Informatyka 2 (ES1D300 017)

Politechnika Białostocka - Wydział Elektryczny

Elektrotechnika, semestr III, studia stacjonarne I stopnia

Rok akademicki 2019/2020

Wykład nr 12 (14.01.2020)

dr inż. Jarosław Forenc

Plan wykładu nr 12

- Systemy plików
  - NTFS
  - ext2
- Zarządzanie pamięcią operacyjną
  - partycjonowanie statyczne i dynamiczne
  - proste stronicowanie

NTFS (New Technology File System)

- wersja 1.0 (połowa 1993 r.) - Windows NT 3.1
- wersja 1.1 (jesień 1994 r.) - Windows NT 3.5
- wersja 1.2 (NTFS 4) - Windows NT 3.51 (1995 r.)
- wersja 3.0 (NTFS 5) - Windows 2000
- wersja 3.1 (NTFS 5.1) - Windows XP/Server 2003/Vista/7/8/10
- teoretyczny rozmiar partycji NTFS wynosi 2^{64}-1 klastrów, ale Windows potrafi obsłużyć tylko 2^{32}-1 klastrów (dla klastra 64 kB - ok. 256 TB)
- tabela partycji w MBR dysku twardego ogranicza rozmiar partycji do 2 TB
- teoretyczna wielkość pliku wynosi 2^{64} bajtów minus 1 kB, ale Windows ogranicza ten rozmiar do 2^{44} bajtów minus 64 kB (ok. 16 TB)

NTFS

- struktura wolumenu (dysku) NTFS:

- Boot Sector rozpoczyna się od zerowego sektora partycji, może zajmować 16 kolejnych sektorów, zawiera podobne dane jak w systemie FAT
- MFT (Master File Table) - specjalny plik, niewidoczny dla użytkownika, zawiera wszystkie dane niezbędne do odczytania pliku z dysku, składa się z rekordów o stałej długości (1 kB - 4 kB)
- pierwsze 16 (NTFS 4) lub 26 (NTFS 5) rekordów jest zarezerwowane dla tzw. metaplików, np.
  - rekord nr: 0 plik: $Mft (główna tablica plików)
  - rekord nr: 1 plik: $MftMirr (główna tablica plików 2)
  - rekord nr: 5 plik: $ (indeks katalogu głównego)

NTFS

- struktura wolumenu (dysku) NTFS:

- pozostała część pliku MFT przeznaczona jest na rekordy wszystkich plików i katalogów umieszczonych na dysku
- jeśli pierwszy rekord MFT jest uszkodzony to system automatycznie odczytuje drugi rekord, w którym zapisana jest kopia pierwszego
- położenie obu metaplików $Mft i $MftMirr zapisane jest w sektorze startowym partycji

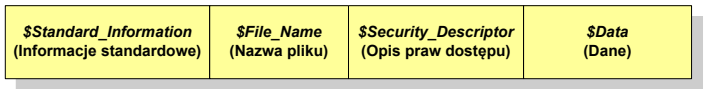
NTFS

- struktura wolumenu (dysku) NTFS:

- plik w NTFS to zbiór atrybutów
- wszystkie atrybuty mają dwie części składowe: nagłówek i blok danych
- nagłówek opisuje atrybut, np. liczbę bajtów zajmowanych przez atrybut, rozmiar bloku danych, położenie bloku danych, znacznik czasu
- bloku danych zawiera informacje zgodne z przeznaczeniem atrybutu

NTFS - Pliki

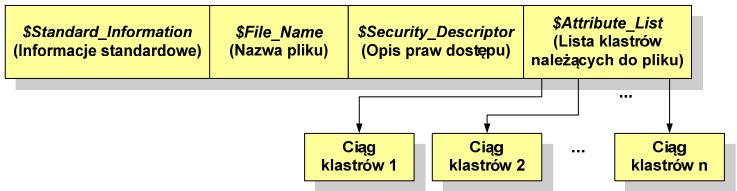
- pliki w systemie NTFS są reprezentowane w MFT przez rekord zawierający atrybuty:
  - $Standard_Information
  - $File_Name
  - $Security_Descriptor
  - $Data



- w przypadku małych plików wszystkie jego atrybuty zapisywane są bezpośrednio w MFT (atrybuty rezydentne)

NTFS - Pliki

- jeśli atrybuty pliku są duże (najczęściej dotyczy to atrybutu $Data), to w rekordzie w MFT umieszczany jest tylko nagłówek atrybutu oraz wskaźnik do jego bloku danych, a sam blok danych przenoszony jest na dysk poza MFT (atrybuty nierezydentne)
- blok danych atrybutu nierezydentnego zapisywany jest w przyległych klastrach
- jeśli nie jest to możliwe, to dane zapisywane są w kilku ciągach jednostek alokacji i wtedy każdemu ciągowi odpowiada wskaźnik w rekordzie MFT



NTFS - Katalogi

- katalogi reprezentowane są przez rekordy zawierające trzy takie same atrybuty jak pliki:
  - $Standard_Information
  - $File_Name
  - $Security_Descriptor

$Standard_Information (Informacje standardowe)	*$File_Name* (Nazwa pliku)	*$Security_Descriptor* (Opis praw dostępu)	*$Index_Root*	*$Index_Allocation*	*$Bitmap*

- zamiast atrybutu $Data umieszczone są trzy atrybuty przeznaczone do tworzenia list, sortowania oraz lokalizowania plików i podkatalogów
  - $Index_Root
  - $Index_Allocation
  - $Bitmap

ext2

- pierwszy system plików w Linuxie: Minix (14-znakowe nazwy plików i maksymalny rozmiar wynoszący 64 MB)
- system Minix zastąpiono nowym systemem nazwanym rozszerzonym systemem plików - ext (ang. extended file system), a ten, w styczniu 1993 r., systemem ext2 (ang. second extended file system)
- w systemie ext2 podstawowym elementem podziału dysku jest blok
- wielkość bloku jest stała w ramach całego systemu plików, określana na etapie jego tworzenia i może wynosić 1024, 2048 lub 4096 bajtów
- w celu zwiększenia bezpieczeństwa i optymalizacji zapisu na dysku posługujemy się nie pojedynczymi blokami, a grupami bloków

Boot Sector	Bloki grupy 1	Bloki grupy 2	...	Bloki grupy N

ext2

- Boot Sector (blok startowy) przechowuje informacje wykorzystywane przez system operacyjny podczas jego uruchamiania

Boot Sector	Bloki grupy 1	Bloki grupy 2	...	Bloki grupy N

- na poziomie logicznym grupę bloków tworzą:
  - deskryptor grupy (32 bajty)
  - blok z mapą zajętości bloków danych (1 blok dyskowy)
  - blok z mapą zajętości i-węzłów (1 blok dyskowy)
  - bloki z tablicą i-węzłów
  - bloki danych

Deskryptor grupy	Blok z mapą bitową zajętości bloków danej grupy	Blok z mapą bitową zajętości i-węzłów danej grupy	Tablica i-węzłów	Bloki danych

ext2

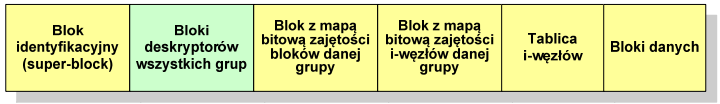
- każda grupa fizyczna bloków zawiera informacje o jednej grupie logicznej, a ponadto pewne informacje o całym systemie plików

Blok identyfikacyjny (super-block)	Bloki deskryptorów wszystkich grup	Blok z mapą bitową zajętości bloków danej grupy	Blok z mapą bitową zajętości i-węzłów danej grupy	Tablica i-węzłów	Bloki danych

- w każdej grupie fizycznej bloków znajduje się kopia tego samego bloku identyfikacyjnego oraz kopia bloków z deskryptorami wszystkich grup
- blok identyfikacyjny zawiera informacje na temat systemu plików:
  - numer urządzenia, na którym jest super-block
  - rodzaj systemu plików
  - rozmiar bloku
  - struktury do synchronizacji dostępu
  - czas dokonanej ostatnio zmiany
  - informacje specyficzne dla konkretnej implementacji

ext2

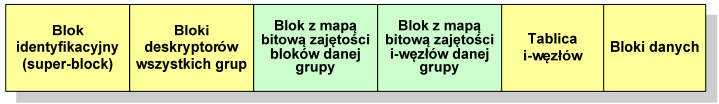
- każda grupa fizyczna bloków zawiera informacje o jednej grupie logicznej, a ponadto pewne informacje o całym systemie plików



- w deskryptorach grupy znajdują się informacje na temat grupy bloków:
  - numer bloku z bitmapą zajętości bloków grupy
  - numer bloku z bitmapą zajętości i-węzłów
  - numer pierwszego bloku z tablicą i-węzłów
  - liczba wolnych bloków
  - liczba wolnych i-węzłów w grupie
  - liczba katalogów w grupie

ext2

- każda grupa fizyczna bloków zawiera informacje o jednej grupie logicznej, a ponadto pewne informacje o całym systemie plików



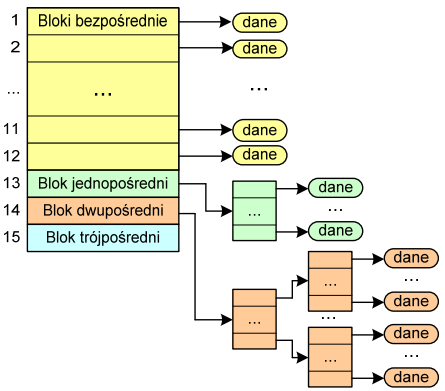
- blok z mapą bitową zajętości bloków danej grupy jest tablicą bitów o rozmiarze jednego bloku
  - jeśli blok ma rozmiar 1 kB to pojedynczą mapą można opisać fizyczna grupę 8096 bloków czyli 8 MB danych
  - jeśli natomiast blok ma rozmiar 4 kB, to fizyczna grupa bloków zajmuje 128 MB danych
- przed tablicą i-węzłów znajduje się blok z mapa bitową zajętości i-węzłów danej grupy - jest to tablica bitów, z których każdy zawiera informację czy dany i-węzeł jest wolny czy zajęty

ext2 - i-węzeł

- pliki na dysku reprezentowane są przez i-węzły (ang. i-node)
- każdemu plikowi odpowiada dokładnie jeden i-węzeł, który jest strukturą zawierającą m.in. następujące pola:
  - numer i-węzła w dyskowej tablicy i-węzłów
  - typ pliku: zwykły, katalog, łącze nazwane, specjalny, znakowy
  - prawa dostępu do pliku: dla wszystkich, grupy, użytkownika
  - liczba dowiązań do pliku
  - identyfikator właściciela pliku
  - identyfikator grupy właściciela pliku
  - rozmiar pliku w bajtach (max. 4 GB)
  - czas utworzenia pliku
  - czas ostatniego dostępu do pliku
  - czas ostatniej modyfikacji pliku
  - liczba bloków dyskowych zajmowanych przez plik

ext2 - i-węzeł

- położenie pliku na dysku określają w i-węźle pola:
  - 12 adresów bloków zawierających dane (w systemie Unix jest ich 10) - bloki bezpośrednie
  - 1 adres bloku zawierającego adresy bloków zawierających dane - blok jednopośredni (ang. single indirect block)
  - 1 adres bloku zawierającego adresy bloków jednopośrednich - blok dwupośredni (ang. double indirect block)
  - 1 adres bloku zawierającego adresy bloków dwupośrednich - blok trójpośredni (ang. triple indirect block)



ext2

- nazwy plików przechowywane są w katalogach, które w systemie Linux są plikami, ale o specjalnej strukturze
- katalogi składają się z ciągu tzw. pozycji katalogowych o nieustalonej z góry długości
- każda pozycja opisuje dowiązanie do jednego pliku i zawiera:
  - numer i-węzła (4 bajty)
  - rozmiar pozycji katalogowej (2 bajty)
  - długość nazwy (2 bajty)
  - nazwa (od 1 do 255 znaków)

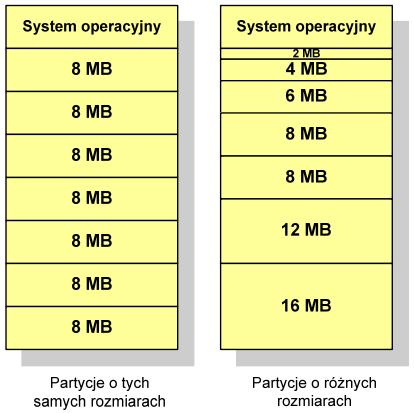
```
struct ext2_dir_entry
{
   _u32  inode                 /*  numer i-wezla               */
   _u16  rec_len               /*  dlugosc pozycji katalogowej */
   _u16  name_len              /*  dlugosc nazwy               */
   char  name[EXT2_NAME_LEN]   /*  nazwa                       */
}
```


Zarządzanie pamięcią

- zarządzanie pamięcią polega na wydajnym przenoszeniu programów i danych do i z pamięci operacyjnej
- w nowoczesnych wieloprogramowych systemach operacyjnych zarządzanie pamięcią opiera się na pamięci wirtualnej
- pamięć wirtualna bazuje na wykorzystaniu segmentacji i stronicowania
- z historycznego punktu widzenia w systemach komputerowych stosowane były/są następujące metody zarządzania pamięcią:
  - partycjonowanie statyczne, partycjonowanie dynamiczne
  - proste stronicowanie, prosta segmentacja
  - stronicowanie pamięci wirtualnej, segmentacja pamięci wirtualnej
  - stronicowanie i segmentacja pamięci wirtualnej

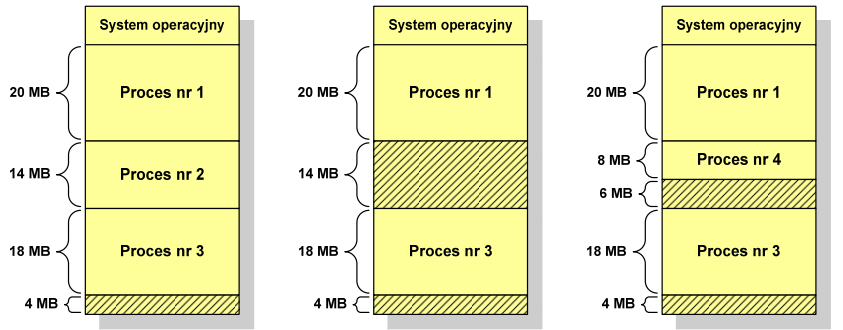
Partycjonowanie statyczne

- podział pamięci operacyjnej na obszary o takim samym lub różnym rozmiarze, ustalanym podczas generowania systemu



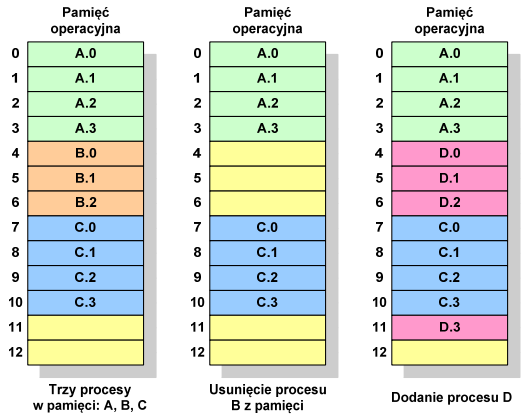
Partycjonowanie dynamiczne

- partycje są tworzone dynamicznie w ten sposób, że każdy proces jest ładowany do partycji o rozmiarze równym rozmiarowi procesu
- partycje mają różną długość, może zmieniać się także ich liczba
- przykład - w systemie działa 5 procesów: 20 MB, 14 MB, 18 MB, 8 MB, 8 MB



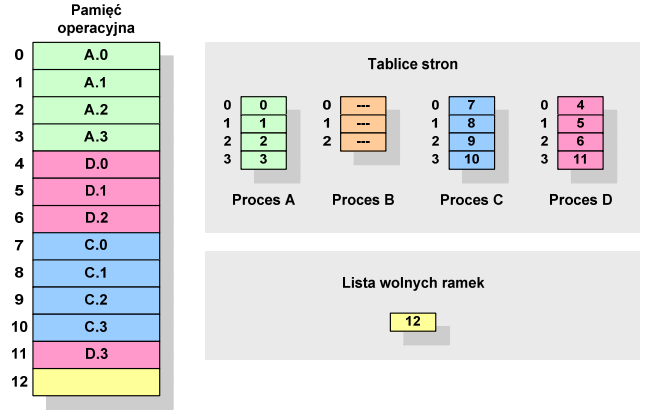
Proste stronicowanie

- pamięć operacyjna podzielona jest na jednakowe bloki o stałym niewielkim rozmiarze nazywane ramkami lub ramkami stron (page frames)
- do tych ramek wstawiane są fragmenty procesu zwane stronami (pages)
- aby proces mógł zostać uruchomiony wszystkie jego strony muszą znajdować się w pamięci operacyjnej



Proste stronicowanie

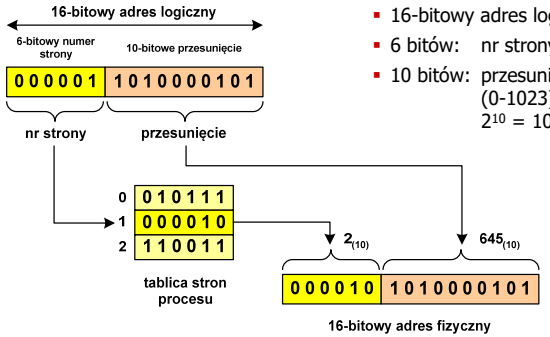
- dla każdego procesu przechowywana jest tablica strony (page table) zawierająca lokalizację ramki dla każdej strony procesu



Proste stronicowanie

- aby mechanizm stronicowania był wygodny ustala się, że rozmiar strony jest liczbą podniesioną do potęgi drugiej - dzięki temu adres względny oraz adres logiczny (numer strony + jej przesunięcie) są takie same

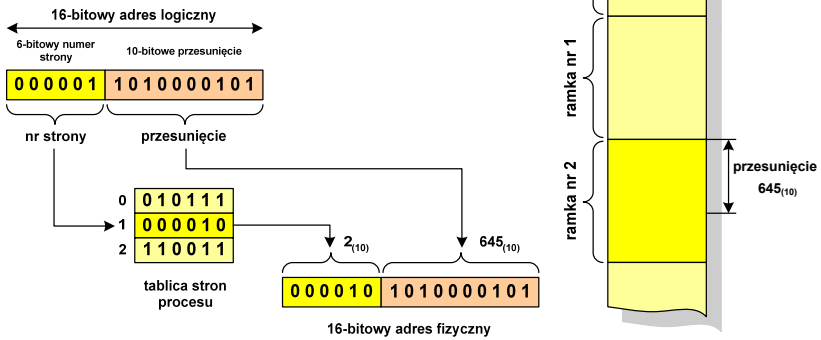
Przykład:



- 16-bitowy adres logiczny
- 6 bitów: nr strony (0-63), max. 2^6 = 64 strony
- 10 bitów: przesunięcie w ramach strony (0-1023), rozmiar strony wynosi: 2^{10} = 1024 bajty = 1 kB

Proste stronicowanie

- zalety: brak fragmentacji zewnętrznej, stronicowanie nie jest widoczne dla programisty
- wady: niewielki stopień fragmentacji wewnętrznej



Koniec wykładu nr 12

Dziękuję za uwagę!