjna baza danych; połączenie tradycyjnych relacyjnych baz danych, obiektów, XML i systemu plików
- reprezentacja pliku nie poprzez ścieżkę, ale poprzez jego właściwości
- spłaszczenie hierarchii, wyszukiwanie pliku poprzez jego atrybuty
- na bazie engine'u Microsoft SQL Server

- nazwa kodowa nowego systemu plików Windows (Windows Future System)
- relacyjna baza danych; połączenie tradycyjnych relacyjnych baz danych, obiektów, XML i systemu plików
- reprezentacja pliku nie poprzez ścieżkę, ale poprzez jego właściwości
- spłaszczenie hierarchii, wyszukiwanie pliku poprzez jego atrybuty
- na bazie engine'u Microsoft SQL Server
- http://blogs.msdn.com/winfs/

- nazwa kodowa nowego systemu plików Windows (Windows Future System)
- relacyjna baza danych; połączenie tradycyjnych relacyjnych baz danych, obiektów, XML i systemu plików
- reprezentacja pliku nie poprzez ścieżkę, ale poprzez jego właściwości
- spłaszczenie hierarchii, wyszukiwanie pliku poprzez jego atrybuty
- na bazie engine'u Microsoft SQL Server
- http://blogs.msdn.com/winfs/
- WinFS Beta 1 została udostępniona w 2005 roku dla Windows XP SP2

- nazwa kodowa nowego systemu plików Windows (Windows Future System)
- relacyjna baza danych; połączenie tradycyjnych relacyjnych baz danych, obiektów, XML i systemu plików
- reprezentacja pliku nie poprzez ścieżkę, ale poprzez jego właściwości
- spłaszczenie hierarchii, wyszukiwanie pliku poprzez jego atrybuty
- na bazie engine'u Microsoft SQL Server
- http://blogs.msdn.com/winfs/
- WinFS Beta 1 została udostępniona w 2005 roku dla Windows XP SP2
- nie wiadomo, kiedy WinFS zostanie włączony do systemów Microsoft Windows

WinFS Type Extensibility

Ma pozwalać deweloperom na rozszerzanie WinFS za pomocą schematów definiujących nowe typy danych i asocjacje. Są to pliki XML (ale nie w rozumieniu W3C). Schematy te mogą być widoczne tylko dla aplikacji, z którą zostały zainstalowane lub być udostępnione na cały system.

Installable File System

File System Driver (FSD)

Sterowniki te zajmują się w Windows formatami systemów plików. Działają w trybie jądra. Używają zestawu funkcji Ntoskrnl. Do tworzenia sterowników potrzebny jest Windows Installable File System (IFS).

Źródła

- http://e2fsprogs.sourceforge.net
- http://www.namesys.com
- Linuxmafia.com Knowledgebase
- Wikipedia
- http://ntfs.com/
- linux ntfs project
- Mark E. Russinovich, David A. Solomon Microsoft Windows Internals