Systemy plików ext3 i XFS

Andrzej Awramiuk

9. stycznia 2004

1 Wstęp

Poniższa prezentacja jest częścią prezentacji porównującej Unixowe systemy plików, przygotowywanej przez Andrzeja Awramiuka, Marię Fronczak i Mateusza Zakrzewskiego. Obejmuje zagadnienia dotyczące lokalnych systemów plików ext3 oraz XFS oraz porównania wydajności niektórych unixowych systemów plików.

2 Kronikowanie

Kronikowanie to technika zapisywania dokonywanych w plikach zmian zamiast bezpośredniego dokonywania konkretnych zmian. Aby zwiększyć funkcjonalność i bezpieczeństwo operacje zapisu na dysku często grupuje się w niepodzielne grupy (transakcje), które stanowią logiczną całość. Takie podejście stosowane w Systemach Zarządzania Bazami Danych pojawiło się w nowoczesnych systemach plików nazywanych systemami plików z kronikowaniem (ang. Journalised File Systems). Struktura danych, która przechowuje opis operacji składających się na jedną całość nazywa się dziennikiem lub kroniką (ang. Journal). Dane w niej zapisywane są nazywane wpisami lub logami. Zazwyczaj scenariusz zapisu danych (lub metadanych) na dysku wygląda następująco:

- umieszczamy wpis w dzienniku co zmienimy i w jaki sposób tego dokonamy (aby niezakończone zmiany można było wycofać i przywrócić stan po awarii).
- zapisujemy zmiany na dysku
- opcjonalnie usuwamy wpis z dziennika

Księgowanie przydaje się w przypadku nagłej awarii zasilania, ponieważ stan danych zapisanych na dysku jest monitorowany, a uruchomienie systemu plików po awarii jest niemal natychmiastowe.

3 System plików ext3

3.1 Główne cechy

Jak wynika z nazwy nowego systemu plików, ext3 jest on nastepca popularnego ext2. Został stworzony przez Stephen'a Tweedie z firmy RedHat w 1999 roku (wersja alfa). Pojawił się w oficjalnym jądrze 2.4.15 choć Red Hat 7.2 ukazał sie znacznie wcześniej i ext3 był tam domyślnym systemem plików. Najważniejszą wprowadzoną zmianą jest to, że jest to system plików z księgowaniem, czyli opisanym wyżej "journallingiem". Główne różnice w stosunku do systemu ext2:

- dziennik, który skraca czas wstawania systemu po awarii, oraz zmniejsza niebezpieczeństwo utraty danych.
- indeksowane katalogi, zdecydowanie zwiększające szybkość dostępu do plików.
- fragmentacja bloków, oszczędzająca zajmowany obszar.
- lista kontroli dostępu
- obsługa skompresowanych i zaszyfrowanych plików(opcjonalnie)
- logiczne usuwanie(undelete)

Celem autora systemu ext3 było usprawnienie ext2, a nie stworzenie zupełnie nowego systemu plików. Dało to następujące korzyści:

- możliwość montowania systemu ext3 jako ext2 (jeśli system został poprawnie odmontowany)
- możliwość korzystania z wielu sprawdzonych programów narzędziowych (m.in. fsck)
- możliwość łatwej i bezpiecznej migracji z ext2 przy pomocy programu tune(nie trzeba nawet odmontowywać systemu ext2 przy przejścia do ext3 i kopiować danych)
- przewidywalność, stabilność i uniknięcie wielu błędów
- zaufanie użytkowników wynikające z wiarygodności ext2

3.2 Kronikowanie w ext3

Dziennikowanie w ext3 jest zaimplementowane pośrednio. Wykorzystane jest do tego API - warstwa Journaling Block Device (JBD). JBD zostało tak stworzone aby w prosty sposób dołączać dziennikowanie do dowolnego urządzenia blokowego. Interfejs JBD zachowuje w journalu całe fizyczne bloki zamiast pojedynczych bajtów. Rozwiązanie to zwiększa rozmiar dziennika a także ilość zapisywanych danych, było prostsze do zaimplementowania i wymaga mniej czasu procesora niż zapisywanie tylko zmienionych fragmentów bloków. JBD oznacza

transakcję za zakończoną poprzez zapisanie w dzienniku bloku zawierającego specjalne sekwencje - korzysta przy tym z założenia, że dysk zawsze zakończy zapisywanie pojedynczego sektora (używając zapasu energii). Niczym nie grozi ponowna awaria sprzętu w trakcie aplikowania zmian z dziennika - przy ponownym uruchamianiu bloki zostaną nagrane ponownie. Dziennik jest oznaczany jako pusty dopiero po zakończeniu całego odtwarzania. System stara się optymalizować zużycie pamięci poprzez "ściskanie"(squish) bloków i sprawdza, czy nie przechowuje już bloku w związku z poprzednią modyfikacją.

W ext3 dziennik tworzony i zarządzany przez JBD trzymany jest w i-node, czyli po prostu w pliku (domyślnie jest to plik .journal w katalogu głównym).

System plików ext3 dostarcza trzy różne poziomy dziennikowania:

- journal - dziennikowane są wszystkie zmiany danych jak i metadanych
- ordered - dziennikowane są wszystkie zmiany metadanych, zmiany na metadanych odbywają się dopiero po zapisaniu danych na dysku
- writeback - dziennikowane są tylko metadane

3.2.1 Poziom journal

Poziom journal jest najbezpieczniejszy a ryzyko utraty danych tutaj najmniejsze. Jednocześnie jest to poziom najwolniejszy. Słabsza wydajność spowodowana jest tym, że aby zatwierdzić jakąkolwiek zmianę na dysku potrzeba dwóch operacji zapisu, najpierw do dziennika a dopiero później rzeczywiste dane.Tak wysoki poziom dziennikowania, w którym bezpieczeństwo danych jest znacznie ważniejsze od szybkości ich przetwarzania dostarcza tylko system ext3.

3.2.2 Poziom ordered

Poziom ordered jest domyślnym poziomem dziennikowania w systemie plików ext3. Operacje zapisu danych i metadanych układane są w transakcje, kiedy przyjdzie czas na zapisanie nowych metadanych, najpierw zapisywane są dane, a dopiero później metadane. Dzięki temu, nie jest możliwe by metadane wskazywały na niezapisane jeszcze na dysku dane. Czekanie na zapis wszystkich danych na dysk zmniejsza nieznacznie wydajność systemu plików, jednak w porównaniu z ext2 jest to zmiana mało zauważalna.

3.2.3 Poziom writeback

Writeback jest poziomem z dziennikowaniem tylko metadanych, a co za tym idzie poziomem o najmniejszej utracie wydajności. Jednocześnie przez to, że zapis danych i metadanych przebiega asynchronicznie mogą się pojawić po awarii niespójności. Na przykład metadane mogą wskazywać na bloki, które były w trakcie zapisywania gdy system uległ awarii. Taki poziom kronikowania (bez dziennikowania danych) jest zaimplementowany w większości pozostałych

systemów plików z dziennikowaniem.

3.3 Indeksowane katalogi

Do przechowywania zbioru plików w katalogu zastąpiono reprezentację listową na indeksowaną. Drzewiasta struktura obsługująca haszowanie znajduje się w pierwszych 512 blokach katalogu. Pierwszy z nich zawiera korzeń oraz nagłówek charakteryzujący całą strukturę. Kolejne bloki zawierają pary (klucz, adres-bloku). Adres ten wskazuje na blok z danymi pliku, lub blok z kolejnego poziomu struktury. Zastosowanie funkcji haszującej oraz B-drzewa znacznie zwiększają wydajność systemu, szczególnie przy dużej ilości plików.

3.4 Fragmentacja bloków

System ext3 może podzielić jeden blok na kawałki o tym samym rozmiarze i w każdym z nich przechowywać małe pliki. Pozwala to znacznie zmniejszyć obszar zajmowany przez dane, gdy w systemie jest wiele małych plików.

4 System plików XFS

4.1 Zarys

Ta część prezentacji będzie dotyczyła głównie następujących zagadnień

- podstawowe cechy systemu
- architektura XFS-a
- struktury danych i implementacja

Twórcą XFS jest znany producent superkomputerów oraz wydajnych, graficznych stacji roboczych - firma Silicon Graphics. Prace nad nim zaczęto już w 1993 roku Podstawą do rozpoczęcia prac był brak przystosowania do wzrostu wydajności i rozmiaru pamięci masowych wówczas stosowanego w systemie operacyjnym IRIX systemu plików EFS(odmiana Unixa dla maszyn SGI). Główne ograniczenia EFS to maksymalny dopuszczalny rozmiar partycji - 8 GB oraz maksymalny dopuszczalny rozmiar pliku 2 GB.

Pierwsza oficjalna wersja systemu XFS pojawiła się w grudniu 1994 r. Już w 1996 XFS wyparł EFS i stał się domyślnym systemem plików IRIX-a. Problemem, który pojawił sie przy próbie przeniesienia XFS-a na Linux-a były wymagania, które stawiała architektura XFS-a dla warstwy VFS oraz dyskowych buforów.

4.2 Podstawowe cechy

Do głównych cech wyróżniających system XFS tych systemów plików należą:

- system plików z zaimplementowanych kronikowaniem.
- podział na autonomiczne grupy alokacji.
- opóźniony zapis wykorzystywany min. do redukcji fragmentacji danych
- listy uprawnień znacznie wygodniejsze od POSIX-owej 9-bitowej maski.
- zapis małych plików w i-węźle

Rozwiązując problem przeniesienia XFS-a na Linux-a zdecydowano się na kod pośredniczący aby zmapować wywołania funkcji zgodnych z Linuxem do wywołań zgodnych z systemem IRIX. Architektura ta składa się z dwóch warstw:

- linvfs - odpowiada ona za współpracę z VFS-em
- pagebuf - odpowiada za połączenie z buforem blokowym

Do zalet rozwiązania można zaliczyć uwspólnienie architektury XFS-a dla obu platform IRIX i Linux, natomiast wadą przyjętego rozwiązania jest utrata wydajności w stosunku do implementacji bezpośredniej. Przy projektowaniu IRIX-a ustalono następujące założenia:

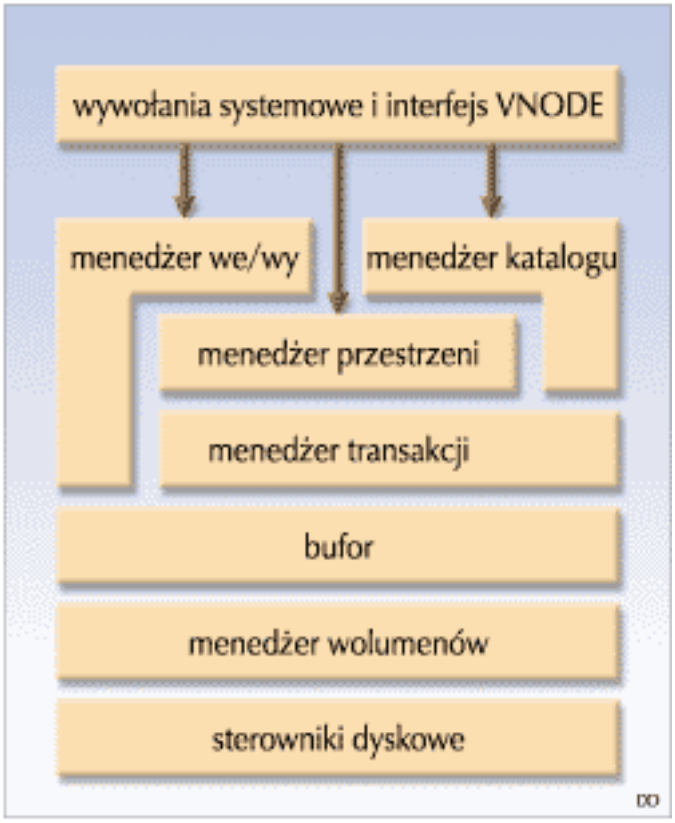
- musi zapewniać wysokie bezpieczeństwo danych i odporność na awarie
- powienien być skutecznie wykorzystywany na naukowych serwerach plików, komercyjnych serwerach oraz serwerach mediów elektronicznych
- powinien sprawnie obsługiwać duże 64-bitowe pliki ale także pliki o bardzo małym rozmiarze (mniej niż 1 Kb)
- powinien także sprawnie obsługiwać pliki rozrzedzone
- liniowe wyszukiwanie musi być wyeliminowane przy dostępie do określonych części pliku
- wydajność operacji na katalogach o dużym rozmiarze
- powinien być kompatybilny z listą kontroli dostępu i innymi funkcjami zgodnymi z POSIX-em
- winien zapewniać obsługę plików o różnych rozmiarach bloków

4.3 Dziennik

XFS odnotowuje w dzienniku wszystkie zmiany dokonywane na metadanych, tzn. w superblokach, nagłówkach grup alokacji, drzewach wolnych obszarów, drzewach i-węzłów, w samych i-węzłach oraz zawartości katalogów. Operacje tego typu są opakowywane w transakcje, będące spójnym i niepodzielnym logicznie ciągiem zmian w blokach systemu plików. Dzięki temu późniejsza naprawa uszkodzonego systemu plików nie polega na naprawie poszczególnych, złożonych struktur, lecz na dokończeniu ostatnio zaplanowanych i przerwanych transakcji. XFS umożliwia tworzenie logów na innym urządzeniu , co zwłaszcza przy synchronicznym trybie pracy może istotnie zwiększyć wydajność (synchroniczny tryb jest wymuszany np. przez NFS).

4.4 Architektura

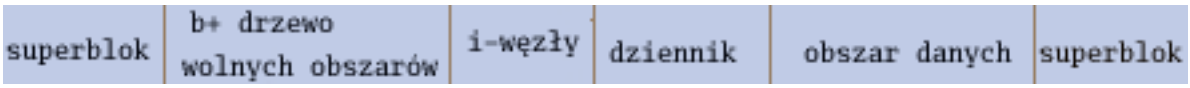
Schemat architektury XFS



XFS ma architekturę modułową. Najważniejszym modułem jest menedżer przestrzeni (Space Manager). Jego zadaniem jest obsługa wszystkich obiektów w systemie plików, a szczególnie zarządzanie i-węzłami oraz ekstentami (ciągły obszar wolnych bloków). Menedżer wejścia/wyjścia (I/O Manager) oraz menedżer katalogów (Directory Manager) są modułami pośredniczącymi pomiędzy VFS IRIX-a a menedżerem przestrzeni. Menedżer wejścia/wyjścia zajmuje się całościową obsługą plików, a drugi - jak sama nazwa wskazuje - katalogów. Menedżer

transakcji zarządza aktualizacją danych. Odpowiada również za operacje księgowania (journaling), które pozwalają na szybką naprawę systemu po awarii. Księgowanie polega na transakcyjności zmian w systemie plików oraz ich logowaniu. W XFS obejmuje ono tylko metadane, takie jak drzewa i-węzłów, i-węzły, zawartość katalogów, drzewa ekstentów zawartości plików, drzewa wolnych ekstentów, bloki nagłówkowe grup alokacji a także superbloki.

Grupa alokacji to autonomiczna jednostka systemu plików XFS. Każda z grup alokacji zawiera struktury danych używane do zarządzania jej zawartością. Schemat grupy alokacji przedstawia poniższy diagram:



Superblok zawiera podstawowe informacje o całej strukturze grupy alokacji. min. wersję XFS, rozmiar bloku, rozmiar poszczególnych części grupy alokacji oraz wskaźnik na drzewo katalogów. Zastąpienie mapy bitowej wolnych obszarów parą B+drzew powoduje, znaczący wzrost wydajności.Pierwsze drzewo to indeks położenia wolnych ekstentów, drugie - ich rozmiarów. Czas znalezienia bloku B+drzewie wynosi O(lgn) podczas gdy w bitmapie O(n). Wspomniane wcześniej B+drzewa przechowują właśnie zbiór ekstentów - pierwsze drzewo pamięta ich adresy, drugie ? rozmiary. I-węzły w XFS nie różnią się znacząco od tych znanych choćby z ext2. Główna różnica polega na przechowywaniu ich w B+drzewie oraz liście.

4.5 Opóźniony zapis danych

Zapis danych w systemie XFS jest dwuetapowy. Po otrzymaniu danych do zapisania XFS trzyma je w specjalnym buforze opóźniając możliwie umieszczenie ich w docelowych blokach na dysku. W ten sposób dane mogą być umieszczone na dysku w sposób bardziej sensowny, np. bez zbędnej fragmentacji. Można także dzięki temu znacząco ograniczyć liczbę operacji na metadanych, a w skrajnych przypadkach mały plik tymczasowy może wogóle nie znaleźć się na nośniku.

4.6 Listy uprawnień

Listy uprawnień do plików (i katalogów) są bardzo wygodnym narzędziem pozwalającym określić uprawnienia dla każdego z użytkowników lub grupy użytkowników osobno.

4.7 Zapis małych plików w i-węźle

XFS potrafi zapisać całą zawartość małego pliku lub katalogu w i-węźle opisującym ten plik, co pozwala zmniejszyć liczbę odwołań do pamięci dyskowej i zaoszczędzić miejsce na dysku.

5 Porównanie wydajności Unixowych systemów plików

Podstawą do krótkiego porównania wydajności kilku Unixowych systemów plików będę tu testy wykonane przez chorwackich studentów z Faculty of Electrical Engineering and Computing at

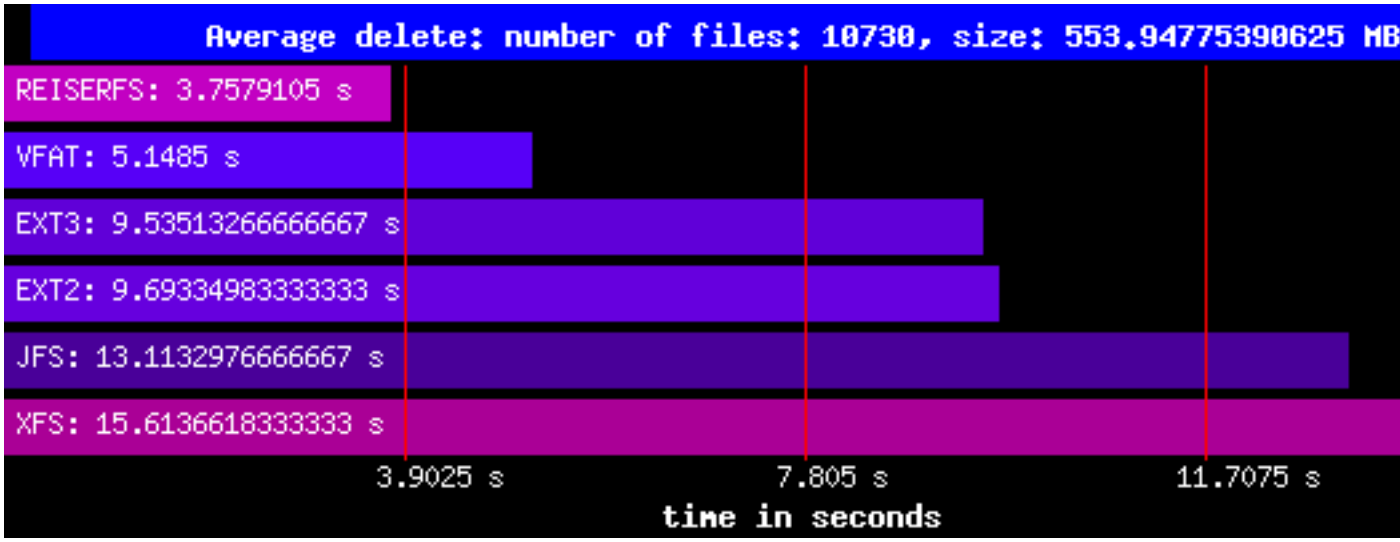
University of Zagreb. Przeprowadzili oni testy wydajnościowe systemów : XFS, ext3, ext3, JFS, vfat, reiserfs na komputerze IBM NetVista, Pentium III 733, 10 GB HD (UDMA66), 384 MB RAM.

5.1 Wyniki testów

Testy podzielono na pracę z małymi plikami (w liczbie około 10000 i łącznym rozmiarze 553 MB) i z dużym plikiem (o rozmiarze 645 MB). Poniżej widać interesujące wyniki:

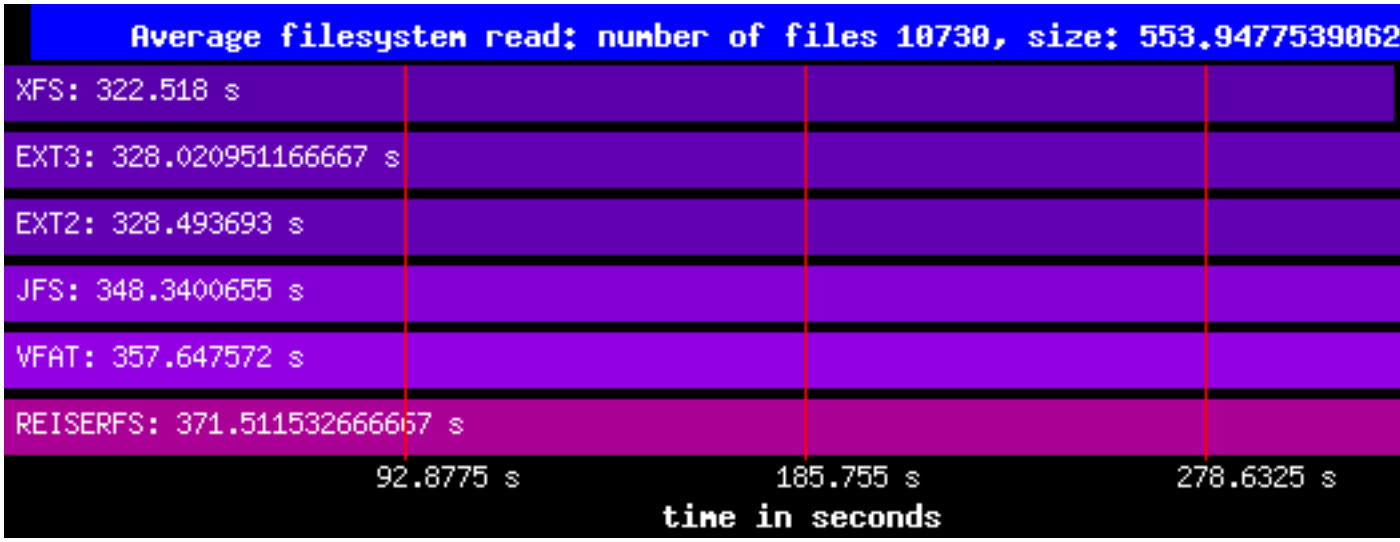
Pliki o małym rozmiarze:

Usuwanie



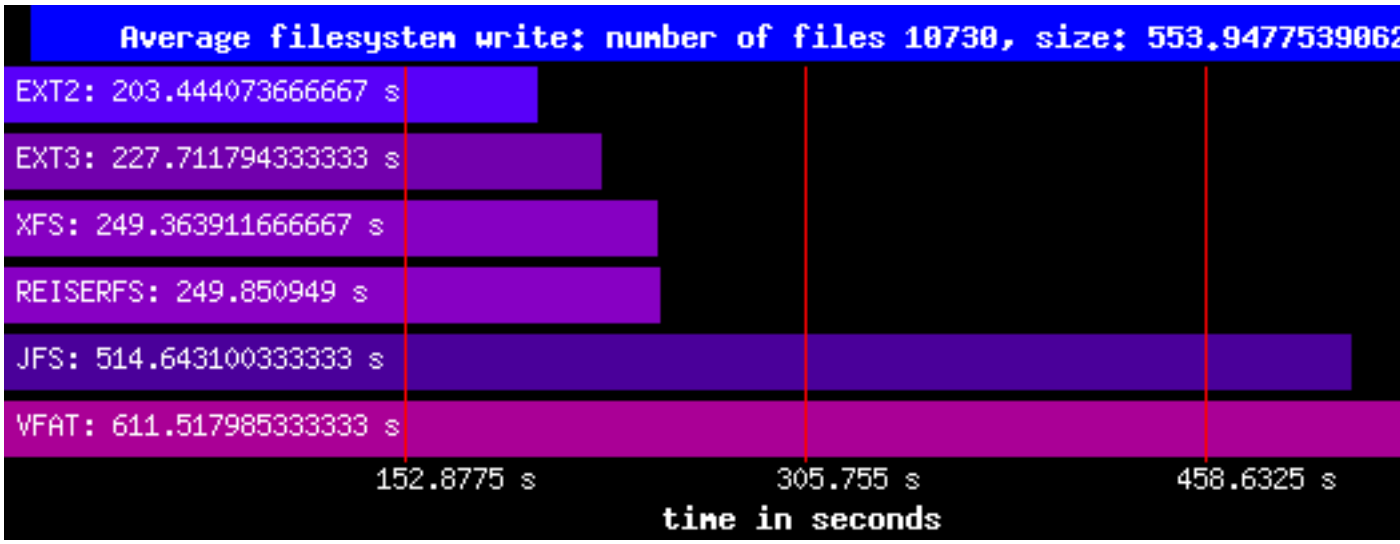
Mamy tutaj absolutnego mistrza reiserFS-a, natomiast XFS wypadł wyjątkowo słabo.

Odczyt



W przypadku odczytu różnice w szybkości są małe i jest tylko około 20między najszybszym i najwolniejszym.

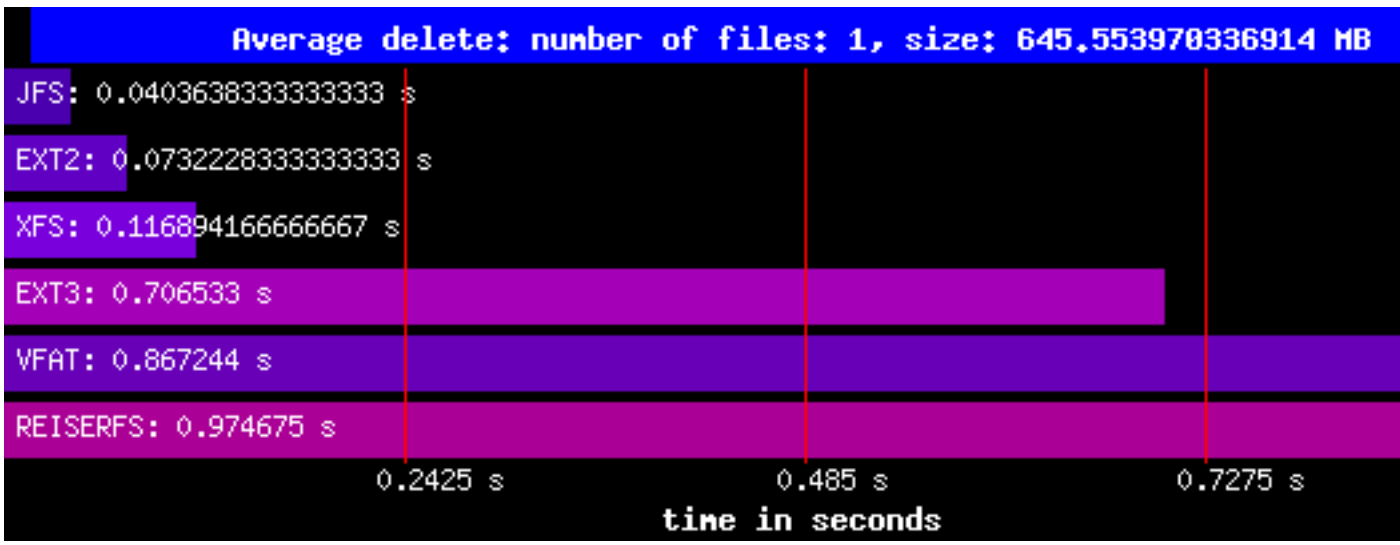
Zapis



Jeśli chodzi o zapis wielu plików o małym rozmiarze to wygląda na to, że ext2, ext3 a również reiserfs wypadają najlepiej.

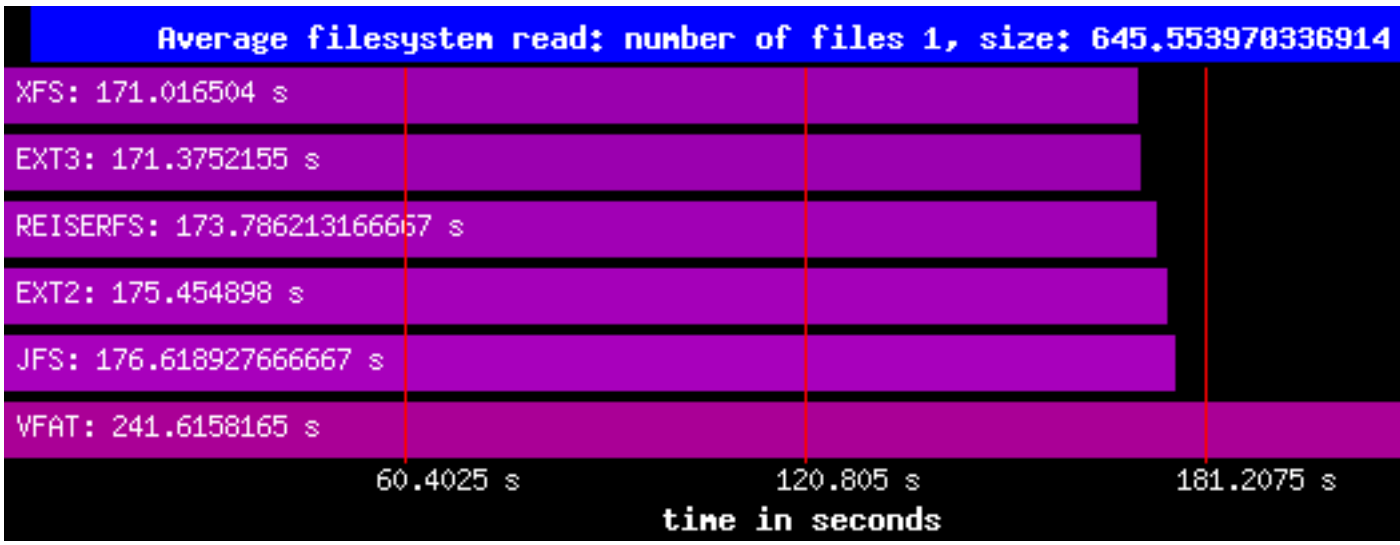
Plik o dużym rozmiarze:

Usuwanie



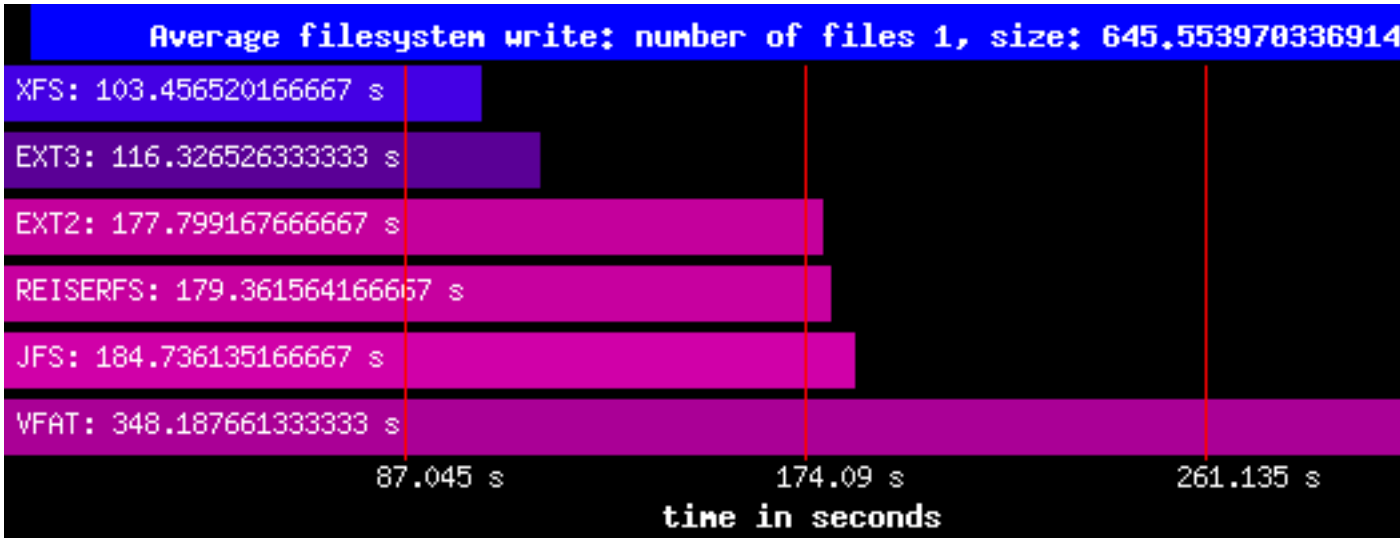
Jeśli chodzi o usunięcie jednego dużego pliku to mimo różnic szybkości są zadowalające.

Odczyt



Odczyt jak widać z wykresu pochłania zbliżoną ilość czasu we wszystkich systemach.

Zapis



W przypadku kopiowania dużego pliku możemy zauważyć, że XFS i ext3 wypadają bardzo dobrze, pozostałe systemy są znacznie wolniejsze.

5.2 Podsumowanie

Jak napisali przeprowadzający testy doszli oni do jednego pewnego wniosku nie jest łatwo zadecydować który system plików jest zwycięzcą.

- ext2 - uznano ten system za dobry wybór jeśli mamy do czynienia z kopiowaniem mniejszych plików i to jest ważne dla przeciętnego użytkownika, przy pracy z dużymi plikami nie jest

on już tak dobry ale użyteczny, oczywiście nie jest to system plików z kronikowaniem, więc dlatego też kwestie mniejsze bezpieczeństwo jest tu ważnym mankamentem

- ext3 - zajął drugie miejsce jeśli chodzi o zapis zarówno dużych i małych plików, z odczytem jest dobrze, jest to system plików z dziennikowaniem i jest bardzo dobrym wyborem
- jfs - jak widać po wynikach szybkość pozostawia wiele do życzenia, pomimo dobrych doświadczeń z AIX, tutaj w testach wypadł raczej słabo
- reiserfs - wypadł on średnio jeśli chodzi o zapis zarówno małych jak i dużych plików, ale w kasowaniu dużej ilości małych plików został uznany za mistrza
- vfat - najgorszy w niemal wszystkich próbach, nie jest polecany
- xfs - testy sugerują, że jest to dobry wybór ale system zawiódł trochę przy próbach kopiowania dużej ilości małych plików.Musimy jednak dodać, że osiągnął najlepsze wyniki na innej platformie i jądrze 2.4.3 w testach przeprowadzonych miesiąc wcześniej.