

# Systemy operacyjne

## Lista zadań nr 11

Na zajęcia 4 stycznia 2024

Przed przystąpieniem do rozwiązywania zadań zapoznaj się z dokumentem: [The Second Extended File System Internal Layout](https://www.nongnu.org/ext2-doc/ext2.html)<sup>1</sup>. W zadaniach 2–6 rozważamy pierwszą korektę ext2 (ang. *revision 1*).

**UWAGA!** W trakcie prezentacji należy być gotowym do zdefiniowania pojęć oznaczonych **wytłuszczoną** czcionką.

**Zadanie 1.** Wyjaśnij czym są **punkty montażowe**, a następnie wyświetl listę zamontowanych systemów plików i wyjaśnij co znajduje się w poszczególnych kolumnach wydruku. Które z punktów montażowych dają dostęp do instancji **pseudo systemów plików**? Na podstawie **mount(8)** wyjaśnij znaczenie następujących atrybutów punktów montażowych: «relatime», «noexec» i «nodev», a następnie podaj scenariusz, w którym ich zastosowanie jest pożądane.

**Wskazówka:** Rozważ semantykę wymienionych atrybutów w kontekście systemu plików na przenośnym dysku USB.

**Zadanie 2.** Korzystając z pól **superbloku** (ang. *superblock*) podaj wzór na wyliczenie wartości: rozmiaru **bloku**, liczby i-węzłów i bloków przechowywanych w **grupie bloków** (ang. *block group*), liczby wpisów **tablicy deskryptorów grup bloków** (ang. *block group descriptor table*). Wymień składowe należące do grupy bloków oraz podaj ich rozmiar w blokach. Które grupy bloków przechowują kopie zapasową superbloku i tablicy deskryptorów grup bloków?

**Zadanie 3.** Podstawowymi operacjami na systemie plików są: wyzeruj lub zapal bit w bitmapie i-węzłów albo bloków, wczytaj / zapisz i-węzeł albo **blok pośredni** (ang. *indirect block*) albo blok danych. Podaj listę kroków niezbędnych do realizacji funkcji dopisującej  $n$  bloków na koniec pliku. Zakładamy, że poszczególne kroki funkcji są zawsze wdrażane **synchronicznie**. Zadbaj o to by funkcje nie naruszyły **spójności systemu plików** w przypadku awarii zasilania. Dopuszczamy powstawanie wycieków pamięci.

**Zadanie 4.** Przy pomocy wywołania systemowego **rename(2)** można przenieść **atomowo** plik do katalogu znajdującego się w obrębie tego samego systemu plików. Czemu «rename» zakończy się błędem «EXDEV» kiedy próbujemy przenieść plik do innego systemu plików? Powtórz polecenia z zadania 3 dla funkcji przenoszącej plik między dwoma różnymi katalogami w obrębie tego samego systemu plików. Zakładamy, że w katalogu docelowym jest wystarczająco dużo miejsca na dodanie wpisu. Pamiętaj, że wpis katalogu nie może przecinać granicy między blokami!

**Zadanie 5.** Przy pomocy wywołania systemowego **unlink(2)** można usunąć plik niebędący katalogiem. Powtórz polecenia z zadania 3 dla funkcji usuwającej plik zwykły z katalogu. Kiedy możliwe jest odwrócenie operacji usunięcia pliku tj. odkasowania (ang. *undelete*)? Zauważ, że usunięcie pliku nie odbiera procesom możliwości czytania jego zawartości, o ile go otworzyły przed wywołaniem **unlink(2)**. Kiedy w takim razie plik zostanie faktycznie usunięty z dysku?

**Zadanie 6.** Wyjaśnij co robi system plików ext2 przy tworzeniu **dowiązania twardego** (ang. *hard link*) i **symbolicznego** (ang. *symbolic link*). Gdzie jest przechowywana zawartość dowiązania symbolicznego? Jak za pomocą dowiązania symbolicznego stworzyć w systemie plików pętlę? Kiedy jądro systemu operacyjnego ją wykryje i zwróci błąd «ELOOP»? Czemu pętli nie da się zrobić z użyciem dowiązania twardego?

**Zadanie 7.** Czemu **fragmentacja systemu plików** jest szkodliwym zjawiskiem? Zreferuj artykuł [The new ext4 filesystem: current status and future plans](https://www.kernel.org/doc/ols/2007/ols2007v2-pages-21-34.pdf)<sup>2</sup>. Opisz w jaki sposób **odroczone przydziały bloków** (ang. *delayed allocation*) [§3.2] zapobiega powstawaniu fragmentacji. Wytłumacz jak **zakresy** (ang. *extents*) [§2.2] pomagają w ograniczaniu rozmiaru metadanych przechowujących adresy bloków należących do danego pliku. Czy po defragmentacji systemu plików ext4 liczba wolnych bloków może wzrosnąć? Jak mógłby wyglądać najprostszy algorytm defragmentacji [§3.3]?

<sup>1</sup><https://www.nongnu.org/ext2-doc/ext2.html>

<sup>2</sup><https://www.kernel.org/doc/ols/2007/ols2007v2-pages-21-34.pdf>

**Zadanie 8.** Przy użyciu programu `debugfs(8)` dla wybranej instancji systemu plików ext4 (np. **partycja** przechowująca główny system plików Twojej instalacji systemu Linux) pokaż:

- fragmentację systemu plików (`freefrag`) i informacje o grupach bloków (`stats`),
- zakresy bloków z których składa się wybrany duży plik (`extents`),
- że dowiązanie symboliczne może być przechowywane w i-węźle (`idump`),
- do jakiego pliku należy wybrany blok (`blocks`, `icheck`, `ncheck`),
- reprezentację liniową małego katalogu (`bdump`).

**Ostrzeżenie!** Narzędzie `debugfs` działa domyślnie w trybie tylko do odczytu, więc możesz go bezpiecznie używać na swoim komputerze. Trybu do odczytu i zapisu używasz na własną odpowiedzialność!

**Zadanie 9 (bonus).** Na podstawie §3 artykułu [A Directory Index for Ext2<sup>3</sup>](#) opisz strukturę danych HTree i operację wyszukiwania wpisu katalogu o zadanej nazwie. Następnie wyświetl reprezentację HTree dużego katalogu, np. `/var/lib/dpkg/info`, używając polecenia `htree` programu `debugfs(8)`.

**Zadanie 10 (bonus).** Czym różni się **księgowanie metadanych** od **księgowania danych**? Na podstawie wydruku polecenia `logdump` programu `debugfs` i opisu [struktur dyskowych ext4<sup>4</sup>](#) opisz format **dziennika**. Opisz znaczenie poszczególnych bloków z których może składać się pojedyncza **transakcja**. Czemu operacje składowane w dzienniku muszą być **idempotentne**?

---

<sup>3</sup><https://www.kernel.org/doc/ols/2002/ols2002-pages-425-438.pdf>

<sup>4</sup>[https://ext4.wiki.kernel.org/index.php/Ext4\\_Disk\\_Layout](https://ext4.wiki.kernel.org/index.php/Ext4_Disk_Layout)