ReiserFS i Reiser 4 FS

ReiserFS

ReiserFS zwany także Reiser3 to system plików rozpowszechniany jako opensource na licencji GPL zaprojektowany i zaimplementowany przez grupę kierowaną przez Hansa Reisera. Powstał w roku 1996. ReiserFS jest obecnie obsługiwany przez Linuksa i może być w przyszłości włączony do innych systemów operacyjnych. (*Pod adresem http://rfsd.sourceforge.net/ można znaleźć implementację systemu pod Windows. Autor udostępnia również kod źródłowy, więc to nie lada gratka dla developera chcącego zobaczyć jak to działa od środka. Wiele osób nie wie że Microsoft udostępnia Instalowalne Systemy Plików (Installable File System (IFS)) zestaw SDK do pisania sterowników systemów plików. Dzięki temu programista ma możliwosc pisania sterowników obsługujące system plików Linux/OS X z poziomu Windows 2000, XP lub 2003 Serwer.*)

ReiserFS jest domyślnym systemem plików dla SuSE, Lindows i Gentoo. To jeden z pierwszych systemów plików z księgowaniem dla Linuksa. Dla jądra Linuksa w wersji 2.4.1, był pierwszym systemem plików z journallingiem, który włączono do standardowego jądra.

ReiserFS jest kronikowanym systemem plików (domyślnie kronikuje się wyłącznie metadane). Księgowanie (journaling) zapewnia atomowość operacji na systemie plików. Za zwiększenie bezpieczeństwa danych płacimy szybkością (trzeba uaktualniać kronikę) i przestrzenią dyskową (kronika zajmuje miejsce).

ReiserFS to system powstały z myślą o linuksie, w przeciwieństwie do takich systemów jak JFS, czy XFS, które zostały na linuksa przeniesione z innych systemów operacyjnych. Większość systemów plików w Linuksie opera się na wcześniej istniejących systemach plików tym czasem ReiserFS został stworzony zupełnie od zera.

Przy projektowaniu swojego systemu Hans Reiser przyjął dwa założenia:

- W rzeczywistych systemach większość plików to pliki małe (mniejsze niż rozmiar bloku). Takie systemy jak EXT2, EXT3, FAT16, FAT32, NTFS, FFS i inne marnują miejsce alokując cały blok na jeden

pojedynczy mały plik. Warto to miejsce zaoszczędzić. Oprócz oszczędności miejsca ważna jest również szybkość. Należy zwrócić uwagę na szybką obsługę dużej ilości małych plików.

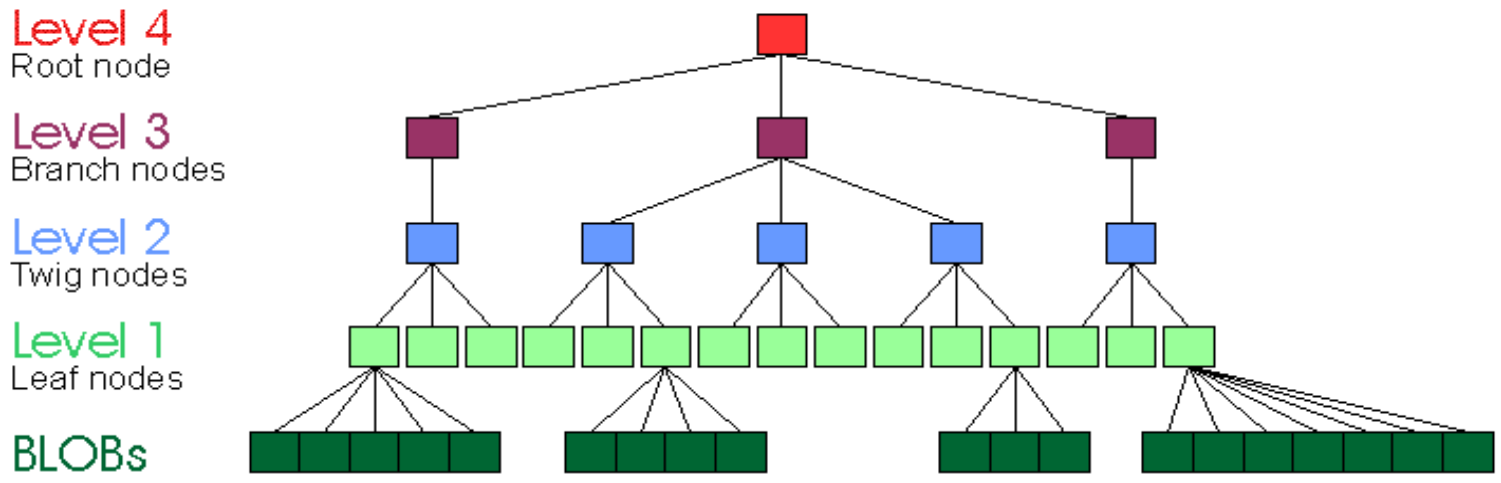
- Drugim założeniem było stworzenie ujednoliconego interfejsu dla plików, katalogów i metadanych, który zapewniłby podobny dostęp do wszystkich tych rodzajów danych w systemie plików.

Podstawowe cechy ReiserFS to:

- fast journaling - minimalizacja czasu sprawdzania integralności dysku
- używanie szybkich drzew zbalansowanych - poprawienie efektywności dla katalogów zawierających bardzo dużo plików
- efektywne wykorzystywanie miejsca - "upychanie" wielu małych plików w jednym bloku

Większość systemów plików przechowuje pliki w całych blokach. Powoduje to, iż mnóstwo miejsca jest marnowane, szczególnie gdy mamy dużo małych plików. Ponadto niepotrzebnie wydłuża się czas operacji wejścia-wyjścia. Na przykład nie jest efektywne przechowywanie typowych obiektów bazodanowych takich jak adresy lub numery telefonów w oddzielnych plikach, ponieważ zaowocuje to stratą ponad 90% miejsca w blokach, które je pamiętają. Reiser upycha małe pliki i kawałki plików do jednego bloku. Daje to wydajność pamięciową około 94% dla małych plików.

Drzewo



W RaiserFS'ie wszystkie dane przechowywane są w drzewie zrównoważonym - B+ drzewie. B+ drzewo jest to B drzewo, w którym elementy są trzymane tylko w liściach natomiast w wierzchołkach wewnętrzych są trzymane wyłącznie dane dodatkowe (klucze).

W strukturze B+ drzewa znajdują się zarówno pliki, katalogi jak i metadane. Dane są umieszczane wyłącznie w liściach lub specjalnych węzłach umieszczonych poniżej poziomu liści, służących do obsługi dużych plików. Takie podejście umożliwia użycie miejsca w węzłach na dodatkowe klucze i wskaźniki, a w rezultacie na zwiększenie stopnia rozgałęzienia drzewa.

Istnieją trzy typy węzłów:

1. Węzły wewnętrzne
   Nie zawierają danych, lecz jedynie wskaźniki do węzłów o poziom niżej, poprzedzielane kluczami. Klucz poprzedzający wskaźnik jest jednocześnie równy pierwszemu kluczowi z poddrzewa, do którego odnosi się ten wskaźnik. Dla układu (klucz1, wsk, klucz2) wszystkie elementy w poddrzewie wskazywanym przez wsk mają klucze z przedziału (klucz1,klucz2).
2. Węzły sformatowane
   Zawierają wpisy. W ReiserFS wyróżniamy cztery rodzaje wpisów(item) - dane bezpośrednie, dane pośrednie, wpisy katalogowe i wpisy informacyjne. Każdy wpis w węźle sformatowanym zawiera swój klucz. Węzły sformatowane są bądź liśćmi bądź mają do siebie podczepione jeszcze liście niesformatowane. Każdy wpis w węźle sformatowanym ma swój nagłówek oraz dane. Zawierają dane (gdy item informacyjny, bezpośredni lub katalogowy) lub listę wskaźników do węzłów niesformatowanych (gdy item pośredni).
3. Liście niesformatowane (BLOB'y czyli Binary Large Objects)
   Nie mają żadnej struktury - służą do przechowywania ciągu bajtów. W liściach niesformatowanych nie ma kluczy. Długość ścieżki do liścia niesformatowanego jest zawsze o jeden dłuższa niż długość ścieżki do węzła sformatowanego (jest to błąd projektowy powodujący spadek wydajności i w wersji Reiser4 został on poprawiony)

W głąb liścia (ITEMS)

B+ drzewo przechowuje w liściach dane rozmiaru bloku (taki jest rozmiar

każdego węzła w B+ drzewie). Dane w liściach nazywa się "pozycjami" (ang. items). W ReiserFS wyróżniamy cztery rodzaje pozycji:

- Dane katalogu

  zawierają wpisy katalogowe (nazwy pozycji w katalogu i ich klucze w drzewie).

- Dane informacyjne (Stat)

  zawierają atrybuty: typ pliku, właściciela i uprawnienia do pliku, jego wielkość itp. informacje w innych systemach plików przechowane w i-węzłach.

- Dane bezpośrednie

  zawierają małe (mieszczące się w jednym bloku) pliki a także "ogony" dużych plików tj. mniejsze od rozmiaru bloku końcówki plików podzielonych na bloki ("reszta z dzielenia"). Jest to szczególnie ważne w przypadku przechowywania na dysku bardzo wielu małych plików. Z drugiej strony, obsługa ogonów niesie ze sobą pewne problemy. Jeśli dopisujemy dane na końcu pliku, to system musi przenosić ogon do innego bloku lub rozbijać ogon na dwa bloki. Możemy wyłączyć mechanizm upychania ogonów, co zwiększy wydajność systemu kosztem większego marnowania miejsca.

- Dane pośrednie

  to wskaźniki do tzw. węzłów niesformatowanych, tj. obszarów zajętych dużymi plikami (ang. BLOB - Binary Large OBject). Te duże pliki znajdują się w drzewie o jeden poziom niżej od liści (nie są to w ścisłym teoriografowym sensie liście, bo wychodzą z tych węzłów krawędzie w dół).

Nadawanie klucza items

Każdemu item nadawany jest klucz, który będzie służył do szeregowania wpisów o pliku w B+ drzewie. Nadawanie klucza jest tak wymyślone by

- item należące do jednego pliku,
- item plików leżących w tym samym katalogu,
- item plików będących w katalogach nadrzędny/podrzędny, znalazły się możliwie blisko siebie w porządku B+ drzewa.

Klucz wyszukiwania w drzewie zawiera:

- identyfikator katalogu nadrzędnego
- identyfikator obiektu
- przesunięcie od początku obiektu do bieżącego bajtu
- typ obiektu (bezpośredni, pośredni, katalog, stat data)

Dzięki takim składowym klucza wyszukiwania obiekty z tego samego katalogu są położone koło siebie oraz fragmenty składowe danego obiektu (pliku lub katalogu) są również w sąsiednich węzłach w drzewie.

Plik składa się ze zbioru wpisów pośrednich, które wskazują na bloki niesformatowane (z danymi) oraz do dwóch węzłów bezpośrednich zawierających ogon pliku. Ogon pliku może wystąpić w dwu węzłach ze względu na mechanizm „upychania ogonów". Jest to mechanizm, który pozwala na przechowywanie w jednym bloku dyskowym ogonów wielu plików co zmniejsza ilość miejsca potrzebną na ich przechowywanie. Jednak powoduje to problem w momencie dopisywania na koniec pliku. Wtedy może zajść potrzeba przeniesienia ogona pliku do innego węzła bezpośredniego lub rozbicia go na dwa wpisy. Stąd ogon może występować w dwu węzłach. Upychanie ogonów można wyłączyć opcją notail.

Wstawianie węzła do drzewa

System ext2 przechowuje informacje o wolnych blokach w bitmapie bloków - 1 bloku dyskowego zawierającego mapę bitową opisującą, które bloki w grupie są wolne, a które zajęte. W systemie ReiserFS używana jest bitmapa wolnych bloków w sposób analogiczny do tego, który zastosowany jest w ext2.

Przeszukuję się bitmapę wolnych bloków zaczynając od lewego sąsiada ostatnio używanego węzła i poruszając się w tym samym kierunku, co ostatnio.

W testach okazało się, że metoda ta jest lepsza od następujących alternatyw:

- wyszukiwanie pierwszego wolnego bloku w bitmapie
- branie pierwszego za ostatnio przydzielonym i poruszanie się w tym samym kierunku, co ostatnio (3% szybsze przy zapisie i 10-20% wolniejsze przy odczycie)
- zaczynanie od lewego sąsiada i poruszanie się w kierunku, branym od prawego sąsiada

Okazało się również, że metoda jest o ~10% szybsza niż gdybyśmy zaczynali od aktualnego węzła, mimo, że ryzykujemy jeden odczyt sięgnięcia do ojca, jeśli lewego sąsiada by nie było.

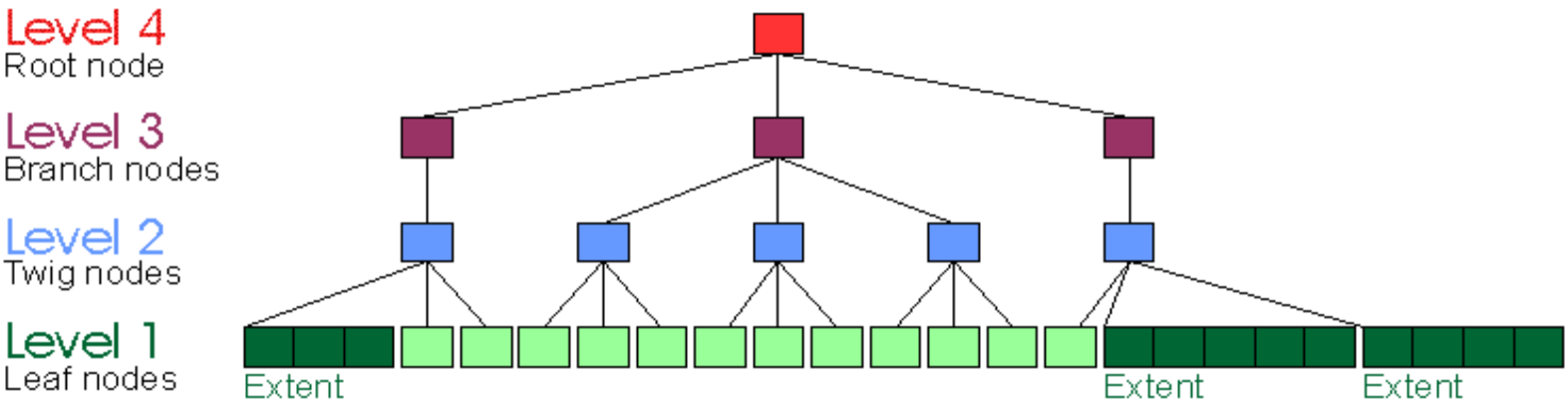
Księgowanie

ReiserFX oferuje asynchroniczny model księgowania metadanych. Oznacza to, że modyfikacje metadanych nie są od razu zapisywane do dziennika, lecz są przez pewien czas przechowywane w buforze dziennika w pamięci. Zapewnione jest przy tym, że żadne dane nie zostaną zapisane na dysk zanim nie zostanie zapisana do dziennika na dysku informacja o tych zmianach. Dzięki asynchronicznemu dziennikowi, informacje o kilku zmianach danych mogą być zapisane do dziennika w jednej operacji dyskowej. Zwiększa to wydajność księgowania, ale powoduje, że po awarii systemu może nie być możliwe odtworzenie wszystkich wykonanych zmian metadanych.

REISER 4 FS

są na tyle znaczne, że zasadne jest używanie nowej nazwy: Reiser4, zamiast ReiserFS. Tym razem twórca położył nacisk na bezpieczeństwo. Reiser4 ma być systemem dla wojska, choć oczywiście ma być również używany przez ludność cywilną. Prace nad tym systemem plików sponsoruje DARPA (Defence Advanced Research Project Agency) i Lindows.com.

Drzewo



Reiser4 powraca do klasycznej definicji drzewa zrównoważonego ze względu na długości ścieżek do liści. Nie próbuje udawać, że węzły, które przechowują obiekty większe niż węzeł, jakimś cudem nie są częścią drzewa. Dzięki temu zużywa się mniej pamięci na pamiętanie wskaźników. B+ drzewa zostały zastąpione "drzewami tańczącymi" (dancing tree).

W drzewie tańczącym wskaźniki do węzłów niesformatowanych przechowujemy poziom wyżej nad liśćmi, tzn. węzły niesformatowane znajdują się na poziomie liści. W zwykłych B+ drzewach po każdej operacji zapewniamy spełnianie warunków B+ drzewa, co jest dosyć kosztowne. Doświadczenia pokazały, że opłaca się opóźniać równoważenie drzewa. Dlatego drzewo tańczące aktualizuje swoją strukturę dopiero po wykonaniu operacji commit przez pewien okres czasu pozostając niezrównoważone. Nie istnieją racje matematyczne, aby przyjąć

takie postępowanie, jest ono podyktowane doświadczeniem.

W drzewie, które zawiera wszystkie dane nt. zawartości dysku, są określone 3 rodzaje wierzchołków:

- **Liscie (leafs)-** nie posiadają dzieci.
  - Wierzchołki, które zawierają itemy są nazywane wierzchołkami sformatowanymi.
  - Niesformatowane liscie (unfleafs) to takie, które zawierają jedynie dane, bez żadnych informacji formatujących. Extent jest sekwencją sasiednich (według numeru bloku) liści, które należa do tego samego obiektu. Wskaźnik na extent zawiera numer bloku startowego extentu i jego długość.

- **Gałązki (twigs)**- są ojcami lisci. Wskaźniki na extenty istnieją tylko w gałązkach.
- **Gałęzie (branch)**- są wierzchołkami wewnętrznymi, które nie są gałązkami.

W Reiserze wszystkie węzły mają ten sam rozmiar. Takie podejście ułatwia alokację nie używanego miejsca pomiędzy węzłami, ponieważ jego rozmiar jest pewną wielokrotnością rozmiaru węzła, zatem nie ma problemu z wolnym miejscem, które jest zbyt małe, aby zmieścić jeden węzeł. Ponadto dyski (stacje dysków) mają interfejs, który przyjmuje równy rozmiar bloków, co jest wygodne dla ich algorytmów korekcji błędów.

EXCENTY

W wersji 4 wprowadzono wreszcie pojęcie przedziału bloków dyskowych (extent) obecne od długiego czasu w innych systemach plików (XFS, JFS, VxFS). Dzięki temu została znacznie poprawiona efektywność obsługi dużych plików (z czym wersja 3 miała poważne problemy).

KRONIKOWANIE

Zwykłe kronikowanie jest dosyć kosztowne, bo wymaga dwukrotnego zapisu danych na dysk - do dziennika i na miejsce docelowe. W Reiser4 wprowadzono

nowatorskie rozwiązanie pozwalające na jednokrotny zapis na dysku. Gdy zapisywanie do dziennika zostaje zakończone, status bloku w którym mamy zapisany dziennik zmienia się i blok staje się "normalnym" elementem systemu plików, gdy tymczasem dziennik zmienia swoje położenie na dysku. Daje to oczywiście duże przyspieszenie.

REPACKER

80% danych na dysku pozostaje nie zmienianych przez długi okres czasu. Wydajnie by
było, gdyby były dobrze rozmieszczone. Reiser4 oferuje specjalny program *repacker*. Przechodzi on drzewo od lewej do prawej, spychając wszystko w lewo i od prawej do lewej, spychając w prawo. Defragmentuje drzewo i upycha dane.

ATOMOWOŚĆ

Wszystkie wywołania systemowe są w Reiserze atomowe. Reiser pozwala definiować nowe atomowe operacje przy użyciu wtyczek. Reiser używa specjalnych algorytmów, które pozwalają uczynić operacje atomowymi przy niewielkim dodatkowym koszcie, podczas gdy inne systemy plików musiałyby płacić wysokie, wręcz niedopuszczalne ceny za coś takiego.

PLUGINY

Reiser4 ma modularną budowę dzięki mechanizmowi wtyczek. Jest osiem rodzajów wtyczek:

- file plugins - udostępniają metody dostępu do plików
- directory plugins - udostępniają metody dostępu do plików
- hash plugins - obsługują haszowanie kluczy w drzewie tańczącym
- security plugins - obsługują bezpieczeństwo
- item plugins - udostępniają metody dostępu do pozycji

- key assignment plugins - zajmują się przydzielaniem kluczy w drzewie
- node search and item search plugins - odpowiedzialne za wyszukiwanie w drzewie węzłów i pozycji

Wtyczki można tworzyć i dodawać. Przy dodawaniu trzeba przekompilować jądro i do-
dać nową wtyczkę na koniec listy. Przewiduje się, że w przyszłych wersjach Reisera dodane
zostaną poprawki, które umożliwią dynamiczne ładowanie wtyczek (w trakcie działania
programu).

Podsumowanie

Reiser4 miał się ukazać już na początku 2003 roku, w tej chwili jest już wreszcie gotowy. Reiser4 wzbudza wielkie nadzieje, ale często zgłaszane są też wątpliwości co do użyteczności proponowanych rozwiązań. System plików Hansa Reisera wersji 4 jest promowany jako najszybszy system plików dla Linuksa. Nie wszedł jeszcze do stabilnej gałęzi jądra lecz jest dostępny w gałęzi tworzonej przez Andrew Mortona (oznaczanej jako -mm), od wersji 2.6.8.1-mm2. Niemniej jednak wydaje się w pełni stabilnym i zasługującym na uwagę systemem plikowym. Testy wydajności Reiser4 na stronie www.namesys.com pokazują, że przy założeniu, że 80% plików ma rozmiar do 8kB jest to system szybszy niż ext2, ext3, reiserFS v3, JFS, XFS. Ważną rzeczą jest tez dokonywanie pełnych translacji, co oznacza pełny zapis lub jego brak dzięki czemu nie pojawia się uszkodzenia plików. Kod Reisera4 jest modularny,czyli będzie można w łatwy sposób rozbudować jego funkcjonalność za pomocą wtyczek.