Systemy plików w Linuksie

Autor: William von Hagen
Tłumaczenie: Przemysław Szeremiota
ISBN: 83-7197-774-3
Tytuł oryginału: Linux Filesystems
Format: B5, stron: 516

Dostępność wydajnych systemów plików z kroniką i rozproszonych systemów plików to jedna z najbardziej ekscytujących cech systemu Linux. Gdy pliki osiągają wielkości setek gigabajtów, a pojemności dysków sięgają terabajtów, tradycyjne rozwiązania przestają wystarczać. Warto wówczas skorzystać z nowoczesnych systemów plików z kroniką zapewniających szybki dostęp do danych także w sytuacjach, gdy danych tych jest bardzo dużo.

Innym wyzwaniem jest coraz częstsza konieczność dzielenia zasobów dyskowych między wiele rozmaitych systemów. Tutaj pomocne są rozproszone systemy plików, takie jak starszy NFS, czy nowszy OpenAFS. Dzięki OpenAFS użytkownicy nie muszą pamiętać, na którym dysku znajdują się dane, co więcej, administrator może przenosić te pliki w obrębie sieci nie zakłócając w żaden sposób dostępu do nich.

Poza przystępnym omówieniem zaawansowanych linuksowych systemów plików, książka ta prezentuje procedury instalacji oprogramowania, umożliwiającego systemom linuksowym współużytkowanie danych z komputerami działającymi pod kontrolą systemów operacyjnych Apple Macintosh, Microsoft Windows oraz Novell NetWare. Dzięki zawartym w książce informacjom zmaksymalizujesz wydajność, elastyczność i niezawodność systemów linuksowych.

Dzięki książce „Systemy plików w Linuksie”:

- Nauczysz się korzystać z możliwości i zalet najpopularniejszych nowoczesnych systemów plików dostępnych dla Linuksa.
- Uświadomisz sobie finansowe, operacyjne i wydajnościowe zalety nowoczesnych systemów plików z kroniką i rozproszonych systemów plików.
- Otrzymasz przejrzyste wskazówki co do sposobów wzbogacenia jądra systemu operacyjnego Linux o obsługę systemów plików z kroniką: ext3, JFS, ReiserFS oraz XFS oraz sposobów konfiguracji, kompilacji, instalacji i uruchamiania rozszerzonych wersji jądra.
- Zaznajomisz się z technologiami związanymi z zarządzaniem woluminami logicznymi, macierzami RAID oraz narzędziami archiwizacyjnymi systemu Linux zwiększającymi wydajność, elastyczność i bezpieczeństwo.
- Dowiesz się jak zapewnić dostęp do wspólnych danych w środowisku heterogenicznym.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Rozdział 7.

System plików z kroniką: XFS

Niniejszy rozdział zawiera opis systemu plików z kroniką XFS, opracowanego przez firmę Silicon Graphics Inc. (SGI) na potrzeby systemu operacyjnego IRIX, a następnie udostępnionego w roku 2000 społeczności linuksowej i reszcie świata jako oprogramowanie open source. Rozdział rozpocznie się od ogólnej charakterystyki systemu XFS i omówienia tych cech projektu i implementacji, które czynią go wysokowydajnym systemem plików z kroniką; następnie przedstawiony zostanie opis sposobu dodania obsługi systemu plików XFS do systemu operacyjnego Linux. Ostatnią część rozdziału tradycyjnie zajmie omówienie kwestii związanych z administracją i korzystaniem z systemu plików XFS, obejmujące również omówienie procedury manualnej weryfikacji spójności systemu plików i metod rozwiązywania ewentualnych problemów.

Opis ogólny

System plików XFS jest wysokowydajnym systemem plików z kroniką, którego od konkurentów (będących również bohaterami tej książki) odróżniają pewne istotne cechy. Oto niektóre z głównych przyczyn wyższości systemu XFS:

- System XFS był od dawna wykorzystywany w różnych systemach komputerowych, jest więc gruntownie przetestowany.
- System XFS obsługuje (przez umieszczenie ich w specjalnym obszarze systemu plików) pliki udostępniane w czasie rzeczywistym.
- System XFS udostępnia obsługę rozszerzonych list kontroli dostępu do plików i katalogów. Zaawansowane funkcje kontroli dostępu, takie jak listy kontroli dostępu wspomagające standardowe zabezpieczenia systemu Linux, są w przypadku pozostałych systemów plików dla Linuksa w fazie planowania; działające implementacje takich mechanizmów można póki co spotkać jedynie w zaawansowanych rozproszonych systemach plików, takich jak AFS.
- System XFS korzysta z zaawansowanych algorytmów wyszukiwania plików i katalogów oraz alokacji przestrzeni dyskowej.

Najbardziej interesującą z tych cech jest fakt, że system był wykorzystywany w środowiskach produkcyjnych od 1994 roku, działając na każdej stacji roboczej SGI w tysiącach instalacji na całym świecie. Ponieważ system plików ma historię dłuższą nawet od historii standardowego linuksowego systemu plików ext2, wybór systemu XFS czy to na użytek prywatny, czy też jako systemu produkcyjnego jest stosunkowo bezpieczną inwestycją. Stabilność i historia systemu plików to ważny czynnik wpływający na decyzję o zastosowaniu systemu plików w każdym środowisku, zwłaszcza w przypadku systemów produkcyjnych lub akademickich. Co prawda wdrożenie systemu plików XFS w systemie Linux może potencjalnie spowodować problemy — systemowy kod obsługi systemów plików i woluminów logicznych różni się całkowicie od kodu XFS. Jednakże tysiące zadowolonych użytkowników i administratorów daje poczucie zaufania do kodu systemu XFS, jego potencjalnie wysokiej wydajności i krótkich czasów przeładowania (i przywracania do stanu spójnego).

Osobom nie znającym firmy Silicon Graphics przypomnę, że firma ta zadebiutowała na początku lat osiemdziesiątych jako producent stacji roboczych wielkości pralki automatycznej. W stacjach tych kładziono nacisk na wysokowydajne przetwarzanie potokowe grafiki o dużej rozdzielczości. Przy takim nacisku na efektywność przetwarzania grafiki potrzebny był wysokowydajny system plików, umożliwiający przechowywanie efektów obróbki graficznej. Pierwszy własny system plików firmy SGI nosił nazwę EFS i był w zasadzie opartą na koncepcji obszarów wersją uniksowego systemu Berkley Fast File System (FFS), wspomnianego w rozdziale 1., zatytułowanym „Wprowadzenie do systemów plików". W wyniku wzrostu pojemności i wydajności dysków twardych SGI zdecydowała się na implementację nowego systemu plików, XFS, rozszerzonego w stosunku do poprzednika o funkcje kroniki i eliminującego wiele ograniczeń systemów FFS i EFS, takich jak wstępna alokacja i-węzłów systemu plików. Premierowe wydanie systemu plików XFS miało miejsce w roku 1994 przy okazji wydania systemu operacyjnego IRIX 5.3 — od tej pory system XFS był stale rozszerzany i ulepszany.

Jako producent uznanych stacji roboczych, przystosowanych do grafiki wysokiej rozdzielczości i multimediów, firma SGI opracowując system plików XFS brała pod uwagę następujące kwestie:

- wysoką przepustowość strumieni audio i wideo
- obsługę dużych plików wymaganych przy przetwarzaniu sekwencji wideo i pracy z programami graficznymi
- zdolność do przechowywania gigantycznych ilości danych
- obsługę dużej liczby plików (a tym samym obsługę dużych katalogów)

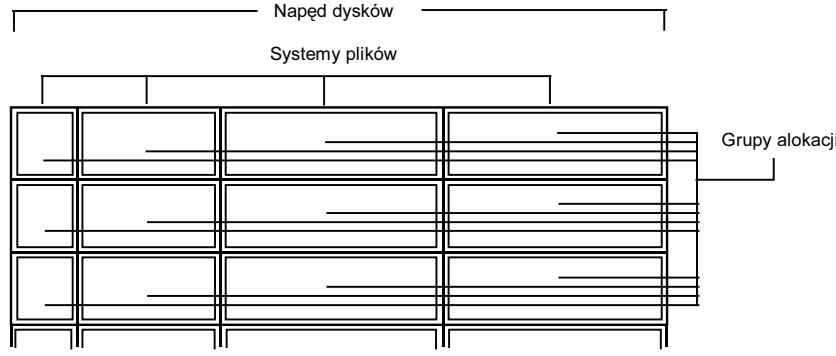
Oto interesująca dygresja: w okresie formowania sieci World Wide Web stacje robocze firmy SGI były ogólnie znane jako znakomite serwery WWW i węzły usługodawców internetowych (ISP) obsługujące wielu użytkowników. Podstawowym powodem takiej popularności stacji SGI jako serwerów WWW była szybkość wykonywania operacji WE-WY, obsługa wielu plików i katalogów oraz łatwość rozszerzania istniejącego systemu plików w celu obsługi większej liczby użytkowników, plików czy katalogów. Brzmi znajomo? Oczywiście — są to cechy systemu plików XFS.

Przechowywanie danych w systemie plików XFS

System plików XFS to w pełni 64-bitowy system plików złożony z trzech podstawowych obszarów:

- sekcji danych, w której przechowywane są właściwe dane i metadane systemu plików;
- dziennika, w którym rejestrowane są transakcje opisujące zmiany metadanych systemu plików (dziennik jest zwykle podczas normalnego działania systemu plików zapisywany sekwencyjnie; odczyt następuje podczas montowania systemu XFS);
- opcjonalnej sekcji czasu rzeczywistego (ang. *real-time*), służącej do przechowywania danych plików wymagających ciągłych, szybkich operacji WE-WY (podwoluminy czasu rzeczywistego są w systemie Linux w fazie rozwoju i obecnie nie powinny być wykorzystywane w systemach produkcyjnych).

Sekcja danych systemu plików XFS jest podzielona na grupy alokacji (patrz rysunek 7.1). Grupy alokacji są niewidocznymi dla użytkowników, rozłącznymi podzbiorami przestrzeni dyskowej, dostępnej w sekcji danych systemu plików XFS. Odpowiadają koncepcji grup cylindrów, zastosowanej w systemie FFS. Grupy alokacji dzielą system plików XFS na niezależne pule przestrzeni dyskowej o rozmiarach od 0,5 do 4 GB.



Rysunek 7.1. *Organizacja systemu plików XFS na dysku*

Każda grupa alokacji utrzymuje własne struktury danych zarządzające wolnym miejscem, i-węzłami oraz blokami zajętymi w ramach grupy alokacji. Takie podejście ma szereg zalet, między innymi:

- Grupy alokacji mogą korzystać z względnych wskaźników bloków i i-węzłów, co pozwala zaoszczędzić miejsce w wewnętrznych strukturach danych grup alokacji.
- Alokacja przestrzeni wewnątrz grupy alokacji tworzy kolejny poziom pośredni w mechanizmie alokacji odwołań do bloków, zwiększając potencjalny rozmiar systemu plików XFS w porównaniu do systemów zmuszonych do utrzymywania jawnych i bezwzględnych wskaźników bloków.

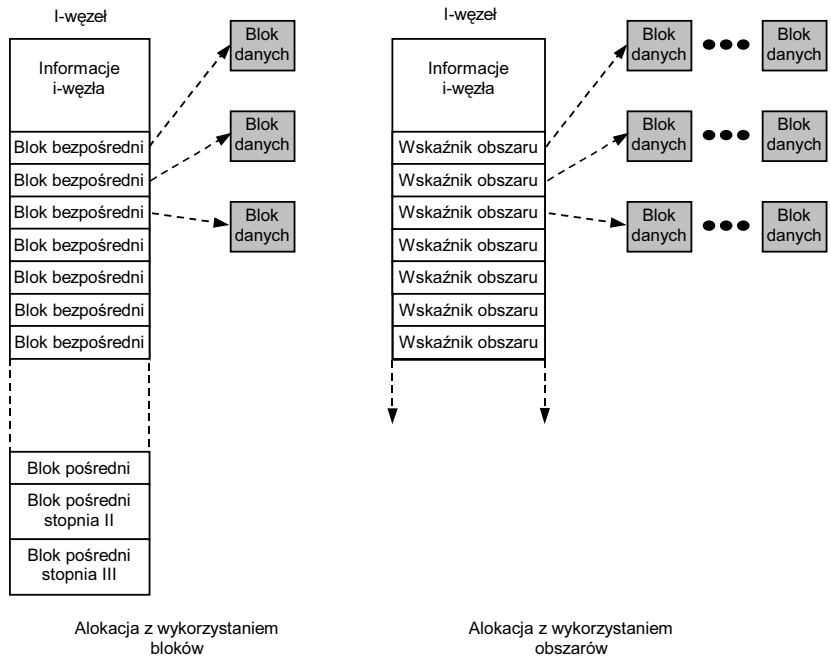
- Utrzymywanie struktur danych, takich jak superbloki, dla każdej grupy alokacji zwiększa wydajność przez redukcję opóźnień blokad i umożliwia równoległą (współbieżną) obsługę operacji przydziału i dostępu do bloków w różnych grupach alokacji.
- Podział systemu plików na pewną liczbę grup alokacji, z których każda utrzymuje własne struktury danych, ułatwia zmiany rozmiaru systemu plików. Zmiany te polegają na prostym dodaniu do systemu plików nowych grup alokacji. W uproszczeniu procesu zwiększania rozmiaru systemu plików pomagają również inne zaawansowane funkcje systemu plików XFS, takie jak dynamiczna alokacja i-węzłów.

Jednym z założeń techniki grup cylindrów, wprowadzonej w systemie FFS, było promowanie koncepcji skupienia dysku (ang. *disk locality*), polegającej na przydzielaniu bloków danych przynależących do tego samego pliku czy katalogu w tej samej grupie cylindrów. Mechanizm taki pozwala zmniejszyć narzuty czasowe związane z koniecznością wielokrotnego przesuwania głowic podczas dostępu do pliku. Rozmiar grup alokacji, stosowany w systemie XFS, redukuje prawdopodobieństwo umieszczenia kolejnych bloków danych pliku czy katalogu w bezpośrednim sąsiedztwie, w zamian wprowadzono więc pojęcie skupienia logicznego (ang. *logical locality*), polegającego na optymalnym w ramach grupy alokacji rozmieszczeniu wszystkich obszarów dyskowych przypisanych do danego pliku.

Aby zwiększyć wydajność operacji przydziału przestrzeni dyskowej do plików, system XFS w miejsce metody alokacji pojedynczych bloków stosuje model alokacji oparty na obszarach (ang. *extents*). Zgodnie z rysunkiem 7.2, obszary są szeregami przylegających do siebie wolnych bloków dyskowych, alokowanych wspólnie. Struktury danych, identyfikujące lokalizację dyskową plików i katalogów w systemach plików opartych na alokacji, przechowują (zamiast ciągu bloków i wskaźników bloków pośrednich) początkowe położenie pierwszego bloku obszaru i jego długość. Rysunek 7.2 prezentuje różnicę pomiędzy alokacją opartą na blokach a alokacją całych obszarów.

Zastosowanie modelu alokacji obszarów pomaga zachować główny cel koncepcji skupienia dysku, a ponadto:

- daje wyższą wydajność operacji przydziału w porównaniu z operacjami przydziału wielu pojedynczych bloków;
- daje wyższą wydajność odczytu danych pliku, ponieważ bloki mogą być odczytywane bezpośrednio po sobie, bez konieczności odszukiwania kolejnych bloków na całej powierzchni dysku; zwiększenie wydajności operacji odczytu zwiększa z kolei współbieżność operacji dostępu do wielu plików w grupie alokacji lub w całym systemie plików;
- redukuje rozmiar struktur danych niezbędnych do przechowywania informacji o przestrzeni dyskowej przydzielonej do pliku, zwiększając wydajność i redukując łączny rozmiar metadanych systemu plików;
- redukuje liczbę operacji wyszukiwania wymaganych podczas odczytu istniejącego pliku; odczyt pojedynczego obszaru jest szybszy niż odczyt wielu osobnych wskaźników bloków;



Rysunek 7.2. *Modele alokacji w systemach plików opartych na blokach i w systemach plików opartych na obszarach*

- ♦ redukuje fragmentację plików, zwiększając szansę zmieszczenia wszystkich danych związanych z danym plikiem w ciągłym obszarze dysku.

System plików XFS korzysta też z zaawansowanego mechanizmu przydziału opóźnionego, stosowanego podczas zapisu plików na dysk. Algorytm ten umożliwia systemowi XFS optymalizację alokacji obszarów dla nowych i aktualizowanych plików. Dzięki redukcji liczby osobnych operacji zapisu w systemie plików osiągnięto również zwiększenie wydajności tych operacji.

Poza korzystaniem z koncepcji obszarów, system XFS stosuje podczas tworzenia katalogu pewne heurystyki, mające na celu dalsze zwiększenie wydajności. Każdy nowy katalog tworzony w systemie plików XFS jest umieszczany w innej grupie alokacji niż jego katalog nadrzędny, a dla przyszłych elementów katalogu rezerwowana jest pewna ilość ciągłej przestrzeni dyskowej. System XFS próbuje również alokować i-węzły plików i obszary w bezpośrednim sąsiedztwie dyskowego obszaru katalogu, w którym pliki te zostały założone.

Podobnie jak każdy inny linuksowy system plików, metadane związane z każdym z plików i katalogów systemu plików utrzymywane są w systemie XFS w strukturach zwanych i-węzłami. Jednakże system XFS alokuje i-węzły w miarę potrzeby (dynamicznie), nie marnując czasu na wstępną alokację i-węzłów podczas zakładania systemu plików. Zmniejsza to ilość zajętego miejsca, zajmowanego przez i-węzły nie istniejących jeszcze plików i katalogów. Co ważniejsze, dzięki takiemu podejściu zwiększana jest elastyczność systemu plików XFS, nie jest on bowiem ograniczony do konkretnej liczby plików i katalogów. Każdy system XFS może zawierać dowolną liczbę plików i katalogów, ograniczoną jedynie ilością dostępnego miejsca i przestrzenią adresową 64-bitowych

wewnętrznych struktur danych. Ułatwia to znacznie operacje rozszerzania istniejących systemów plików XFS, odpada bowiem konieczność wykonywania żmudnych ćwiczeń w rodzaju wykonywania kopii danych, zwiększania rozmiaru systemu plików i przywracania danych z kopii zapasowej.

Do systemu plików mogą być w prosty sposób dodawane nowe grupy alokacji — operacje takie wymagają jedynie niewielkich zmian metadanych systemu plików, nie jest bowiem konieczna wstępna alokacja i-węzłów, a większość struktur danych, odnoszących się do nowych grup alokacji, przechowywanych jest wewnątrz tychże grup. Jedynym rzeczywistym ograniczeniem zwiększania rozmiaru systemu plików XFS jest to, że nowe grupy alokacji muszą mieć ten sam rozmiar, co grupy alokacji utworzone podczas zakładania systemu plików.

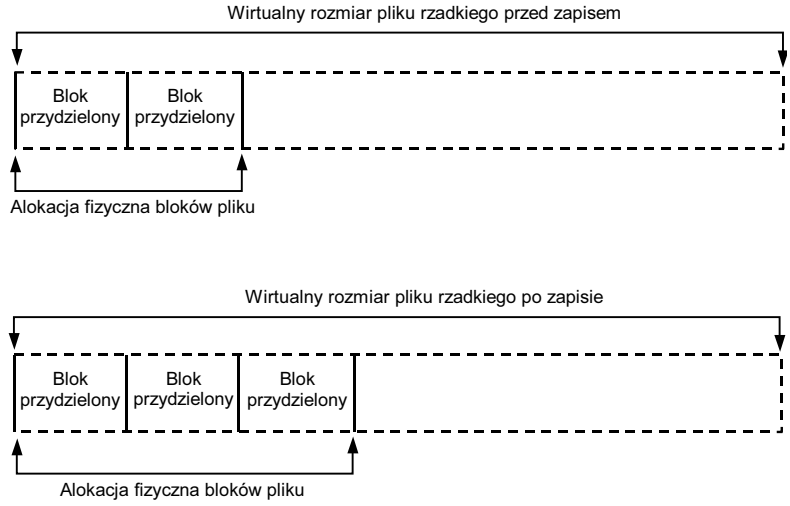
Rozmiar bloków logicznych, wykorzystywanych przez system XFS (podobnie jak w przypadku innych nowoczesnych systemów plików), waha się od 512 bajtów do 64 kB. W połączeniu z 64-bitowymi strukturami danych daje to maksymalny rozmiar systemu plików wynoszący 18 petabajtów (18 milionów gigabajtów) przy maksymalnym rozmiarze pliku wynoszącym 9 petabajtów. Co prawda system operacyjny Linux ogranicza rozmiar obsługiwanych urządzeń blokowych w znacznie większym stopniu, ale w momencie, w którym z jądra Linuksa usunięte zostaną te ograniczenia, system plików XFS będzie już gotów do wykorzystania nowych możliwości.

Specjalne cechy systemu XFS

System plików XFS obsługuje wewnętrznie tzw. rzadkie pliki (ang. *sparse files*), będące plikami o dużych rozmiarach zawierającymi stosunkowo niewielkie ilości danych. Dane te mogą być zapisywane do dowolnego miejsca pliku rzadkiego, bez konieczności przydzielania z góry większej ilości miejsca niż jest to potrzebne do przechowywania danych pliku (poza pewną niewielką ilością metadanych systemu plików). Mimo że rozmiar plików rzadkich zwracany przez system plików jest zawsze ich rozmiarem maksymalnym, bloki danych są do niego przydzielane dopiero podczas zapisu danych w pliku — odczyt niezapisanego wcześniej fragmentu pliku da w wyniku blok wypełniony zerami. Pliki rzadkie świetnie nadają się do niektórych baz danych i innych dużych, podzielonych na rekordy plików; aplikacje korzystające z plików rzadkich mają dostęp do znacznej przestrzeni adresowej, nie marnując przy okazji przestrzeni dyskowej, której jeszcze nie wykorzystują. Pojęcie logicznego rozmiaru pliku rzadkiego i metody alokacji bloków w takim pliku ilustruje rysunek 7.3.

Na wydajność systemu plików XFS pozytywny wpływ ma nie tylko implementacja koncepcji grup alokacji i metody alokacji opartej na obszarach; system XFS przewyższa inne systemy plików również dzięki przechowywaniu metadanych systemu plików w węzłach zrównoważonych drzew binarnych (B+drzew). Koncepcja B+drzew omówiona została w rozdziale 1., a konkretnie w podrozdziale „Model systemu plików w systemie Unix (Linux)”. System XFS wykorzystuje B+drzewa do:

- Zarządzania dynamicznie alokowanymi i-węzłami. I-węzły są przydzielane wewnątrz każdej grupy alokacji w zestawach po 64 i-węzły, a B+drzewo odwzorowuje zarówno ich położenie, jak i to, które i-węzły z danego zestawu są zajęte.



Rysunek 7.3. *Alokacja bloków w pliku rzadkim*

- Śledzenia położenia wolnych obszarów (B+drzewa zastępują tu mapy bitowe stosowane w tradycyjnych systemach plików). System XFS wykorzystuje parę B+drzew do odwzorowywania stanu przydziału obszarów w ramach każdej grupy alokacji; jedno z drzew jest indeksowane blokami początkowymi każdego z obszarów (co pomaga w minimalizacji fragmentacji podczas przydzielania obszarów), drugie indeksowane jest rozmiarem każdego z obszarów (przydatnym podczas szybkiego wyszukiwania wolnych obszarów o określonych rozmiarach).
- Indeksowania nazw wpisów katalogów (w miejsce liniowego przeszukiwania katalogów stosowanego w tradycyjnych systemach plików). Ponieważ nazwy plików mają zmienną długość (od 1 do 256 znaków), B+drzewa przechowujące nazwy wpisów katalogu wykorzystują jako klucze indeksujące 4-bajtowe skróty nazw plików. Korzystanie z kluczy o ustalonym rozmiarze wprowadza jednak niekiedy (w wyniku kolizji skrótów kluczy) konieczność dwukrotnego przeszukania drzewa.
- Odwzorowywania przydziału obszarów do pliku, dzięki czemu zwiększana jest liczba bezpośrednich wpisów obszarów przechowywana w pojedynczym i-węźle.

Jeżeli weźmiemy pod uwagę fakt, że głównym zadaniem stacji roboczych firmy SGI było efektywne przetwarzanie strumieni multimedialnych, wymagania co do wydajności systemu plików XFS nie będą dla nas specjalnie zaskakujące. Jednakże poza wspólnymi dla wszystkich nowoczesnych systemów plików cechami zwiększającymi wydajność, takimi jak alokacja obszarów, dynamiczny przydział i-węzłów i szerokie wykorzystanie B+drzew, system XFS posiada pewne szczególne cechy, zapewniające mu wydajność wymaganą przez zaawansowane aplikacje multimedialne.

Najbardziej interesującą z tych cech jest unikalna zdolność tworzenia podwoluminów czasu rzeczywistego (ang. *real-time volumes*), zapewniających utrzymanie tzw. GRIO (Guaranted Ratio I/O), czyli gwarantowanej przepustowości operacji wejścia-wyjścia. Woluminy te przeznaczone są na użytek plików wymagających przesyłania z ustaloną

szybkością transmisji. Podwoluminy czasu rzeczywistego systemu XFS to fragmenty systemu plików, przechowujące wyłącznie dane plików przetwarzanych w czasie rzeczywistym. Wszystkie metadane związane z tymi plikami przechowywane są w głównej części systemu plików. Wskazanie, że dany plik wymaga operacji czasu rzeczywistego, odbywa się za pośrednictwem wywołania funkcji `ioctl` podczas tworzenia pliku. Istniejące pliki mogą zostać skopiowane do obszaru czasu rzeczywistego za pośrednictwem polecenia `xfs_rtcp`, które automatycznie nadaje plikom atrybuty plików przetwarzanych w czasie rzeczywistym. Podwoluminy czasu rzeczywistego systemu plików XFS korzystają z odmiennej strategii przydziału bloków, zoptymalizowanej pod kątem szybkiej alokacji i minimalnej fragmentacji. Metoda ta nie jest tak sprawna, jak metoda stosowana w standardowych obszarach systemu plików XFS, gwarantuje jednak szybką alokację i minimalne narzuty związane z przesunięciami głowic podczas dostępu do pliku.

Obsługa podwoluminów czasu rzeczywistego w systemie Linux nie jest jeszcze kompletna. Funkcja ta nie powinna być więc stosowana (według stanu z okresu powstawania tej książki). Cierpliwości!

Podsumowując, system XFS to wysokowydajny system plików, korzystający z wyrafinowanych algorytmów, struktur danych i zaawansowanych funkcji heurystycznych. Gruntowne przetestowanie i wieloletnia historia systemu jako podstawowego systemu plików stacji roboczych firmy SGI wskazuje, że system ten jest jednym z poważniejszych kandydatów dla środowisk produkcyjnych kontrolowanych przez system operacyjny Linux.

Oprogramowanie narzędziowe dla systemu plików XFS

Jak dotychczas żadna z wykorzystywanych przeze mnie komercyjnych dystrybucji systemu Linux nie zawierała obsługi systemu plików XFS. To, czy dana dystrybucja zawiera obsługę systemu plików XFS, zależy wyłącznie od dobrej woli osób przygotowujących pakiety dystrybucyjne. W bieżącym podrozdziale przedstawię programy narzędziowe dla systemu XFS i pakiety linuksowe, w których narzędzia te można znaleźć. Jeżeli Twoja dystrybucja Linuksa nie zawiera obsługi systemu plików XFS, możesz dodać ją samodzielnie, pobierając uprzednio:

- kod źródłowy lub pliki binarne odpowiedniej wersji jądra systemu Linux, obsługującej system XFS;
- kod źródłowy i łaty programowe z obsługą systemu XFS dla aktualnie stosowanej wersji jądra;
- kod źródłowy lub pliki binarne pakietów, które zawierają narzędzia służące do montowania, modyfikacji i administracji systemem XFS.

Jeżeli stosowana przez Ciebie dystrybucja Linuksa i stosowana w niej wersja jądra systemowego nie obsługuje systemu plików XFS, na płycie CD-ROM dołączonej do książki znajduje się przykładowy kod źródłowy jądra (w podkatalogu */kernels*) oraz

łaty programowe przystosowujące jądro do obsługi systemu XFS (w podkatalogu */xfs*). Procedury instalacji kodu źródłowego jądra, łat oraz kompilacji jądra omówione zostały w dalszej części rozdziału.

Jeżeli Twoja dystrybucja nie uwzględnia obsługi systemu plików XFS, oprócz jądra będziesz musiał zainstalować i skompilować kilka pakietów programów narzędziowych, umożliwiających instalację, stosowanie i administrowanie systemem plików XFS. Poniżej znajduje się lista programów narzędziowych dla systemu XFS. Są one umieszczone w podkatalogu */xfs* płyty CD-ROM dołączonej do książki.

- `acl-1.1.2` lub nowszy — służy do obsługi list kontroli dostępu do plików i katalogów systemu plików XFS. Listy kontroli dostępu (ACL) stanowią otwarty, konfigurowalny zestaw uprawnień w stosunku do plików i katalogów, które możemy przypisywać poszczególnym użytkownikom i ich grupom. Listy kontroli dostępu systemu XFS są zgodne z proponowaną wersją standardu POSIX 1003.1e (nie jest to co prawda najbardziej aktualny standard, ale należy docenić chęć jakiejkolwiek standaryzacji list kontroli dostępu implementowanych przez SGI w systemie XFS).
- `chacl` — pozwala na modyfikację, analizę lub usunięcie użytkownika, grupy, maski lub innych obiektów listy kontroli dostępu w stosunku do wybranych plików lub katalogów.
- `getfacl` — pozwala analizować listy kontroli dostępu do plików i katalogów.
- `setfacl` — pozwala ustawiać atrybuty list kontroli dostępu do plików i katalogów.
- `attr-1.1.3` lub nowszy — udostępnia obsługę rozszerzonych atrybutów plików i katalogów systemu plików XFS. Atrybuty rozszerzone to pary typu nazwa-wartość, które można przypisać do różnych obiektów systemu plików i które można przypisać do różnych obiektów systemu plików. Atrybuty rozszerzone mogą być wykorzystywane do przekazywania wskazówek odnoszących się do zawartości plików. Wskazówkami tymi może być np. rodzaj stosowanego zestawu znaków, istnienie podglądu plików graficznych itp..
- `attr` — służy do dodawania, tworzenia i usuwania atrybutów rozszerzonych plików.
- `dmapi-0.2.2` lub nowszy — biblioteki interfejsu Data Management API dla systemu plików XFS. DMAPI to standardowy interfejs programistyczny (API) dla systemu zarządzania pamięciami wielopoziomowymi (ang. *hierarchical storage management*). System ten zdefiniowany został przez konsorcjum X/Open w dokumencie „System Management: Data Storage Management (XDSM) API” (w lutym 1997 roku).
- `xfsdump-1.1.3` lub nowszy — program narzędziowy do wykonywania kopii zapasowej i przywracania danych z kopii dla systemu plików XFS; odpowiednik standardowego linuksowego zestawu `dump/restore`.
- `xfs_copy` — narzędzie do kopiowania systemu plików XFS na jeden lub wiele urządzeń docelowych, plików itd. Źródłowy system plików musi być w czasie kopiowania odmontowany, aby zagwarantować jego (i dziennika) niezmienność na czas operacji kopiowania.

- `xfsdq` — program, za pomocą którego wykonywany jest zrzut informacji o kontyngentach dyskowych (ang. *quota*) systemu plików XFS.
- `xfsrq` — program służący do przywracania ustawienia kontyngentów dyskowych systemu XFS na podstawie pliku wygenerowanego przez program `xfsdq`.
- `xfsdump` — program przyrostowego wykonywania kopii zapasowej systemu plików XFS; odpowiednik standardowego linuksowego programu `dump`, rozszerzony o obsługę metadanych charakterystycznych dla systemu XFS, takich jak atrybuty list kontroli dostępu, identyfikatory plików itp.
- `xfs_estimate` — za pomocą tego narzędzia można oszacować rozmiar, jaki zająłby istniejący katalog w systemie plików XFS. Jest to przydatne podczas przenoszenia istniejących systemów plików i całych hierarchii katalogów do systemu XFS. Opcje programu umożliwiają eksperymentowanie z szacunkowym rozmiarem systemów plików XFS przy różnych rozmiarach bloków, różnych rozmiarach dziennika, różnych położeniach dziennika itd.
- `xfs_fsr` — program określający optymalną dla istniejącego systemu plików XFS organizację systemu na dysku. Program ten działa na zamontowanym systemie plików, defragmentując system plików w celu uzyskania najlepszego możliwego zagospodarowania obszarów.
- `xfsinvutil` — program służący do sprawdzania bazy danych przyrostowej kopii zapasowej zapisanej przez program `xfsdump`, weryfikujący jej spójność i ewentualnie usuwający niepotrzebne (nieaktualne już) wpisy.
- `xfsrestore` — program przywracający zawartość systemu plików XFS z kopii zapasowej wykonanej programem `xfsdump`. Program `xfsrestore` może służyć do przywracania pojedynczego zestawu kopii `xfsdump` (w tzw. trybie prostym — ang. *simple mode*) lub może przyrostowo przywracać hierarchiczny zbiór zestawów kopii w celu całkowitego przywrócenia zawartości systemu plików XFS (w trybie zbiorczym — ang. *cumulative mode*).
- `xfsprogs-1.3.5` lub nowszy — podstawowy zestaw programów narzędziowych do tworzenia, rozszerzania, korygowania i weryfikacji spójności systemów plików XFS.
- `fsck.xfs` — jeden z moich ulubionych programów dla systemu XFS. Jest to krótki program napisany w języku C, który niezależnie od sposobu wywołania zwraca po prostu wartość `true`; program został dołączony jedynie dla zgodności ze standardowymi konwencjami nazewniczymi oprogramowania narzędziowego sprawdzającego spójność systemów plików. Gwarancję spójności systemu plików XFS stanowi procedura odtwarzania dziennika systemu plików podczas jego montowania.
- `mkfs.xfs` — program tworzący system plików XFS. Program ten jest wywoływany za pośrednictwem polecenia `mkfs -t xfs`. Opcje programu umożliwiają operatorowi sterowanie procesem tworzenia systemu plików. Dokładniejsze omówienie tego programu znajduje się w punkcie „Tworzenie systemu plików XFS”.
- `xfs_admin` — narzędzie wykorzystujące program `xfs_db` do analizy lub modyfikacji parametrów (odmontowanego) systemu plików XFS.

- `xfs_bmap` — program ten drukuje listę bloków dyskowych zajmowanych przez pliki systemu XFS.
- `xfs_check` — program sprawdzający spójność (odmontowanego lub zamontowanego w trybie tylko do odczytu) systemu plików XFS. W przypadku wykrycia błędów system plików może zostać naprawiony za pomocą programu `xfs_repair`. Jeżeli wykryte błędy są błędami potencjalnie krytycznymi, możliwe jest wykonanie kopii zapasowej systemu za pomocą programu `xfsdump`, jednak powodzenie tej operacji zależne jest od rodzaju uszkodzenia systemu plików.
- `xfs_db` — program przeznaczony do analizy i modyfikacji systemu plików XFS. Program ten jest rzadko wykorzystywany bezpośrednio przez operatora, jest za to często wywoływany przez rozmaite skrypty administracyjne.
- `xfs_freeze` — program ten zawiesza (lub przywraca) dostęp do zamontowanego systemu plików. Zawieszenie dostępu do systemu XFS daje możliwość opróżnienia kolejki oczekujących operacji zapisu i utworzenia chwilowo spójnego stanu systemu plików w celu wykonania jego obrazu lub kopii zapasowej.
- `xfs_growfs` — umożliwia zmianę rozmiaru istniejącego, zamontowanego systemu plików.
- `xfs_info` — wyświetla informacje o zamontowanym systemie plików XFS i weryfikuje ustawienia parametrów przekazywanych do programu `xfs_growfs`.
- `xfs_logprint` — drukuje zawartość dziennika systemu plików XFS. Polecenie to jest zwykle używane w procesie diagnozowania błędów systemu plików.
- `xfs_mkfile` — tworzy jeden lub więcej plików o określonych rozmiarach.
- `xfs_ncheck` — drukuje listę i-węzłów oraz ścieżek (względem punktu montowania) dostępu do plików, do których te i-węzły się odnoszą.
- `xfs_repair` — umożliwia naprawę uszkodzeń i niespójności systemu plików XFS wykrytych przez program `xfs_check`.
- `xfs_rtcp` — kopiuje jeden lub więcej plików do sekcji czasu rzeczywistego systemu plików XFS.

Dodawanie do systemu operacyjnego obsługi systemu plików XFS

Wiemy już, że obsługa systemu plików XFS nie jest na razie dostępna w żadnej ze „sklepowych” dystrybucji systemu Linux, z których miałem okazję korzystać. Na szczęście możliwość dodawania do systemu operacyjnego nowych usług systemowych i aplikacji to jedna z najbardziej fascynujących cech systemu Linux.

Bieżący podrozdział zawiera podręczną instrukcję kompilacji i instalacji jądra z obsługą systemów plików XFS, oraz informacje dotyczące instalacji, aktywacji, korzystania i administrowania samym systemem plików XFS.

Kompilacja jądra z obsługą systemu plików XFS

Dołączona do książki płyta CD-ROM zawiera kod źródłowy wersji 2.4.9 jądra systemu Linux, najbardziej zaawansowanej wersję jądra dostępną w czasie pisania tej książki. Wersję tę wybrałem, aby zademonstrować łatwość dodania do systemu Linux obsługi systemu plików XFS. Płyta CD-ROM zawiera również łaty programowe dla jądra, niezbędne do obsługi systemu plików XFS.

W niemal każdym rozdziale powtarzam, że jeżeli nigdy wcześniej nie kompilowałeś jądra, masz teraz niepowtarzalną okazję do zdobycia ogromnego doświadczenia. Jakkolwiek koncepcja pakietowej instalacji Linuksa ma na celu uniknięcie konieczności samodzielnego „wyrabiania" własnego jądra i narzędzi systemowych, niemniej kompilacja jądra jest znakomitym sposobem zapoznania się z koncepcją budowy i organizacji systemu Linux. Postępowanie zgodnie z zamieszczonymi tu instrukcjami powoduje utworzenie jądra posiadającego własną nazwę i przez to nie nadpisującego aktualnie wykorzystywanej wersji jądra systemu Linux.

Instalacja łat z obsługą systemu plików XFS dla jądra 2.4.9

Dodatek A, zatytułowany „Korzystanie z dołączonej płyty CD-ROM: instalacja, aktualizacja i kompilacja jądra", zawiera szczegółowe informacje na temat instalacji kodu źródłowego jądra systemu Linux i wyboru odpowiednich opcji konfiguracyjnych jądra. Tutaj zamieściłem jedynie skrócone instrukcje instalacji konkretnej wersji jądra, która będzie wykorzystywana w przykładach prezentowanych w dalszej części rozdziału. Procedury opisane w tym punkcie można zastosować podczas dodawania obsługi systemu plików XFS do innych wersji jądra systemu Linux, czy to do wersji najnowszej, czy też po prostu do wersji wykorzystywanej aktualnie w Twoim systemie komputerowym.

Jeżeli próbowałeś już kompilować jądro systemu Linux i chciałbyś zainstalować i uaktywnić obsługę systemu XFS w najnowszym jądrze Linuksa, możesz pobrać ostatnią wersję jądra z centralnego repozytorium źródeł systemu Linux, pod adresem *http://www.kernel.org*. Następnie powinieneś sprawdzić stronę WWW projektu ReiserFS (*http://oss.sgi.com/projects/xfs/download*) i sprawdzić, czy dostępne są już łaty dla pobranej wersji jądra systemu Linux. Pewnego dnia system XFS będzie standardowo częścią jądra systemu Linux — i to będzie to, na co wszyscy czekamy!

Aby dodać obsługę systemu XFS do jądra Linuksa w wersji 2.4.9 (i umożliwić sobie śledzenie przykładów prezentowanych w dalszej części rozdziału), należy po instalacji podstawowej wersji jądra 2.4.9 zainstalować jeszcze odpowiednie łaty programowe. Łaty te są plikami zawierającymi opis zmian wprowadzanych do kodu źródłowego jądra. Informacje o łatach i ich instalacji znajdują się również w dodatku A.

Kod źródłowy jądra w wersji 2.4.9 umieszczony jest w podkatalogu */kernels* na dołączonej do książki płycie CD-ROM. Sposób instalacji kodu źródłowego opisany jest w podrozdziale „Instalacja kodu źródłowego jądra" w dodatku A. Aby dokonać instalacji, należy (w skrócie) wykonać następujące czynności:

1. Upewnić się, że uruchomiony jest system X Window System i wykonać polecenie `su` (aby otrzymać w oknie terminala uprawnienia użytkownika root).
2. Zmienić katalog roboczy na katalog utworzony podczas instalacji kodu źródłowego jądra systemu Linux:

```
cd /usr/src/linux-2.4.9
```

3. Zastosować polecenie `patch`, aby dołączyć do plików źródłowych łaty związane z obsługą systemu plików XFS (łaty umieszczone są w podkatalogu */patches* na dołączonej do książki płycie CD-ROM):

```
patch -p1 < /mnt/cdrom/patches/patch-2.4.9-xfs-2001-08-19
```

4. Opcja `-p1` informuje program `patch` o tym, jaką część nazwy pliku (ile kolejnych katalogów w górę) należy wziąć pod uwagę podczas lokalizacji pliku wyznaczonego do załatania.

Program `patch` wyświetla listę nazw plików poddawanych aktualizacji. Po zakończeniu działania programu jesteśmy gotowi do konfiguracji jądra uwzględniającej obsługę sytemu plików XFS.

Aktywacja obsługi systemu XFS w konfiguracji jądra

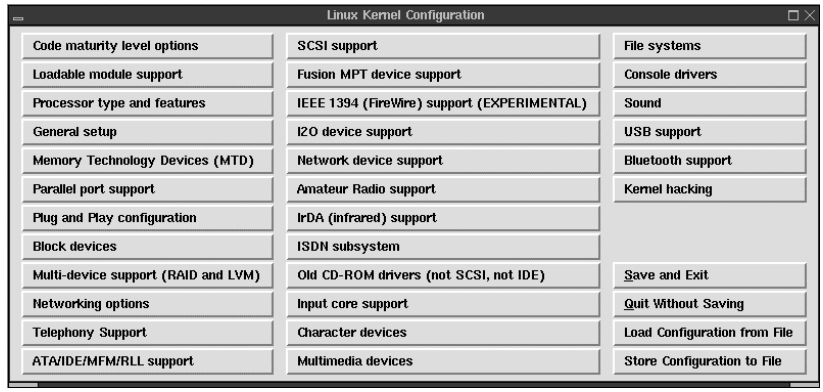
Zaprezentowany zostanie teraz sposób uaktywnienia obsługi systemu plików XFS w kodzie źródłowym jądra 2.4.9. Szerszy przegląd kwestii związanych z kompilacją nowoczesnego jądra systemu Linux i aktywacja obsługi najrozmaitszych funkcji i urządzeń fizycznych znajduje się w podrozdziale „Konfiguracja jądra" w dodatku A.

Dobrym pomysłem jest włączenie do jądra obsługi wszystkich tych systemów plików, z których regularnie korzystamy. Systemy plików wykorzystywane okazjonalnie (takie jak AppleTalk i inne sieciowe systemy plików) mogą zostać skompilowane jako moduły, co pozwoli zmniejszyć rozmiar jądra. Moduły te są w razie potrzeby automatycznie ładowane, nie stanowią jednak części pliku wykonywalnego jądra.

Jeżeli kompilowałeś już jądro dla systemu komputerowego, na którym przeprowadzasz teraz kompilację nowego jądra, możesz skorzystać z informacji konfiguracyjnych pozostałych po poprzedniej kompilacji, oszczędzając czas potrzebny na ponowną konfigurację nowego jądra. Sposób ponownego wykorzystania informacji konfiguracyjnych aktualnego jądra przedstawiony został w dodatku A.

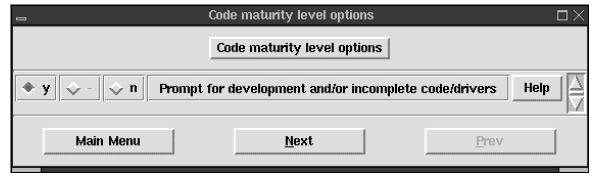
Poniżej znajduje się lista opcji charakterystycznych dla obsługi systemu plików XFS w kompilowanym jądrze. Sposób konfiguracji jądra demonstrowany jest na przykładzie programu konfiguracyjnego jądra systemu Linux, działającego w środowisku X Window System:

1. Zaloguj się jako `root` lub wykonaj polecenie `su` tak, aby uzyskać uprawnienia użytkownika *root*.
2. W oknie terminala w systemie X Window System przejdź do katalogu */usr/src/linux-2.4.9* i wykonaj polecenie `make xconfig`. Wyświetlone zostaną polecenia związane z kompilacją i załadowaniem okna programu konfiguracji jądra. Ostatecznie wyświetlone zostanie okno konfiguratora jądra systemu Linux (rysunek 7.4).



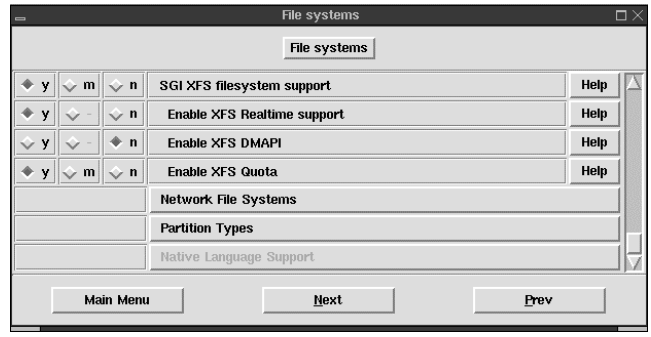
Rysunek 7.4. *Program konfiguracji jądra Linuksa działający w środowisku X Window System*

3. Naciśnij przycisk z etykietą *Code maturity level options*. Wyświetlone zostanie nowe okno dialogowe, prezentowane na rysunku 7.5.



Rysunek 7.5. *Okno dialogowe Code maturity level options*

4. Kliknij przycisk *y* obok opcji *Prompt for development and/or incomplete code/drivers*, a następnie kliknij przycisk *Main Menu*, aby wrócić do głównego okna programu konfiguracyjnego.
5. Naciśnij przycisk *File systems*. Wyświetlone zostanie nowe okno dialogowe (patrz rysunek 7.6).



Rysunek 7.6. *Okno dialogowe opcji konfiguracji systemów plików — File systems*

6. Przewiń zawartość okna do pozycji *Page buffer support* i kliknij znajdujący się obok przycisk *y*; uaktywni to pamięć buforową, stosowaną przez system XFS i wykorzystującą system buforów stron systemu Linux. Pamięć buforowa zwiększa wydajność systemu XFS przez grupowanie operacji zapisu.
7. Naciśnij przycisk *y* obok pozycji *SGI XFS filesystem support*.
8. Naciśnij przycisk *y* obok pozycji *Enable XFS Realtime support*. To opcjonalna funkcja systemu plików XFS umożliwiająca zarezerwowanie części tego systemu plików jako podwoluminu czasu rzeczywistego — na użytek plików wymagających wysokiej przepustowości operacji wejścia-wyjścia, takich jak strumieniowe pliki multimedialne. Funkcja ta nie jest jeszcze — ale już wkrótce będzie — zaimplementowana w systemie Linux. Warto wyrobić sobie nawyk zaznaczania tej opcji.
9. Naciśnij przycisk *n* obok pozycji *Enable XFS DMAPI*, chyba że zamierzasz korzystać z urządzeń systemu pamięci wielopoziomowej do wykonywania kopii zapasowej swoich danych.
10. Naciśnij przycisk *y* obok pozycji *Enable XFS Quota*. Umożliwi to stosowanie kontyngentów dyskowych dla użytkowników systemu plików XFS. Informacje o kontyngentach dyskowych przechowywane są w metadanych systemu plików XFS.
11. Naciśnij przycisk *Main Menu*, aby powrócić do głównego okna programu konfiguracyjnego.
12. Po zaznaczeniu dowolnych innych opcji konfiguracyjnych istotnych dla działania nowego jądra systemu (na przykład uwzględniających obsługę sprzętu zainstalowanego w Twoim komputerze) zapisz nową konfigurację jądra.
13. Jeżeli korzystałeś wcześniej z opcji wczytania istniejącej konfiguracji jądra, wybierz opcję *Store Configuration to File* — wyświetlone zostanie okno dialogowe umożliwiające zapisanie w sposób jawny nowej konfiguracji jądra w pliku o nazwie *.config*, umieszczonym w głównym katalogu konfigurowanego jądra.
14. Jeżeli konfigurowałeś jądro od nowa (bez wczytywania danych konfiguracyjnych innego jądra), wystarczy, że naciśniesz przycisk *Save and Exit*.
15. Kliknij przycisk *OK* potwierdzający zakończenie konfiguracji i wróć do znaku zachęty powłoki systemu Linux.

Kompilacja i instalacja jądra z obsługą systemu XFS

Po zainstalowaniu łat programowych i dokonaniu konfiguracji jądra 2.4.9 jesteśmy gotowi do jego kompilacji i instalacji w systemie Linux. Wszystkie informacje niezbędne w procesie kompilacji i instalacji jądra znajdują się w dodatku A. Tutaj prezentowana jest jedynie lista kontrolna czynności wymaganych przez ten proces.

Otóż po zainstalowaniu kodu źródłowego, załataniu i skonfigurowaniu jądra 2.4.9 z obsługą systemu plików XFS należy wykonać poniższe czynności:

1. Wpisać do pliku *Makefile* specjalną nazwę dla nowego jądra, ułatwiającą jego identyfikację i wskazującą na jego specjalne funkcje. Zazwyczaj jądra obsługujące różne systemy plików opatruję nazwami zgodnymi ze schematem `linux-<wersja-jądra>-<typ-systemu-plików>`.
2. Wykonać poniższe polecenia `make` w podanej kolejności, umożliwiającej określenie zależności pomiędzy modułami oraz umożliwiającej kompilację i instalację nowego jądra i wszelkich modułów ładowalnych:
   - `make dep`
   - `make`
   - `make modules`
   - `make install`
   - `make modules_install`
3. Sugeruję wykonanie powyższych czynności w postaci oddzielnych poleceń; ułatwi to wykrycie ewentualnych błędów kompilacji lub instalacji. Odważniejsi mogą zredukować te czynności do pojedynczego wiersza polecenia, ale mądrze byłoby w takim przypadku zostawić sobie możliwość przejrzenia wyników kompilacji dla upewnienia się o braku błędów:

   ```
   make dep; make install; make modules; make modules_install
   ```
4. Zmodyfikować pliki konfiguracyjne procesu ładowania systemu operacyjnego zgodnie z opisem w punkcie „Ładowanie nowego jądra" dodatku A i w razie potrzeby uruchomić program ładujący `lilo`.

Jesteśmy już prawie gotowi do przeładowania systemu i uruchomienia nowego jądra z obsługą systemu plików XFS. Jednak zanim tego dokonamy, należałoby skompilować i zainstalować oprogramowanie narzędziowe umożliwiające montowanie, wykorzystywanie i administrowanie systemem plików XFS. Czynność instalacji oprogramowania opisana jest niżej.

Instalacja oprogramowania narzędziowego

Niniejszy podrozdział omawia sposób konfiguracji, kompilacji i instalacji rozmaitych programów narzędziowych umożliwiających administrację i korzystanie z systemu plików XFS w systemie Linux. Programy te obsługują wszystkie podstawowe funkcje systemu plików jako takiego, w tym tworzenie systemu plików, jego naprawę, zmianę rozmiaru oraz usuwanie, ponadto umożliwiają skorzystanie z funkcji zaawansowanych, takich jak listy kontroli dostępu czy rozszerzone atrybuty plików i katalogów.

Kompilacja i instalacja pakietu xfsprogs

Instalacja pakietu `xfsprogs` (umieszczonego na dołączonej do książki płycie CD-ROM) w wersji 1.3.5 lub nowszej wymaga wykonania następujących czynności:

1. Zamontowania płyty CD i rozpakowania pakietu `xfsprogs` z archiwum w pliku */xfs/xfsprogs-1.3.5.src.tar.gz* do wybranego katalogu za pośrednictwem polecenia podobnego do poniższego:

W przeciwieństwie do pakietów narzędziowych przeznaczonych do obsługi wielu innych systemów plików omawianych w tej książce, niektóre programy dla systemu XFS korzystają z plików nagłówkowych i bibliotek będących częścią innych pakietów dla systemu XFS. Dlatego też ważna jest instalacja tych pakietów w odpowiedniej kolejności. Jeżeli skrypt konfiguracyjny lub proces `make` zgłosi błąd podczas konfiguracji lub kompilacji któregoś z programów omawianych w tym rozdziale, należy koniecznie sprawdzić, czy kolejność instalacji pakietów odpowiada kolejności prezentowanej w książce i czy po instalacji każdego z pakietów wykonano polecenie `make install-dev`, instalujące publiczne wersje plików nagłówkowych i bibliotek wykorzystywanych przez nowo zainstalowane programy narzędziowe systemu XFS.

```
cd <katalog-docelowy>
tar zxvf <ścieżka-dostępu-do-CD>/xfs/xfsprogs-1.3.5.src.tar.gz
```

2. Zmiany katalogu roboczego na *<katalog-docelowy>/xfsprogs-1.3.5* i uruchomienia skryptu dokonującego konfiguracji pliku *Makefile* odpowiedniej dla Twojego systemu i stosowanej przez Ciebie dystrybucji:

   ```
   ./configure
   ```

   Skrypt ten wyświetla mnóstwo informacji, których powielanie w tym miejscu jest niepotrzebne.

3. Wykonania (po zakończeniu procesu konfiguracji pliku *Makefile*) polecenia `make`, inicjującego kompilację nowej wersji programów składowych pakietu `xfsprogs`:

   ```
   make
   ```

   Proces kompilacji również wygeneruje znaczną ilość komunikatów, których prezentowanie w tym miejscu jest zbędne.

4. Instalacji skompilowanych programów w systemie za pomocą poniższych poleceń:

   ```
   make install
   make install-dev
   ```

W systemie zainstalowana zostanie nowa wersja pakietu `xfsprogs` i stosowne pliki dokumentacji *man*.

Kompilacja i instalacja biblioteki XFS DMAPI

Aby zainstalować bibliotekę XFS DMAPI (umieszczoną na dołączonej do książki płycie CD-ROM) w wersji 0.2.2, należy wykonać następujące czynności:

1. Zamontować płytę CD, rozpakować pakiet `dmapi` z archiwum w pliku */xfs/dmapi-0.2.2.src.tar.gz* i umieścić ten pakiet w wybranym katalogu:

   ```
   cd <katalog-docelowy>
   tar zxvf <ścieżka-dostępu-do-CD>/xfs/dmapi-0.2.2.src.tar.gz
   ```

2. Zmienić katalog roboczy na *<katalog-docelowy>/dmapi-0.2.2* i uruchomić skrypt konfiguracji pakietu `dmapi`, dostosowujący zawartość pliku *Makefile* do Twojego systemu i stosowanej przez Ciebie dystrybucji:

   ```
   ./configure
   ```

Skrypt ten wyświetla mnóstwo informacji, których powielanie w tym miejscu jest niepotrzebne.

3. Wykonać (po zakończeniu procesu konfiguracji pliku *Makefile*) polecenie `make`, inicjujące kompilację nowych wersji składników biblioteki XFS DMAPI:

```
make
```

Proces kompilacji wygeneruje znaczną ilość komunikatów, których prezentacja w tym miejscu również byłaby zbędna.

4. Zainstalować w systemie skompilowane składniki biblioteki za pomocą poniższych poleceń:

```
make install
make install-dev
```

W systemie zainstalowana zostanie nowa wersja biblioteki XFS DMAPI i stosowne pliki dokumentacji *man*.

Kompilacja i instalacja pakietu acl

Instalacja pakietu `acl` (umieszczonego na dołączonej do książki płycie CD-ROM) w wersji 1.1.2 lub nowszej, polega na wykonaniu następujących czynności:

1. Zamontowaniu płyty CD i rozpakowaniu pakietu `acl` z archiwum w pliku */xfs/acl-1.1.2.src.tar.gz* do wybranego katalogu za pośrednictwem polecenia podobnego do poniższego:

```
cd <katalog-docelowy>
tar zxvf <ścieżka-dostępu-do-CD>/xfs/acl-1.1.2.src.tar.gz
```

2. Zmianie katalogu roboczego na *<katalog-docelowy>/acl-1.1.2* i uruchomieniu skryptu konfiguracji. Plik *Makefile* zostanie uzupełniony o informacje niezbędne do przeprowadzenia kompilacji w Twoim systemie:

```
./configure
```

Skrypt ten wyświetla mnóstwo informacji, których powielanie w tym miejscu jest niepotrzebne.

3. Wykonaniu polecenia `make`, inicjującego kompilację nowej wersji programów składowych pakietu `acl`:

```
make
```

Proces kompilacji generuje znaczną ilość komunikatów, których prezentowanie w tym miejscu również jest niepotrzebne.

4. Instalacji skompilowanych programów w systemie za pomocą poniższych poleceń:

```
make install
make install-dev
```

Efektem tych poczynań jest uaktualnienie systemu o nową wersję pakietu `acl` i stosowne pliki dokumentacji *man*.

Kompilacja i instalacja pakietu attr

Instalacja pakietu `attr` (umieszczonego na dołączonej do książki płycie CD-ROM) w wersji 1.1.3, wymaga wykonania następujących czynności:

1. Zamontowania płyty CD i rozpakowania pakietu `attr` z archiwum w pliku */xfs/attr-1.1.3.src.tar.gz* do wybranego katalogu:

   ```
   cd <katalog-docelowy>
   tar zxvf <ścieżka-dostępu-do-CD>/xfs/attr-1.1.3.src.tar.gz
   ```

2. Zmiany katalogu roboczy na *<katalog-docelowy>/attr-1.1.3* i uruchomienia skryptu konfiguracji pakietu `attr`. Skrypt ten uzupełni plik *Makefile* o informacje niezbędne do przeprowadzenia kompilacji w Twoim systemie:

   ```
   ./configure
   ```

   Skrypt wyświetli szereg niepotrzebnych nam w tej chwili informacji.

3. Wykonania (po zakończeniu procesu konfiguracji pliku *Makefile*) polecenia `make`, inicjującego kompilację nowej wersji programów składowych pakietu `attr`:

   ```
   make
   ```

   Proces kompilacji również wygeneruje znaczną ilość komunikatów, których prezentacja w tym miejscu byłaby zbędna.

4. Zainstalowania świeżo skompilowanych programów w systemie za pomocą poniższych poleceń:

   ```
   make install
   make install-dev
   ```

W systemie zainstalowana zostanie nowa wersja pakietu `attr` i stosowne pliki dokumentacji *man*.

Kompilacja i instalacja pakietu xfsdump

Instalacja pakietu `xfsdump` (umieszczonego na dołączonej do książki płycie CD-ROM) w wersji 1.1.3, wymaga wykonania następujących czynności:

1. Zamontowania płyty CD i rozpakowania pakietu `xfsdump` z archiwum w pliku */xfs/xfsdump-1.1.3.src.tar.gz* do wybranego katalogu za pośrednictwem polecenia podobnego do poniższego:

   ```
   cd <katalog-docelowy>
   tar zxvf <ścieżka-dostępu-do-CD>/xfs/xfsdump-1.1.3.src.tar.gz
   ```

2. Zmiany katalogu roboczego na *<katalog-docelowy>/xfsdump-1.1.3* i uruchomienia skryptu konfiguracji. Plik *Makefile* zostanie uzupełniony o informacje niezbędne do przeprowadzenia kompilacji w Twoim systemie:

   ```
   ./configure
   ```

   Skrypt ten wyświetli mnóstwo informacji, których powielanie w tym miejscu jest niepotrzebne.

3. Wykonania polecenia `make`, inicjującego kompilację nowej wersji programów składowych pakietu `xfsdump`:

   ```
   make
   ```

   Proces kompilacji wygeneruje nieprzydatne nam w tej chwili komunikaty.

4. Instalacji skompilowanych programów w systemie za pomocą poniższych poleceń:

   ```
   make install
   make install-dev
   ```

Wykonanie powyższych czynności zaowocuje uaktualnieniem systemu o nową wersję pakietu `acl` i stosowne pliki dokumentacji *man*. Teraz możesz przeładować system; przed rozpoczęciem dalszych eksperymentów z systemem plików XFS koniecznie upewnij się, że podczas inicjalizacji systemu załadowane zostało odpowiednie (przygotowane do obsługi systemu XFS) jądro Linuksa.

Administracja systemem plików XFS

Znajomość kwestii projektowych odnoszących się do rozmaitych systemów plików i funkcji przez nie oferowanych uważam za niezwykle istotną. Ten rodzaj wiedzy gwarantuje prawidłowy wybór systemu plików do różnych zastosowań, odpowiadający raz wymogom domowego stanowiska pracy, innym zaś razem szkolnej sieci komputerowej czy korporacyjnego systemu informatycznego; przydaje się też podczas pogawędek przy piwie. Tym niemniej prawdziwy (i najciekawszy) test rozbudowanego systemu plików ma miejsce podczas jego codziennego wykorzystywania.

System plików XFS firmy SGI posiada niezwykle bogate oprogramowanie narzędziowe, liczebnością przewyższające zestawy narzędziowe wszystkich innych systemów plików z kroniką omawianych w tej książce; oprogramowanie to w przypadku systemu XFS umożliwia przeprowadzanie niemal dowolnych operacji na systemie plików. Jest to po części efektem rodowodu systemu XFS, używanego przez wiele lat w stacjach roboczych firmy SGI. W ciągu tych lat programiści firmy SGI i jej klienci stale ulepszali i rozszerzali funkcje programów narzędziowych; w niektórych przypadkach rozwój ten zaowocował mnóstwem (jak przystało na prawdziwy program uniksowy) opcji wiersza polecenia, w innych — implementacją prostych narzędzi, potrzebnych do codziennej pracy z systemem XFS i koniecznych do rozwiązywania ewentualnych problemów.

Nikt nie wie lepiej, jak szybko pojawiają się proste i niezawodne narzędzia zaspokajające zauważone nowe potrzeby, niż użytkownik systemu Linux — w ten sposób narodził się przecież cały system. Dzięki opublikowaniu przez firmę SGI kodu źródłowego systemu XFS wszyscy możemy korzystać z owoców pracy tej firmy. Nie znaczy to, że w systemie XFS nie można już nic poprawić — dalszy rozwój i ulepszanie systemu XFS ma miejsce stale, również w chwili, gdy czytasz te słowa. W systemie Linux przydałaby się na przykład obsługa podwoluminów czasu rzeczywistego, zaimplementowana w systemach XFS działających pod kontrolą systemu operacyjnego IRIX. Niemniej, jak przekonasz się w czasie lektury tego podrozdziału, system XFS posiada i tak znaczną przewagę — w tym pod względem funkcjonalności oprogramowania narzędziowego — nad innymi systemami plików omawianymi w niniejszej książce.

Tworzenie systemu plików XFS

Podobnie jak większość pozostałych nowoczesnych systemów plików dla Linuksa, program tworzący system plików XFS uruchamiany jest za pośrednictwem programu-otoczki `/sbin/mkfs` wywołanego z opcją `-t xfs`. Powoduje to uruchomienie programu `/sbin/mkfs.xfs` i przekazanie do niego wszelkich parametrów określonych w wierszu polecenia programu `/sbin/mkfs`.

W prezentowanym niżej przykładzie program `mkfs.xfs` uruchamiany jest bezpośrednio, w celu wyróżnienia nazwy programu, który rzeczywiście realizuje operacje związane z utworzeniem systemu plików (i nazwy, którą należy podać jako parametr polecenia `man`, aby wyświetlić dokumentację programu `mkfs.xfs`). Poprzez wiersz polecenia do programu `mkfs.xfs` przekazywane są parametry, które umożliwiają dopasowanie właściwości systemu plików do własnych potrzeb; w poniższym przykładzie program został jednak wywołany w swojej najprostszej postaci:

```
# mkfs.xfs /dev/hdc2
meta-data=/dev/hdc2              isize=256    agcount=8, agsize=65552 blks
data     =                       bsize=4096   blocks=524412, imaxpct=25
         =                       sunit=0      swidth=0 blks, unwritten=0
naming   =version 2              bsize=4096
log      =internal log           bsize=4096   blocks=1200
realtime =none                   extsz=65536  blocks=0, rtextents=0
```

Jeżeli tworzysz system plików XFS na partycji lub dysku, na którym istnieje już inny typ systemu plików, powinieneś skorzystać z opcji `-f`, wymuszającej nadpisanie istniejącego systemu plików.

Program `mkfs.xfs` jest wyjątkowo szybki, ponieważ nie musi wykonywać wielu czasochłonnych czynności wymaganych przez systemy plików, takie jak ext2 (odpada między innymi wstępna alokacja struktur systemu plików, np. i-węzłów). Informacje wygenerowane w czasie działania programu `mkfs.xfs` opisują rozmaite sekcje nowo utworzonego systemu plików XFS.

Sekcja parametrów metadanych systemu plików (`meta-data`):

- `isize` (rozmiar i-węzła) — 256 bajtów (wartość domyślna).
- `agcount` (liczba grup alokacji) — 8 (wartość domyślna dla systemów o rozmiarach z zakresu od 128 MB do 8 GB).
- `agsize` (rozmiar grupy alokacji) — obliczany na podstawie liczby grup alokacji i rozmiaru dysku lub partycji, na której tworzony jest system plików. Grupy alokacji mogą przyjmować rozmiary z zakresu od 16 MB do niecałych 4 GB.

Sekcja parametrów danych systemu plików (`data`):

- `bsize` (rozmiar bloku) — 4096 bajtów (wartość domyślna dla systemów z 4-kilobajtowym rozmiarem strony, takich jak Linux).
- `blocks` (całkowita liczba bloków) — obliczana na podstawie rozmiaru bloku i rozmiaru dysku lub partycji, na której tworzony jest system plików,

z wyłączeniem miejsca zajmowanego przez dziennik, opcjonalne sekcje czasu rzeczywistego oraz miejsca zajmowanego przez struktury wewnętrzne grup alokacji, sam system plików itp.

- `imaxpct` (maksymalny procentowy obszar systemu plików zajmowany przez i-węzły) — 25 procent (wartość domyślna).
- `sunit` (z ang. *stripe unit*, rozmiar jednostki przeplotu danych na dysku lub partycji) — 0 (domyślnie przeplatanie jest wyłączone).
- `swidth` (z ang. *stripe width*, szerokość wstęgi przeplotu) — 0 bloków (domyślnie przeplatanie jest wyłączone).
- `unwritten` (wartość logiczna wskazująca, czy obszary niezapisane mają być oznaczane) — 0 (wartość domyślna — funkcja ta nie jest jeszcze zaimplementowana dla systemu Linux).

Sekcja parametrów katalogów (`naming`):

- System XFS obsługuje dwa rodzaje katalogów: wersję 1. i wersję 2. Katalogi wersji 2. to domyślny typ katalogów zalecany dla systemu Linux ze względu na sposób, w jaki funkcja pobierania wpisów katalogu (`getdents()`) biblioteki `glibc` współpracuje z jądrem systemu Linux.
- `bsize` (rozmiar bloku katalogu) — 4096 bajtów (wartość domyślna).

Sekcja parametrów dziennika (`log`):

- System plików XFS może przechowywać dziennik transakcji wewnętrznie (wewnątrz systemu plików, do którego przypisany jest dziennik) lub zewnętrznie.
- `bsize` (rozmiar bloku dziennika) — 4096 bajtów (wartość domyślna).
- `blocks` (całkowita liczba bloków w dzienniku) — obliczana na podstawie rozmiaru tworzonego systemu plików.

Sekcja parametrów obszarów czasu rzeczywistego (`real-time`):

System plików XFS może zawierać opcjonalną sekcję czasu rzeczywistego, w której przechowywane są dane plików wymagających realizacji operacji wejścia-wyjścia z określoną przepustowością. Sekcja czasu rzeczywistego systemu plików XFS podzielona jest na szereg obszarów o stałym rozmiarze.

Jak już wspomniałem, podwoluminy czasu rzeczywistego nie są jeszcze w pełni obsługiwane w systemie Linux i nie powinny być stosowane.

- `extsz` (rozmiar obszaru) — 65 536 bajtów (wartość domyślna).
- `blocks` (całkowita liczba bloków przydzielonych do podwoluminu czasu rzeczywistego) — 0 (wartość domyślna w przypadku, kiedy wolumin czasu rzeczywistego nie jest stosowany).
- `rtextents` (liczba obszarów czasu rzeczywistego) — 0 (wartość domyślna w przypadku, kiedy wolumin czasu rzeczywistego nie jest stosowany).

Podczas migracji z istniejącego systemu plików do systemu plików XFS bardzo przydatny jest program `xfs_estimate`, będący świetnym przykładem bogactwa kolekcji programów narzędziowych dla systemu XFS. Program ten analizuje istniejący system plików (bez wykraczania poza punkt montowania) i określa szacunkowy rozmiar tego systemu plików po przeniesieniu go do systemu XFS. Za pomocą opcji `-b` (rozmiar bloku) możemy analizować rezultaty konwersji istniejącego systemu plików do systemu XFS przy różnych rozmiarach bloków, z wykorzystaniem dziennika wewnętrznego lub zewnętrznego, przy różnych rozmiarach dziennika itd.

Eksperymentowanie z różnymi rozmiarami bloków jest nieco nużące, ale ustalenie z pomocą narzędzia `xfs_estimate` optymalnego rozmiaru bloku może zaowocować znaczącą oszczędnością miejsca i systemem plików najlepiej dostosowanym do typów i rozmiarów przechowywanych w nim plików. Znakomitymi przykładami takich systemów plików są systemy przechowujące kolejki serwera pocztowego, list dyskusyjnych itp.

Polecenie `mkfs.xfs` obsługuje wiele opcji dających operatorowi możliwość sterowania procesem zakładania systemu plików XFS. Kolejne trzy punkty poświęcone są niektórym z tych opcji i prezentują najpopularniejsze sposoby dostosowania tworzonego systemu plików XFS do własnych potrzeb.

Określanie rozmiaru dziennika systemu XFS

Jednym z najczęściej modyfikowanych parametrów tworzonego systemu plików XFS jest rozmiar dziennika rejestrującego aktualizacje metadanych systemu plików. Domyślnie rozmiar dziennika jest obliczany podczas tworzenia systemu plików i jest pochodną rozmiaru systemu plików. Jednakże niekiedy może zajść potrzeba zwiększenia rozmiaru dziennika, na przykład na potrzeby aplikacji wykonujących częste i długotrwałe operacje na systemie plików lub w przypadku systemu plików intensywnie używanego przez wielu użytkowników.

Określenie rozmiaru dziennika systemu plików XFS wymaga zastosowania opcji `-l size=`*`rozmiar`*, gdzie `rozmiar` jest rozmiarem dziennika. Rozmiar ten może być określony na kilka sposobów:

- w bajtach — rozmiar jest wtedy podawany bez dodatkowych oznaczeń;
- w blokach (rozmiar dziennika zależny będzie od rozmiaru bloku) — przez umieszczenie za liczbą litery `b`;
- w kilobajtach — przez umieszczenie za liczbą litery `k`;
- w megabajtach — przez umieszczenie za liczbą litery `m`;
- w gigabajtach — przez umieszczenie za liczbą litery `g`.

Osoby biegle posługujące się systemami szesnastkowymi lub ósemkowymi mogą podać rozmiar w jednym z tych systemów, poprzedzając liczbę przedrostkiem `0X` (zero x) dla wartości szesnastkowych lub przedrostkiem `0` (wielka litera O) dla wartości ósemkowych.

Tak więc utworzenie systemu plików XFS na urządzeniu `/dev/hdc2` z dziennikiem o rozmiarze 10 MB wymaga od osób nie obdarzonych umiejętnością operowania zapisem ósemkowym i szesnastkowym liczb i szesnastkowo wydania następującego polecenia:

```
# mkfs.xfs -f -l size=10m /dev/hdc2
```

Spowoduje to wyświetlenie następujących informacji:

```
meta-data=/dev/hdc2              isize=256    agcount=8, agsize=65552 blks
data     =                       bsize=4096   blocks=524412, imaxpct=25
         =                       sunit=0      swidth=0 blks, unwritten=0
naming   =version 2              bsize=4096
log      =internal log           bsize=4096   blocks=2560
realtime =none                   extsz=65536  blocks=0, rtextents=0
```

Dziennik jest nadal przechowywany wewnątrz systemu plików, składa się teraz jednak z 2 560 bloków po 4 kB, co daje razem 10 MB.

Określanie położenia dziennika

Stosunkowo często pojawia się również potrzeba zmiany położenia dziennika systemu plików. Standardowo dziennik systemu plików XFS przechowywany jest wewnątrz tego systemu i zajmuje oddzielną sekcję systemu plików. Potrzeba określenia alternatywnego położenia dziennika danego systemu plików XFS może mieć rozmaite przyczyny, jak choćby chęć zwiększenia współbieżności operacji zapisu w dzienniku i w samym systemie plików. Przemieszczenie dziennika systemu plików XFS do innej partycji na tym samym dysku będzie w tym przypadku stratą czasu, ponieważ zapisy w dzienniku i w systemie plików będą wciąż angażowały te same głowice, obsługujące ten sam dysk; jednakże ulokowanie dziennika jednego lub kilku systemów plików XFS na osobnym dysku lub jego partycji może znacząco zwiększyć wydajność systemu plików, umożliwiając równoczesne rejestrowanie zmian metadanych i aktualizacje samego systemu plików. Przyspieszenie może być szczególnie widoczne w przypadku intensywnie wykorzystywanych systemów plików.

Aby określić położenie dziennika rejestrującego zmiany systemu plików XFS, należy skorzystać z opcji `-l logdev=`*`urządzenie`*, gdzie *`urządzenie`* jest nazwą partycji, na której chcemy umieścić dziennik tworzonego systemu plików.

Tak więc, aby utworzyć system plików XFS na partycji `/dev/hdc2` i umieścić dziennik o wielkości 10 MB na partycji `/dev/hdb1`, należy wykonać następujące polecenie:

```
# mkfs.xfs -f -l logdev=/dev/hdb1 /dev/hdc2
```

Spowoduje to wygenerowanie wydruku podobnego do przedstawionego poniżej:

```
meta-data=/dev/hdc2              isize=256    agcount=8, agsize=65552 blks
data     =                       bsize=4096   blocks=524412, imaxpct=25
         =                       sunit=0      swidth=0 blks, unwritten=0
naming   =version 2              bsize=4096
log      =/dev/hdb1              bsize=4096   blocks=2560
realtime =none                   extsz=65536  blocks=0, rtextents=0
```

Widzimy, że dziennik został umieszczony na partycji `/dev/hdb1`, na której przydzielono do dziennika 2 560 bloków po 4 kB każdy.

Określanie rozmiaru bloku

Tworzenie systemów plików o małym rozmiarze bloku to standardowa metoda optymalizacji systemu plików. W systemach plików dedykowanych do określonych zadań, takich jak obsługa repozytorium poczty elektronicznej czy grup dyskusyjnych, możliwe

Podczas tworzenia dziennika na osobnej partycji należy określić zarówno lokalizację dziennika, jak i jego rozmiar, ewentualnie podać partycję, której rozmiar odpowiada pożądanemu rozmiarowi dziennika. Maksymalny rozmiar dziennika systemu XFS to 65 536 bloków. Jeżeli jako partycję docelową podamy partycję większą niż 65 536 bloków i nie określimy żądanego rozmiaru dziennika, program `mkfs.xfs` zakończy działanie zgłoszeniem komunikatu o błędzie

```
log size XXXXXX blocks too large, maximum size is 65536 blocks
```

poprzedzonym informacją o sposobie korzystania z programu `mkfs.xfs`.

jest zwykle przewidzenie średniego rozmiaru plików i zastosowanie mniejszych (czy większych) niż domyślne bloków w celu zminimalizowania ilości miejsca marnowanego przez pliki o rozmiarach mniejszych od rozmiarów bloków. Pliki tekstowe zawierające artykuły do grup dyskusyjnych są zwykle niewielkie — zastosowanie domyślnego rozmiaru bloku wynoszącego 4 kB w systemie plików przechowującym głównie artykuły grup dyskusyjnych może spowodować marnotrawstwo znacznej ilości miejsca.

Niektóre systemy plików z kroniką, w szczególności system ReiserFS, wykorzystują skomplikowane algorytmy w rodzaju mechanizmu *tail-packing*, grupującego „końcówki" plików i przechowującego je we wspólnych blokach, co pozwala zmniejszyć stopień marnotrawstwa. Mechanizm alokacji obszarów stosowany w systemie XFS nie jest przystosowany do tego rodzaju algorytmów, ponieważ jego zadaniem jest maksymalizacja wydajności i szybkości alokacji ciągłych zbiorów bloków (obszarów), a nie pieczołowite „ciułanie" resztek plików. W przypadku systemu plików XFS wybór optymalnego rozmiaru bloku ułatwiany jest przez program `xfs_fsr` (XFS Filesystem Recognize), służący do optymalizacji organizacji dyskowej systemu plików i mechanizmu alokacji tak, aby jak najlepiej wykorzystać przestrzeń dyskową systemu plików.

Aby określić rozmiar bloku dla systemu plików XFS, należy skorzystać z opcji `-b log=`*`rozmiar`* lub `-b size=`*`rozmiar`*, gdzie rozmiar jest rozmiarem bloku w bajtach. W przypadku opcji `-b log=`*`rozmiar`* podaje się wartość logarytmu przy podstawie 2. W przypadku opcji `-b size=`*`rozmiar`* rozmiar jest podawany bezpośrednio w bajtach.

Przykładowo, utworzenie systemu plików XFS na partycji `/dev/hdc2` przy rozmiarze bloku wynoszącym 1 kB należy wykonać jedno z następujących poleceń:

```
# mkfs.xfs -f -b log=10 /dev/hdc2
```

lub

```
# mkfs.xfs -f -b size=1024 /dev/hdc2
```

W obu przypadkach efekt będzie podobny do wydruku przedstawionego poniżej:

```
meta-data=/dev/hdc2              isize=256    agcount=8, agsize=262206 blks
data     =                       bsize=1024   blocks=2097648, imaxpct=25
         =                       sunit=0      swidth=0 blks, unwritten=0
naming   =version 2              bsize=4096
log      =/dev/hdb1              bsize=1024   blocks=3072
realtime =none                   extsz=65536  blocks=0, rtextents=0
```

Montowanie systemu plików XFS

System plików XFS (podobnie jak pozostałe systemów plików z kroniką) podczas montowania automatycznie odtwarza dziennik transakcji. Czynność ta jest podstawową gwarancją spójności systemu plików.

Program `mount` rozpoznaje całą gamę opcji charakterystycznych dla systemu XFS, z których kilka służy do nadpisania wartości domyślnych parametrów systemu plików, takich jak lokalizacja dziennika i rozmaite ustawienia przeplotu dysku. Pozostałe opcje przeznaczone są do dostrojenia wydajności i uaktywnienia funkcji specjalnych. Zwykle nie ma potrzeby śrubowania domyślnych ustawień programu `mount` dla systemu plików XFS, ale na wszelki wypadek przedstawię kilka opcji montowania systemu XFS, które mogą okazać się przydatne:

- `biosize=`*`rozmiar`* — ustawia rozmiar buforów dla buforowanych operacji wejścia-wyjścia systemu plików; podana liczba musi być wartością logarytmu o podstawie 2. Dopuszczalne wartości to 13 (8 kB — jedyna opcja dla systemów z czterokilobajtowym rozmiarem stron, takich jak Linux), 14 (16 kB), 15 (32 kB) oraz 16 (64 kB). Zmiana wartości domyślnej może okazać się konieczna w systemach plików o krytycznym znaczeniu dla ciągłości działania systemu komputerowego — w takich systemach może zajść potrzeba minimalizacji liczby buforowanych operacji wejścia-wyjścia; niemniej jednak redukcja rozmiarów operacji buforowanych oznacza zwykle spadek wydajności, spowodowany częstszymi zapisami danych na dysk.
- `dmapi` lub `xsdm` — włącza wywołania DMAPI (Data Management API lub Data Migration API), umożliwiając przechowywanie systemu plików na urządzeniach pamięci wielopoziomowej. Funkcja ta znana jest również jako XDSM (eXtended Data Storage Management) API, stąd różne nazwy opcji.
- `logbufs=`*`wartość`* — ustawia rozmiar buforów dziennika przechowywanych w pamięci. Dopuszczalne są wartości z zakresu od 2 do 8 (włącznie). Domyślną wartością jest 8 buforów w przypadku systemów plików o rozmiarze bloku 64 kB, 4 bufory w systemach o rozmiarze bloku 32 kB, 3 bufory w systemach o rozmiarze bloków 16 kB oraz 2 bufory dla wszystkich pozostałych rozmiarów bloków. Zwiększenie liczby buforów może przyczynić się do wzrostu wydajności, a to dzięki redukcji częstotliwości, z jaką bufory dziennika są zapisywane na dysk; większa liczba buforów wymaga jednak przydzielenia dodatkowej pamięci operacyjnej i zwiększa ilość danych narażonych na zagubienie w przypadku awarii komputera.
- `logbsize=`*`wartość`* — ustawia rozmiar buforów dziennika przechowywanych w pamięci. Dopuszczalne wartości to 16 kB i 32 kB. Domyślny rozmiar bufora na komputerach wyposażonych w 32 MB pamięci to 16 kB — na komputerach o większej ilości pamięci domyślny rozmiar bufora to 32 kB.
- `logdev=`*`device`* — pozwala określić zewnętrzne urządzenie jako urządzenie przechowujące dziennik systemu plików XFS.
- `noatime` — blokuje aktualizacje atrybutu czasu ostatniego dostępu do pliku podczas jego odczytu. Opcja ta pozwala na zaoszczędzenie znacznej ilości czasu,

zmniejszając liczbę aktualizacji metadanych systemu plików — oczywiście pod warunkiem, że nie są dla ciebie istotne informacje o czasie ostatniego dostępu do pliku.

- `norecovery` — umożliwia zamontowanie systemu plików bez odtwarzania transakcji zarejestrowanych w dzienniku. Opcja ta została omówiona w punkcie „Montowanie systemu plików XFS bez odtwarzania zawartości dziennika".
- `osyncisdync` — operacje zapisu do plików otwartych ze znacznikiem `O_SYNC` (tryb synchroniczny dla danych i metadanych pliku) wykonywane są tak, jak gdyby pliki te zostały otwarte ze znacznikiem `O_DSYNC` (tryb synchroniczny dla danych pliku). Opcja ta decyduje o tym, czy po każdym zapisie do pliku opróżnienie buforów plikowych będzie realizowane przez funkcję `fsync()`, czy `fdatasync()`.Niestety, w systemie Linux działanie obu tych funkcji jest identyczne. Jeżeli dysponujemy systemem z właściwie zaimplementowaną funkcją `fdatasync()`, zastosowanie tej opcji może zwiększyć wydajność dostępu do systemu plików, kosztem zwiększenia ryzyka utracenia metadanych systemu plików (takich jak aktualizacje znaczników czasowych) na skutek awarii komputera.
- `quota`, `userquota` lub `uqnoenforce` — uaktywnia kontyngenty dyskowe użytkowników.
- `grpquota` lub `gqnoenforce` — uaktywnia kontyngenty dyskowe grup użytkowników.
- `sunit=`*`wartość`* `i swidth=`*`wartość`* — umożliwia nadpisanie standardowych wartości rozmiaru wstęgi przeplotu i szerokości wstęgi przeplotu określonych podczas tworzenia systemu plików. Opcje te są użyteczne jedynie w odniesieniu do systemów plików XFS umieszczonych na sprzętowych macierzach RAID. We wszystkich pozostałych przypadkach parametry te są definiowane wyłącznie podczas tworzenia systemu plików i odczytywane później z jego superbloku.

Systemy plików XFS są zwykle montowane podczas ładowania systemu, na podstawie informacji odczytanych z pliku */etc/fstab*. W pliku tym system plików oznaczany jest jako `xfs`, ale partycja czy dysk, na którym ten system jest zainstalowany, mogą być identyfikowane na trzy sposoby:

- Przez jawne wskazanie partycji zawierającej system plików i oznaczenie jej jako partycji systemu XFS. Taki zapis może znajdować się w pliku */etc/fstab*, możliwe jest również podanie wymaganych parametrów bezpośrednio do polecenia `mount`: `mount -t xfs`.
- Przez wskazanie etykiety partycji systemu plików. Systemy plików XFS mogą być oznaczane etykietami podczas ich tworzenia za pomocą opcji `-L` polecenia `mkfs.xfs` lub za pośrednictwem opcji `-L` polecenia `xfs_admin` (w przypadku już istniejących systemów plików). Aby usunąć etykietę systemu plików należy po opcji `-L` programu `xfs_admin` podać ciąg `--` (dwa myślniki).
- Przez wskazanie uniwersalnego unikalnego identyfikatora systemu plików (UUID). Identyfikatory UUID mają postać ciągu znaków, np. c1b9d5a2-11cf-9ece-0020afc76f16 i są generowane automatycznie podczas tworzenia

systemu plików XFS. Możliwe jest również własnoręczne przypisanie identyfikatora UUID do istniejącego systemu plików — za pośrednictwem opcji `-U` *identyfikator-UUID* programu `xfs_db`.

Standardowym sposobem montowania systemu plików jest montowanie przez podanie nazwy partycji, na której system ten jest umieszczony. Przykładowo system XFS, utworzony na partycji `/dev/hdc2`, montuje się do katalogu `/xfs_test` za pomocą następującego polecenia:

```
mount -t xfs /dev/hdc2 /xfs_test
```

Rosnąca popularność montowania ze wskazaniem etykiety partycji wynika z tego, że etykiety umożliwiają odwoływanie się do konkretnych systemów plików bez konieczności wskazywania przypisanego im urządzenia fizycznego. Jest to szczególnie wygodne w przypadku systemów plików umieszczonych na urządzeniach zewnętrznych, ułatwia bowiem przenoszenie tych urządzeń pomiędzy systemami komputerowymi i montowanie ich bez konieczności sprawdzania, w jakiej kolejności podłączone są do koncentratora. Cecha ta może okazać się przydatna również w przypadku wewnętrznych napędów IDE. Możemy sobie bowiem wyobrazić sytuację, w której posiadamy systemy plików na dysku podrzędnym (ang. *slave*) drugiego kontrolera IDE. Po podłączeniu do tego kontrolera dysku nadrzędnego (ang. *master*) musielibyśmy dokonać modyfikacji zawartości pliku */etc/fstab* i zmienić nazwy partycji, na których umieszczone są interesujące nas systemy plików. Gdyby systemy te były identyfikowane przez etykietę, poprawne montowanie plików nie wymagałoby żadnych zmian.

Program `mkfs.xfs`, tworząc system plików, nie nadaje mu automatycznie etykiety. Przed odczytaniem etykiety systemu plików lub własnoręcznym jej określeniem należy system plików odmontować. Poniższy przykład ilustruje sposób określenia etykiety odmontowanej partycji systemu XFS:

```
# xfs_db -c 'label' /dev/hdc2
label = ""
```

Kolejny przykład demonstruje procedurę przypisywania wybranej etykiety do systemu plików umieszczonego na partycji `/dev/hdc2`:

```
# xfs_admin -L /xfs_test /dev/hdc2
writing all SBs
new label = "xfs_test"
```

Przy założeniu, że systemowi plików umieszczonemu na partycji `/dev/hdc2` przypisano etykietę „`/xfs_test`", poniższe wpisy w pliku */etc/fstab* należy traktować jako tożsame:

```
/dev/hdc2           /xfs_test           xfs          defaults          1 0
LABEL=/xfs_test     /xfs_test           xfs          defaults          1 0
```

Trzecim sposobem montowania systemu plików jest podanie jego identyfikatora UUID. Identyfikacja systemów plików za pośrednictwem identyfikatora UUID daje podobną jak w przypadku etykiet niezależność od urządzenia, na którym umieszczono system plików, ponadto daje też możliwość odwoływania się do systemu plików — zgodnie z nazwą — unikalnych. Nie bez przyczyny identyfikatory te noszą miano uniwersalnych i unikalnych. Powyższy przykład ilustrujący sposób przypisywania etykiet do systemów

plików pokazuje, że użytkownicy (również ja sam) mają tendencję do przypisywania dyskom etykiet identycznych z nazwą katalogu będącego punktem montowania tych dysków (w przypadku dystrybucji Red Hat jest to wręcz domyślny sposób ustalania wartości etykiet podczas instalacji systemu operacyjnego). Choć takie etykiety systemów plików mają pewną wartość informacyjną (dają odpowiedź na pytania w rodzaju „gdzie też ja chciałem zamontować tę partycję?"), tym niemniej każdy system operacyjny Linux posiada partycję montowaną w katalogu /, co czyni etykietę tej partycji nazwą raczej mało unikalną, powtarzającą się w rozmaitych systemach komputerowych.

Rolą uniwersalnych unikalnych identyfikatorów (UUID) jest powiązanie urządzeń i całych systemów komputerowych z unikalnymi, ale rozpoznawalnymi sekwencjami znaków, co ma szczególne znaczenie w środowiskach sieciowych. W przypadku systemów plików identyfikatory te uniemożliwiają przypadkowe dwukrotne zamontowanie tego samego systemu plików w rzadkich przypadkach występowania w systemie komputerowym urządzeń pamięci, dla których zdefiniowano różne ścieżki dostępu do tych samych napędów i partycji. Identyfikatory UUID systemów plików XFS, w przeciwieństwie do etykiet, generowane są automatycznie podczas tworzenia systemu plików. Poniższy przykład ilustruje sposób określenia identyfikatora UUID odmontowanej partycji XFS:

```
# xfs_db -c 'uuid' /dev/hdc2
uuid = 8180d55b-4ed4-4966-8ff1-91eb099d1cf0
```

Jeżeli przyjmiemy, że system plików umieszczony na partycji `/dev/hdc2` posiada identyfikator UUID o wartości 8180d55b-4ed4-4966-8ff1-91eb099d1cf0, poniższe wpisy w pliku */etc/fstab* możemy traktować jako tożsame:

```
/dev/hdc2            /xfs_test            xfs            defaults        1 0
UUID=8180d55b-4ed4-4966-8ff1-91eb099d1cf0     /xfs_test     xfs      defaults       1 0
```

Identyfikator UUID może zostać przypisany do systemu plików XFS na kilka różnych sposobów. Po cóż jednak własnoręcznie ustalać wartość UUID, jeżeli jej główną cechą ma być unikalność? Najczęstszą sytuacją, w której zachodzi konieczność własnoręcznego ustalenia identyfikatora UUID, jest próba zamontowania kopii istniejącego systemu XFS. Kopie takie są zwykle obrazem systemu plików XFS przechowywanym w woluminie logicznym, ale możliwe jest również własnoręczne wykonanie wiernej co do bloku kopii systemu plików za pomocą polecenia `dd`. W takim przypadku zachodzi potrzeba samodzielnego ustalenia wartości UUID za pomocą jednego z następujących sposobów:

- automatycznie, za pośrednictwem programu `xfs_db` pracującego w trybie `expert`, generującego losowy identyfikator UUID po wydaniu polecenia

  ```
  # xfs_db -x -c 'uuid_generate' /dev/hdc2
  clearing log and setting uuid
  writing all SBs
  new uuid = ace30ae3-d2e2-459f-b2fd-a7db090b257e
  ```

- automatycznie, korzystając z programu `xfs_admin` i opcji `-U`

  ```
  # xfs_adimn -U 'uuidgen' /dev/hdc2
  clearing log and setting uuid
  writing all SBs
  new uuid = c6ff0c75-0795-4c6f-bd91-1238fd5c2686
  ```

- ręcznie, określając wartość ciągu UUID i przekazując ją jako parametr opcji `-U` polecenia `xfs_admin`

```
# xfs_adimn -U bc4eeccf-9638-4f75-a80a-3ba92ba499af /dev/hdc2
clearing log and setting uuid
writing all SBs
new uuid = bc4eeccf-9638-4f75-a80a-3ba92ba499af
```

Zanim — w celu skorzystania z etykiet czy identyfikatorów UUID systemu plików XFS — dokonasz zmian wpisów najważniejszych partycji w pliku */etc/fstab*, poćwicz na partycjach mniej ważnych. Spróbuj zamontować i odmontować zmodyfikowane systemy plików i sprawdzić w ten sposób, czy składnia nowych wpisów pliku */etc/fstab* jest poprawna.

Konfiguracja systemu kontroli dostępu do plików i katalogów systemu XFS

System plików XFS, jako jedyny spośród systemów plików z kroniką omawianych w tej książce, obsługuje własne listy kontroli dostępu (ang. *access control list*, ACL), będące otwartymi, konfigurowalnymi zbiorami uprawnień, nadawanych poszczególnym użytkownikom bądź ich grupom, i odnoszącymi się do konkretnych plików i katalogów. Listy kontroli dostępu zaimplementowane w ramach systemu XFS są zgodne ze standardem POSIX ACL, który do dziś nie może doczekać się powszechnego wdrożenia w systemach operacyjnych z rodziny Unix. Niezależnie od tego, kiedy reszta świata zaimplementuje wreszcie standardowe listy kontroli dostępu, użytkownicy systemu plików XFS (oraz rozproszonego systemu plików AFS omawianego w rozdziale 13., zatytułowanym „Rozproszony system plików: OpenAFS") mogą już dziś korzystać z list ACL, dających kontrolę nieporównywalnie lepszą od standardowych uprawnień systemu Linux i dających niektórym użytkownikom możliwość wykonywania określonych operacji w stosunku do pewnych plików i katalogów. Listy kontroli dostępu systemu XFS umożliwiają doprecyzowanie uprawnień przydzielanych użytkownikom i grupom użytkowników, zdefiniowanych w systemie Linux. Listy kontroli dostępu systemu AFS są znacznie bardziej rozbudowane: umożliwiają definiowanie własnych wirtualnych grup użytkowników i przydzielanie im wybranych plików i katalogów.

Lista kontroli dostępu (ACL) to — w uproszczeniu — lista użytkowników i grup użytkowników systemu Linux i przydzielonych im uprawnień do konkretnych plików i katalogów. Listy kontroli dostępu umożliwiają zdefiniowanie bardzo szczegółowych uprawnień, na przykład: „tylko użytkownicy `wvh`, `kwalter` i `jeffpk` mogą zapisywać dany plik; użytkownicy `darth` i `davep` mogą go co najwyżej odczytać". Uprawnienia te mogą być nadawane bez konieczności tworzenia setek specjalnych, wirtualnych grup użytkowników.

W systemie plików XFS dostępne są dwa podstawowe rodzaje list kontroli dostępu:

- listy kontroli dostępu do określonych plików i katalogów;
- katalogowe listy kontroli dostępu, znane też jako maski ACL, definiujące domyślną listę kontroli dostępu, przypisywaną do wszystkich plików tworzonych w danym katalogu.

Podstawowym ograniczeniem standardowych uniksowych mechanizmów kontroli dostępu jest to, że mają one dość ograniczony zasięg. Niektóre natomiast czynności, zwiększające elastyczność systemu kontroli dostępu, przypisane są jedynie użytkownikowi `root`. Przykładowo: w systemie pozbawionym list kontroli dostępu nie ma możliwości przyznania uprawnienia zapisu pliku więcej niż jednemu użytkownikowi, jeżeli nie utworzy się dla tych użytkowników osobnej grupy — czynność tę może wykonać jedynie `root`.

Jeżeli administrator Twojego systemu komputerowego jest Twoim kolegą (albo jeżeli oglądasz go codziennie w lustrze), w razie potrzeby możesz uzgodnić z nim utworzenie nowej grupy użytkowników obejmujących Ciebie i wszystkich, którym chcesz udostępnić plik w trybie do zapisu. Możesz przypisać tej grupie prawa własności do wszelkich plików i katalogów, do których chcesz zachować możliwość zapisu, i wymusić na wszystkich użytkownikach, mających należeć do nowej grupy, wykonanie polecenia `newgrp` w celu zmiany przynależności do grupy. Mimo że kocham system Unix, przejrzystość i elastyczność tego rozwiązania jest dla mnie co najmniej wątpliwa: metoda ta dorównuje tu czystości rynsztoka, a procedura postępowania jest równie prosta jak zapamiętanie dwudziestu adresów IP w formacie ósemkowo-kropkowym.

Listy kontroli dostępu wyświetlane są w standardowym formacie, składającym się z trzech pól oddzielonych znakami dwukropka:

- Pierwszym polem wpisu na liście ACL jest identyfikator rodzaju wpisu: `user` (u), `group` (g), `other` (o) lub `mask` (m).
- Drugie pole wpisu ACL to nazwa użytkownika, liczbowy identyfikator użytkownika, nazwa grupy lub liczbowy identyfikator grupy (w zależności od rodzaju wpisu). Jeżeli pole to jest puste, lista ACL odnosi się do użytkowników bądź grup będących właścicielami danego pliku czy katalogu. Maski i inne listy kontroli dostępu muszą mieć w drugim polu wartość pustą.
- Trzecie pole to uprawnienia dostępu dla bieżącej listy kontroli dostępu. Uprawnienia te są reprezentowane na dwa sposoby:
  - W postaci standardowego uniksowego ciągu znaków `rwx` (znaki te reprezentują odpowiednio uprawnienie do odczytu, zapisu i wykonywania, przy czym w przypadku katalogu uprawnienie do wykonywania oznacza możliwość przeszukiwania tego katalogu). Każda litera może być zastąpiona przez znak - (myślnik), co oznacza brak danego uprawnienia. Uprawnienia muszą być określone we właściwej kolejności.
  - W postaci symbolicznej, poprzedzanej przez znak + (plus) lub ^ (potęga), podobnej do postaci uprawnień przekazywanych do polecenia `chmod` przez osoby, którym obca jest sztuka ósemkowego zapisu liczb. W tej reprezentacji po znaku + lub ^ następuje ciąg znaków `r`, `w` lub `x`, wskazujący na uprawnienia, jakie chcemy dodać (+) lub usunąć (^) ze zbioru uprawnień danego pliku czy katalogu.

Poszczególne pola listy kontroli dostępu przechowywanej w pliku są oddzielane znakami spacji lub nowej linii. Wszelkie znaki pomiędzy znakiem # a najbliższym znakiem nowej linii są traktowane jak komentarz (tzn. są ignorowane).

W podrozdziale „Oprogramowanie narzędziowe dla systemu plików XFS" wspomniałem, że narzędzia do obsługi list kontroli dostępu systemu XFS umieszczone są w pakiecie `acl`, składającym się z trzech programów:

- `chacl` — programu umożliwiającego modyfikację, przeglądanie i usuwanie list ACL użytkowników, grup i masek w stosunku do plików i katalogów;
- `getfacl` — programu umożliwiającego przeglądanie list kontroli dostępu plików i katalogów;
- `setfacl` — programu umożliwiającego ustawianie list kontroli dostępu plików i katalogów.

Aby zademonstrować sposób korzystania z list ACL, skorzystajmy z katalogu o następującej zawartości i uprawnieniach:

```
# ls -al
drwxr-xr-x    2 wvh      wvh               23 Aug 31 00:19 .
drwxr-xr-x    3 wvh      wvh               47 Aug 31 00:19 ..
-rw-r-----    1 wvh      wvh            31372 Aug 31 00:19 resume.xml
```

Domyślna lista ACL dla tego katalogu ma następującą postać:

```
# getfacl .
# file: .
# owner: wvh
# group: wvh
user::rwx
group:r-x
other::r-x
```

Domyślna lista kontroli dostępu dla pliku *resume.xml* wygląda tak:

```
# getfacl resume.xml
# file: resume.xml
# owner: wvh
# group: wvh
user::rw-
group:r--
other::---
```

Domyślna lista ACL dla pliku tworzonego w katalogu, dla którego nie ustawiono jeszcze jawnie domyślnej maski ACL, odpowiada standardowym uniksowym uprawnieniom, przydzielanym użytkownikowi tworzącemu ten plik. Domyślne ustawienia uprawnień systemu Unix uzależnione są natomiast od wartości zmiennej środowiskowej `umask`.

Istnieją trzy powszechnie stosowane sposoby zmiany listy ACL pliku lub katalogu:

- jawne ustawienie listy za pośrednictwem polecenia `setfacl`,
- zmodyfikowanie istniejącej listy za pośrednictwem polecenia `chacl`,
- zmodyfikowanie istniejącej listy za pośrednictwem opcji `-m` polecenia `setfacl`.

Zmienna środowiskowa `umask` to klasyczny uniksowy mechanizm ustawiania domyślnych zabezpieczeń dla tworzonych przez użytkowników plików i katalogów. Domyślnie zmienna `umask` jest trzycyfrową liczbą ósemkową odejmowaną od standardowej wartości uprawnienia dla nowo tworzonego pliku (666) lub katalogu (777). W większości dystrybucji Linuksa wartość zmiennej `umask` wynosi 002, co oznacza, że każdy tworzony plik będzie objęty uprawnieniami reprezentowanymi ósemkową wartością 664 (zarówno właściciel pliku, jak i grupa, do której należy, mogą odczytywać i modyfikować plik; pozostali użytkownicy mogą jedynie odczytywać plik). Analogicznie każdy nowo utworzony katalog będzie objęty uprawnieniami reprezentowanymi w postaci ósemkowej jako 775, co oznacza, że właściciel katalogu i grupa, do której on należy, mogą tworzyć pliki wewnątrz katalogu. Pozostali użytkownicy uprawnieni są jedynie do odczytywania zawartości katalogu i wyszukiwania w nim plików.

Większość użytkowników ustawia zmienną `umask` na wartość 022, tak aby nowo tworzone pliki mogły być zapisywane wyłącznie przez właściciela (pliki są wtedy tworzone z uprawnieniami 664); katalogi otrzymują podobne uprawnienia (775). Możliwość zmiany zmiennej `umask` to element składowy powłoki systemu Unix; Szczegółowe informacje na ten temat znajdują się w dokumentacji *man* dołączonej do używanej przez Ciebie powłoki.

W przykładach prezentowanych w bieżącym rozdziale wykorzystałem polecenie `chacl`, ponieważ nie powoduje ono nadpisania aktualnej listy ACL dla danego pliku czy katalogu (co następuje w przypadku wykonania polecenia `setfacl`), lepiej przy tym ilustrując sposób działania list ACL.

Listy kontroli dostępu dają możliwość przyznawania szczegółowych uprawnień dostępu do plików i katalogów, wykraczających poza możliwości standardowych mechanizmów zabezpieczających systemu Unix. Przykładowo dodanie do systemu użytkownika `kwalter`, mającego dostęp do pliku *resume.xml* w trybie do odczytu, wymaga jedynie wykonania polecenia `chacl`, jak pokazano poniżej:

```
# chacl u::rw-,g::r--,o::---,u:kwalter:r--,m::rw- resume.xml
```

Nie jest to wbrew pozorom ciąg modemowego połączenia z Internetem (podejrzewam, że polecenie takie rozpoznawane jest przez wiekowy edytor TECO) — jest to rzeczywista postać listy kontroli dostępu dla pliku. Jak już wspomniałem, lista ACL składa się z trzech oddzielonych dwukropkami pól, reprezentujących uprawnienia użytkownika (właściciela pliku), grupy (grupy, do której należy właściciel pliku) oraz pozostałych użytkowników. Modyfikując listę ACL danego pliku, musimy najpierw podać poprzednią postać listy, a dopiero potem określić nowe jej elementy. Fragment `u::rw-,g::r--,o::---` w powyższym przykładzie to właśnie dotychczasowa zawartość listy ACL dla pliku *resume.xml*; część `u:kwalter:r--,m::rw-` określa nowego użytkownika pliku (`kwalter`) i tzw. *maskę praw uzyskanych* przez tego użytkownika. Maska praw uzyskanych jest sumą wszystkich uprawnień istniejących użytkowników i grup do pliku lub katalogu. Podczas dodawania użytkownika do listy ACL konieczne jest określenie maski praw uzyskanych.

O tym, że użytkownik `kwalter` rzeczywiście został dodany do listy kontroli dostępu pliku *resume.xml*, przekona nas wydruk polecenia `getfacl`:

```
# getfacl resume.xml
# file: resume.xml
# owner: wvh
```

```
# group: wvh
user::rw-
group:r--
other::---
user:kwalter:r--
mask::r--
```

Tymczasem polecenie `ls -al` ujawnia, że standardowe uprawnienia systemu Unix do tego pliku nie zostały zmodyfikowane:

```
# ls -al
drwxr-xr-x    2 wvh      wvh               23 Aug 31 00:19 .
drwxr-xr-x    3 wvh      wvh               47 Aug 31 00:19 ..
-rw-r-----    1 wvh      wvh            31372 Aug 31 00:19 resume.xml
```

Możemy jednak zweryfikować uprawnienia użytkownika `kwalter`, nakazując mu wykonanie próby odczytu pliku *resume.xml* — próba ta powinna się udać (jeżeli znasz hasło tego użytkownika możesz sam spróbować sprawdzić skuteczność działania polecenia `chacl`, nawiązując z komputerem połączenie telnetowe, logując się jako użytkownik `kwalter` i próbując otworzyć plik w dowolnym edytorze tekstu lub przekazać plik do poleceń takich jak `more` lub `cat`).

Bardziej jednak niż możliwość nadawania konkretnym użytkownikom prawa do odczytu pliku interesuje nas oczywiście zdolność wskazywania pojedynczych użytkowników jako uprawnionych do zapisu danego pliku. W systemie XFS aby wskazać użytkownika `kwalter` jako osobę mającą uprawnienia tak do odczytu, jak i do zapisu pliku *resume.xml*, należy wykonać następujące polecenie:

```
# chacl u::rw-,g::r--,o::---,u:kwalter:rw-,m::rw- resume.xml
```

Polecenie `getfacl` pokaże aktualną zawartość listy ACL pliku *resume.xml* uwzględniającą nowe uprawnienia użytkownika `kwalter`:

```
# getfacl resume.xml
# file: resume.xml
# owner: wvh
# group: wvh
user::rw-
group:r--
other::---
user:kwalter:rw-
mask::rw-
```

Tak jak uprzednio, weryfikacja skuteczności działania polecenia `chacl` jest możliwa przy współpracy użytkownika `kwalter`. Współpraca ta polega na wykonaniu próby odczytu i zapisu pliku *resume.xml* (znając hasło tego użytkownika można samodzielnie sprawdzić działanie nowej listy kontroli dostępu poprzez nawiązanie z komputerem połączenia telnetowego, zalogowanie się w imieniu użytkownika `kwalter` i otwarcie pliku w dowolnym edytorze tekstu lub przekazanie go do poleceń takich jak `more` lub `cat`).

Jestem wielkim fanem systemu list kontroli dostępu, ponieważ daje on doświadczonemu użytkownikowi pełną kontrolę nad tym, kto i w jakim trybie może uzyskać dostęp do plików będących własnością danego użytkownika. Listy kontroli dostępu są obsługiwane w systemie plików XFS, ale docelowo mają być obsługiwane przez wszystkie linuksowe

systemy plików; w tym momencie wyeliminowana zostanie najbardziej dokuczliwa wada administracyjna systemów operacyjnych z rodziny Unix (i straci aktualność jeden z wielu argumentów za wyższością systemów Windows NT, 2000 i XP). Skłamałbym, gdybym stwierdził, że stosowanie list ACL systemu XFS jest proste, ale z Linuksem jest jak z wytrawnym winem: jakość przychodzi z czasem. Polecam lekturę rozdziału 13., zawierającego opis niezwykle rozbudowanych (i łatwych w użyciu) list kontroli dostępu systemu plików OpenAFS (i jego komercyjnego kuzyna — systemu AFS) firmy IBM.

Kontrola spójności systemu plików XFS

Zdążyłeś już pewnie zapamiętać, że podstawową zaletą systemów plików z kroniką są krótsze czasy przeładowania systemu i minimalizacja ilości zmian systemu plików traconych w przypadku awarii komputera. Program `fsck.xfs` kończy działanie bezpośrednio po uruchomieniu, opierając się na założeniu, że spójność systemu plików XFS jest gwarantowana przez mechanizm odtwarzania oczekujących transakcji zapisanych w dzienniku. Zwykle jest to mechanizm wystarczająco skuteczny, niemniej jednak zdarzają się sytuacje, w których wymuszenie nagłego przeładowania systemu może narazić na szwank spójność systemu plików XFS. Najczęstszymi sytuacjami tego rodzaju są awarie dysku, na którym przechowywany jest system plików. Awarie te mogą dotknąć sam dysk lub jego kontroler i objawiają się użytkownikom w postaci komunikatów o błędach dostępu do dysku.

Oprogramowanie narzędziowe systemu XFS zawiera dwa programy sprawdzające spójność systemu plików: `xfs_check` (omawiany teraz), wykrywający nieprawidłowości struktur wewnętrznych systemu plików oraz `xfs_repair` (omawiany nieco dalej), służący do korygowania wykrytych nieprawidłowości. Podział oprogramowania na sprawdzające i korygujące spójność systemu plików uzasadniony jest dwoma względami:

- Program `xfs_check` jest skryptem powłoki wykorzystującym program `xfs_db` (w trybie nieinteraktywnym) i analizującym za jego pośrednictwem stan systemu plików. Program ten działa zdecydowanie szybciej niż `xfs_repair`, ponieważ zajmuje się jedynie przeszukiwaniem systemu plików i nie musi alokować struktur danych i informacji wymaganych podczas korygowania ewentualnych błędów systemu plików.
- Przez wiele lat nie istniał żaden odpowiednik programu `xfs_repair`, ponieważ stabilność i niezawodność systemu plików XFS czyniła taki program zbędnym. Odtwarzanie dziennika podczas montowania przywraca spójność systemu plików w niemal wszystkich przypadkach, a sytuacje awaryjne spowodowane uszkodzeniami sprzętu są zwykle rozwiązywane przez wymianę problematycznego sprzętu i przywrócenie systemu plików z kopii zapasowej.

Przed uruchomieniem programu `xfs_check` należy odmontować system plików. Uruchomienie programu `xfs_check` w stosunku do zamontowanego systemu plików powoduje niejednokrotnie błędy spowodowane aktywnością dziennika i realizacją operacji zapisu w systemie plików. Dobrze jest wyrobić sobie nawyk każdorazowego odmontowywania systemu plików przed uruchomieniem programu `xfs_check`.

W większości przypadków sprawdzany system plików XFS nie będzie obarczony błędami spójności, a program `xfs_check` zakończy działanie bez wyświetlania żadnych komunikatów; sytuację taką ilustruje poniższy przykład:

```
# xfs_check /dev/hdc2
#
```

Wyjątkowi pechowcy mogą jednak ujrzeć pewną liczbę komunikatów o błędach, np:

```
# xfs_check .dev.hdc2
xfs_check: unexpected XFS magic number 0xa7b9acbd
bad superblock magic number a7b9acbd, giving up
#
```

Jeżeli zobaczysz podobny komunikat (lub każdy inny groźnie brzmiący komunikat) skorzystaj z programu `xfs_repair`, opisywanego w dalszej części rozdziału.

Archiwizacja i przywracanie systemu plików XFS z kopii zapasowej

Wykonywanie kopii zapasowej — niezależnie od niezwykłej niezawodności systemu plików — jest nieodzownym elementem każdej poważnej instalacji komputerowej, której przydatność uwarunkowana jest dostępem do danych. Dane te mogą mieć różną wartość informacyjną — od prywatnej kolekcji babcinych przepisów po bazę danych urzędu skarbowego przechowującą deklaracje podatkowe każdego obywatela z ostatnich 10 lat. Niektórzy jednak uważają, że świat byłby piękniejszy bez wynalazków w rodzaju babcinej sałatki szpinakowej czy podatków i tabunów urzędników skarbowych pukających od drzwi do drzwi.

Kopie zapasowe dają możliwość przywrócenia plików, które zostały przypadkowo usunięte, ale — co ważniejsze — stanowią również swego rodzaju zabezpieczenie przed wypadkami losowymi: od awarii dysku czy kontrolera dyskowego po całkowite zniszczenie pomieszczeń komputerowych w wyniku pożaru, trzęsienia ziemi, wybuchu pobliskiego wulkanu czy uderzenia małej komety. Wykonywanie kopii zapasowych zajmuje dużo czasu i nie daje natychmiastowych korzyści — podobnie jak na przykład sen, równie, wydawałoby się, bezużyteczny. Jednak brak zarówno jednego (snu), jak i drugiego (archiwizowania) daje się po czasie we znaki; brak aktualnych kopii zapasowych może narazić na niebezpieczeństwo Ciebie i Twoje otoczenie (równie ważne jak samo wykonywanie kopii jest sprawdzanie, czy stan wykorzystywanych do tego nośników pozwala na późniejszy odczyt tych danych i czy są one rzeczywiście zapisywane; raz już sparzyłem się na bagatelizowaniu stanu nośników i nie zamierzam więcej popełniać tego błędu).

Standardowy linuksowy (uniksowy) program `dump` (do wykonywania kopii systemu plików) i program `restore` to wydajne narzędzia o dużych możliwościach, przetestowane w ciągu wielu lat użytkowania, skuteczniejsze w ratowaniu posad niż osławiona wazelina. Niestety, standardowy linuksowy zestaw `dump-restore` nie jest w tej chwili przystosowany do obsługi systemów plików takich jak XFS, które nie korzystają ze standardowych i-węzłów i nagłówków systemu plików i systemów plików zawierających

niestandardowe struktury danych, takie jak listy kontroli dostępu czy atrybuty rozszerzone. Ponadto programy `dump` i `restore`, jako uniwersalne narzędzia projektowane na potrzeby każdego systemu plików, nie potrafią współpracować z rozmaitymi mechanizmami optymalizującymi, będącymi częścią nowoczesnych systemów plików.

Wykonanie kopii zapasowej systemu plików XFS za pomocą programu `dump` jest niemożliwe. Jeżeli z jakichś względów wzbraniasz się przed skorzystaniem z programu `xfsdump`, jedyną możliwością jest samodzielne napisanie programu archiwizacji systemu plików (Hm…), skorzystanie z dostępnych programów obsługujących system plików XFS lub uruchomienie ograniczonych, ale sprawdzonych programów takich jak `dd`.

Na szczęście w skład zestawu narzędziowego XFS wchodzą programy `xfsdump` i `xfsrestore`. Każdy z tych programów — jak się zapewne domyślasz — obsługuje wiele opcji. Niżej znajduje się opis najpopularniejszych z nich. Pełny opis dostępnych opcji programów `xfsdump` i `xfsrestore` znajduje się w dokumentacji *man* dla tych programów.

Program xfsdump

Program `xfsdump`, tak jak i jego linuksowy (uniksowy) przodek, umożliwia użytkownikowi wykonanie pełnej lub przyrostowej kopii zapasowej systemu plików. Pełne kopie (znane też jako *epoki* lub *kopie poziomu zerowego*) to kopie zapasowe zawierające wszystkie pliki i katalogi przechowywane w danym systemie plików. Kopie przyrostowe to kopie wszystkich plików i katalogów przechowywanych w danym systemie plików, które zostały zmodyfikowane od momentu sporządzenia poprzedniej kopii. Program `xfsdump` przechowuje informacje o wykonanych kopiach zapasowych w plikach binarnych w katalogu */var/xfsdump/Inventory* (program `dump` korzysta w tym celu z katalogu */etc/dumpdates*). Dzięki temu program `xfsdump` może (w porównaniu z prostymi wpisami tworzonymi przez program `dump`) przechowywać znacznie więcej informacji o wykonanych kopiach. Zawartość takiego pliku można w każdej chwili wyświetlić za pomocą opcji `-I` programu `xfsdump`, który powinien wtedy wygenerować wydruk podobny do przedstawionego poniżej:

```
# xfsdump -I
file system 0:
      fs id:          c1fba169-94be-45e0-bade-d80dab54d93e
      session 0:
        mount point:  journal.vonhagen.org:/home/wvh/writing
        device:       journal.vonhagen.org:/dev/vg1/xfs_vol
        time:         Sun Sep  2 16:40:23 2001
        session label:  "Std_Sunday"
        session id:   2a773dcd-51c6-4d7e-bfbf-d53f62078c95
        level:        0
        resumed:      NO
        subtree:      NO
        streams:      1
        stream 0:
                pathname:       /dev/st0
                start:          ino 132 offset 0
                end:            ino 2097287 offset 0
                interrupted:    NO
```

```
media files:    1
media file 0:
      mfile index:      0
      mfile type:       data
      mfile size:       105805312
      mfile start:      ino 132 offset 0
      mfile end:        ino 2097287 offset 0
      media label:      "Yellow"
      media id: 723c222-7ad8-4548-b282-59017752c18d
```

Rejestrowanie wszystkich tych informacji ułatwia przywracanie systemu plików, ponieważ dzięki nim program `xfsrestore` może określić kolejność odtwarzania kopii na poszczególnych nośnikach archiwizacyjnych. Opcja `-I` programu `xfsdump` pozwala również na określenie kilku parametrów, ograniczających wyświetlane informacje do interesujących nas zbiorów; określanie parametrów oparte jest na kryteriach takich, jak poziom kopii (ang. *dump level*), punkt montowania systemu plików, urządzenie, na którym umieszczony jest archiwizowany system plików, identyfikator systemu plików, etykieta nośnika (ang. *media label*) i tym podobne.

Standardowy wiersz polecenia wywołujący program `xfsdump` oraz wygenerowany przez niego wydruk wyglądają następująco:

```
[wvh]# xfsdump -l 0 -o -L Std_Sunday -M Yellow /dev/st0 /dev/vg1/xfs_vol
xfsdump: version 3.0. - Running single-threaded
xfsdump: level 0 dump of journal.vonhagen.org:/home/wvh/writing
xfsdump: dump date: Sun Sep  2 16:40:23 2001
xfsdump: session id: 2a773dcd-51c6-4d7e-bfbf-d53f62078c95
xfsdump: session label: "Std_Sunday"
xfsdump: ino map phase 1: skipping (no subtrees specified)
xfsdump: ino map phase 2: constructing initial dump list
xfsdump: ino map phase 3: skipping (no pruning necessary)
xfsdump: ino map phase 4: skipping (size estimated in phase 2)
xfsdump: ino map phase 5: skipping (only one dump stream)
xfsdump: ino map construction complete
xfsdump: estimated dump size: 105794176 bytes
xfsdump: creating dump session media file 0 (media 0, file 0)
xfsdump: dumping ino map
xfsdump: dumping directories
xfsdump: dumping non-directory files
xfsdump: ending media file
xfsdump: media file size 105805312 bytes
xfsdump: dump size (non-dir files) : 105772248
xfsdump: dump complete: 348 seconds elapsed
```

W wierszu polecenia zostały zastosowane następujące opcje:

- `-l 0` — opcja ta pozwala na określenie poziomu wykonywanej kopii. W tym przypadku wykonałem kopię poziomu zerowego (epokę) systemu plików, czyli zarchiwizowałem wszystkie przechowywane w nim pliki.
- `-o` — opcja ta umożliwia nadpisanie wszelkich danych znajdujących się już na nośniku archiwizacyjnym. Domyślnie program `xfsdump` nie nadpisuje danych znajdujących się na nośnikach wyznaczonych do archiwizacji, o ile dane te stanowią kopię zapasową systemu plików XFS (utworzoną przez program `xfsdump`).

- `-L Std_Sunday` — za pośrednictwem tej opcji określiłem etykietę sesji archiwizacyjnej. Treść tej etykiety powinna być chronologiczna i znacząca dla operatora — przeważnie na etykietach zapisuje się dzień wykonania kopii, poziom kopii i inne przydatne informacje. Etykietę można wykorzystać podczas identyfikacji informacji związanych z konkretną sesją archiwizacyjną, zamieszczoną na wydruku wygenerowanym przez program `xfsdump` wywołany z opcją `-I`.
- `-M Yellow` — opcja `-M` pozwala na określenie etykiety zapisywanej na nośniku, na którym wykonywana jest kopia zapasowa. Również treść tej etykiety powinna znaczyć coś dla operatora procesu archiwizacji, może więc na przykład wskazywać na zestaw taśm, na których składowane są kopie (jak zdążyłeś się pewnie domyślić, ja stosuję oznaczenia kolorowe) lub inne tego typu informacje. Etykieta taka może zostać później przekazana do polecenia `xfsdump -I mobjlabel=<etykieta>`, w wyniku czego wydrukowane zostaną wyłącznie informacje dotyczące kopii zapisanych na nośniku opatrzonym podaną etykietą.
- `-f /dev/st0` — opcja `-f` pozwala określić plik lub urządzenie, na którym ma zostać zapisana wykonywana kopia zapasowa.
- `/dev/vg1/xfs_vol` — nazwa urządzenia, na którym przechowywany jest system plików będący obiektem archiwizacji.

Poza wspomnianymi już możliwościami program `xfsdump` oferuje kilka wyrafinowanych funkcji, takich jak wykonywanie kopii jedynie określonych plików lub podkatalogów, przerywanie i wznawianie procesu wykonywania kopii itp. Jeżeli jesteś administratorem systemu i korzystasz z systemu plików XFS, będziesz mile zaskoczony szerokimi możliwościami tego programu.

Program xfsrestore

Program `xfsrestore` stanowi uzupełnienie programu `xfsdump` i umożliwia odczytywanie taśm archiwizacyjnych i przywracanie ich zawartości. Program `xfsrestore` udostępnia wiele opcji pozwalających na kontrolę procesu przywracania kopii zapasowych, z przywracaniem pojedynczych plików lub plików spełniających określone warunki włącznie.

Najpopularniejszym sposobem wywoływania programu `xfsrestore` jest podanie urządzenia zawierającego nośnik kopii zapasowej i określenie punktu montowania systemu plików, który ma być na podstawie tej kopii odtworzony. Poniższy przykład ilustruje sposób wywołania programu `xfsrestore` i efekty jego działania:

```
# xfsrestore -f /dev/st0 /home/wvh/writing
xfsrestore: version 3.0 - Tunning single-threaded
xfsrestore: searching media for dump
xfsrestore: examining media file 0
xfsrestore: dump description:
xfsrestore: hostname: journal.vonhagen.org
xfsrestore: mount point /home/wvh/writing
xfsrestore: volume: /dev/vg1/xfs_vol
xfsrestore: session time: Sun Sep  2 17:55:21 2001
xfsrestore: level: 0
```

```
xfsrestore: session label: "Std_Sunday"
xfsrestore: media label: "Yellow"
xfsrestore: file system id: c1fba169-94be-45e0-bade-d80dab54d93e
xfsrestore: session id: odcd2372-67b1-4411-b085-35386f25b90e
xfsrestore: media id: aa9aec93-9ea4-459a-b1b9-2c6736a83454
xfsrestore: using online session inventory
xfsrestore: searching media for directory dump
xfsrestore: reading directories
xfsrestore: directory post-processing
xfsrestore: restoring non-directory files
xfsrestore: restore complete: 114 seconds elapsed
```

Jeżeli nie masz pewności, że plik, którego szukasz, znajduje się na taśmie archiwizacyjnej, możesz skorzystać z opcji `-i` programu `xfsrestore`. Opcja ta działa podobnie do opcji `-i` standardowego linuksowego programu `restore`. Po wywołaniu polecenia `xfsrestore` z opcją `-i` program konstruuje w pamięci miniaturę systemu plików i przenosi operatora do konsoli wewnętrznej programu. Z poziomu tej konsoli możliwe jest przeglądanie wirtualnych katalogów kopii, sprawdzanie obecności poszukiwanych plików i oznaczanie ich jako przeznaczonych do przywrócenia z kopii.

Pełna dokumentacja programu `xfsrestore` dostępna jest w ramach dokumentacji systemowej wyświetlanej po wykonaniu polecenia `man xfsrestore` lub `info xfsrestore`.

Rozwiązywanie problemów dotyczących systemu plików XFS

Lektura punktu zatytułowanego „Kontrola spójności systemu plików XFS" uświadomiła nas, że system plików XFS przez znaczną część swojego okresu „życia" nie posiadał programu sprawdzającego spójność systemu plików — głównie dlatego, że nie był on potrzebny. Niezależnie jednak od tego, jak rzadko zdarza się oglądać system plików XFS w stanie niespójnym, niemożliwym do wyeliminowania poprzez zwykłe odtworzenie zawartości dziennika transakcji, sytuacja taka jest prawdopodobna. Jak już wspomniałem, większość takich sytuacji spowodowanych jest uszkodzeniami sprzętu i innymi problemami sprzętowymi, mimo to ważna jest świadomość dostępności środków programowych umożliwiających usunięcie ewentualnego problemu.

Jeżeli system plików XFS znajduje się w stanie niespójnym i niespójność ta nie może zostać wyeliminowana poprzez odtworzenie dziennika transakcji, automatyczne przeładowanie systemu zakończy się błędem: niemożliwe jest bowiem automatyczne zamontowanie systemu plików oznaczonego jako niepoprawny (ang. *dirty*). W takim przypadku należy sprawdzić dziennik ładowania systemu, szukając w nim informacji na temat przyczyny nieudanego montowania systemu plików i upewnić się, czy urządzenie, na którym przechowywany jest system plików, zostało w procesie inicjalizacji systemu poprawnie wykryte. Jeżeli urządzenie nie zostało odnalezione, jest urządzeniem wewnętrznym, a pozostałe urządzenia podłączone do tego samego kontrolera zostały wykryte poprawnie, uszkodzeniu uległ najprawdopodobniej napęd, na którym przechowywany był system plików. Jeżeli natomiast system nie wykrył żadnego z urządzeń obsługiwanych przez dany kontroler, bardziej prawdopodobne jest uszkodzenie kontrolera.

Jeżeli problem rzeczywiście wynika z awarii sprzętu, omówienie rozwiązania tego problemu wykracza poza tematykę tej książki. W skrócie jednak sprowadza się ono do realizacji następującej procedury: jeżeli awaria wymaga zastąpienia jednego z dysków, po jego wymianie wynieś z piwnicy taśmy z kopiami zapasowymi i przywróć (za pomocą programu `xfsrestore`) zawartość systemu plików. Jeżeli nie posiadasz kopii zapasowych, a szczęśliwym trafem twoje CV znajduje się na partycji innego, nieuszkodzonego dysku, radzę rozpocząć jego drukowanie — wkrótce będzie Ci potrzebne.

Jeżeli problem nie jest bezpośrednio związany z awarią sprzętu, masz sporo szczęścia: istnieje jeszcze możliwość wdrożenia rozwiązań zajmujących mniej czasu niż zamiana dysków i odtworzenie kopii zapasowej. Ostatnie dwa zagadnienia tego rozdziału poświęcone są omówieniu dostępnych w tej sytuacji możliwości. Wybór jednej z nich uzależniony jest od umiejscowienia pechowego fragmentu systemu plików, będącego przyczyną niespójności systemu plików XFS. Tak więc ostatnie dwa punkty omawiają odpowiednio sposób eliminowania niespójności w obszarze dziennika oraz sposób eliminowania niespójności w pozostałych obszarach systemu plików.

Montowanie systemu pliku XFS bez odtwarzania zawartości dziennika

W niektórych przypadkach uszkodzeniu może ulec dziennik systemu plików XFS. Sytuacja taka może być spowodowana błędami kodu jądra systemowego, błędami kodu obsługującego kronikę systemu plików XFS lub nawet chwilowymi uszkodzeniami sprzętu. Do eliminowania uszkodzeń dziennika i innych niespójności systemu plików XFS przeznaczony jest program `xfs_repair`.

Jeżeli system plików XFS nie może zostać poprawnie zamontowany z powodu niemożności odtworzenia uszkodzonego dziennika, a mimo wszystko musimy podjąć próbę odtworzenia danych systemu plików jeszcze przed uruchomieniem programu `xfs_repair`, wykonując poniższe polecenie możemy zamontować system plików bez odtwarzania transakcji dziennika:

```
mount -t xfs -o ro,norecovery system_plików punkt_montowania
```

Polecenie to powoduje zamontowanie systemu plików XFS w trybie tylko do odczytu i bez odtwarzania transakcji zapisanych w dzienniku systemu plików. Ponieważ dziennik nie jest odtwarzany, system plików będzie niemal na pewno niespójny — wszelkie zmiany metadanych zarejestrowane w dzienniku nie zostały bowiem utrwalone w systemie plików; daje to jednak szansę dostępu do plików przechowywanych na problematycznej partycji. Należy bezzwłocznie wykonać ich kopię zapasową. Następnie można odmontować system plików i uruchomić program `xfs_repair`.

Program xfs_repair

Zadaniem programu `xfs_repair` jest eliminacja przyczyn niespójności systemu plików XFS. Podobnie jak wiele innych programów kontrolujących spójność systemów plików, program `xfs_repair` wprowadza zmiany do naprawianego systemu plików, polegające między innymi na przenoszeniu plików do katalogu */lost+found*, zerowaniu lub usuwaniu i-węzłów, uaktualnianiu zawartości struktur odwzorowujących przypisanie bloków do plików (w przypadku, kiedy kilka plików rości pretensje do tego samego bloku danych), przepisywaniu superbloku systemu plików z jednej z jego kopii (w przypadku uszkodzenia głównego superbloku systemu plików) itd.

W systemie plików XFS katalog */lost+found* jest tworzony automatycznie po uruchomieniu programu `xfs_repair`. Jeżeli w systemie plików XFS, w którym uruchamiany jest program `xfs_repair` istnieje już katalog */lost+found*, katalog ten jest usuwany i tworzony ponownie, co gwarantuje brak konfliktów nazw zapisywanych w nim plików.

Program `xfs_repair` kończy działanie, jeżeli nie potrafi odnaleźć poprawnego superbloku systemu plików; może również zgłosić błąd w przypadku niemożności odczytania bloków systemu plików spowodowanych uszkodzeniem powierzchni dysku. W tym ostatnim przypadku szansą odzyskania zawartości systemu plików jest jego skopiowanie na inną partycję o identycznym rozmiarze, odmontowanie pierwotnego systemu plików i uruchomienie programu `xfs_repair` w celu naprawienia kopii. Wszelkie dane przechowywane w uszkodzonych blokach zostaną utracone, tym niemniej istnieje szansa na odtworzenie reszty systemu plików.

Program `xfs_repair` podejmuje próbę naprawy uszkodzonego systemu plików XFS, wykonując w tym celu siedem czynności naprawczych. Są to:

1. Odnalezienie i sprawdzenie poprawności superbloku systemu plików. Jeżeli program `xfs_repair` nie może odczytać głównego superbloku, przeszukuje system plików w poszukiwaniu superbloków rezerwowych. Po odnalezieniu pierwszego poprawnego superbloku program naprawczy przepisuje jego zawartość do głównego superbloku systemu plików.
2. Sprawdzenie spójności wewnętrznego dziennika i wyzerowanie jego zawartości. Program skanuje kolejne porcje danych systemu plików w celu zweryfikowania poprawności wszelkich struktur danych systemu plików, wykrycia uszkodzonych i-węzłów itd.
3. Przeszukanie każdej z grup alokacji i weryfikacja jej wewnętrznej spójności, wykrycie i-węzłów nie znajdujących się na liście i-węzłów przydzielonych do plików, usunięcie węzłów nie powiązanych z żadnymi plikami itd.
4. Sprawdzenie list bloków przypisanych do plików w danej grupie alokacji, rozstrzygnięcie problemów takich jak bloki zawłaszczone przez kilka plików itp.
5. Weryfikacja struktur danych wszystkich grup alokacji, weryfikacja list zaalokowanych i-węzłów i innych wewnętrznych struktur danych.
6. Sprawdzenie i ewentualna korekta wszelkich niedowiązanych lub dowiązanych krzyżowo plików i katalogów systemu plików polegające na przeszukaniu całego drzewa katalogów i i-węzłów, dowiązaniu (o ile to możliwe) wszelkich plików i katalogów nie wskazywanych przez żaden katalog do katalogu */lost+found*.
7. Sprawdzenie i poprawienie wszystkich liczników dowiązań do plików.

Program `xfs_repair` rozpoznaje stosunkowo niewielką liczbę opcji. Najbardziej przydatne z nich to:

- `-l` *`urządzenie-przechowujące-dziennik`* — umożliwia określenie zewnętrznego urządzenia przechowującego dziennik uszkodzonego systemu plików XFS.
- `-n` — zgłasza wykryte problemy i kończy działanie, nie podejmując żadnych kroków naprawczych. Opcja ta może spowodować zakończenie działania

programu dużo wcześniej, niż w przypadku podejmowania prób naprawy uszkodzeń, ponieważ w wielu przypadkach kolejne analizy i próby naprawy mogą zależeć od sposobu naprawy błędów wykrytych wcześniej.

Poniżej znajduje się wydruk wygenerowany przez program `xfs_repair` podczas naprawy mocno uszkodzonego systemu plików:

```
# xfs_repair /dev/hdc2
Phase 1 - find and verify superblock...
bad primary superblock - bad magic number !!!
attempting to find secondary superblock...
.....................................................................
.....................................................................found
candidate secondary superblock...
verified secondary superblock...
writnig modified primary superblock
sb root inode value 18446744073709551615 inconsistent with calculated value
13835049877664432192
resetting superblock root inode pointer to 18446744069414584384
sb realtime bitmap inode 18446744073709551615 inconsistent with calcucalted value
13835049877664432193
resetting superblock realtime bitmap ino pointer to 18446744069414584385
sb realtime bitmap inode 18446744073709551615 inconsistent with calcucalted value
13835049877664432194
resetting superblock realtime bitmap ino pointer to 18446744069414584386
Phase 2 - using internal log
        - zero log...
        - scan filesystem freespace and inode maps...
zeroing unused portion of secondary superblock 0 sector
bad length 65552 for agf 0, should be 262206
flfirst -1 in agf 0 too large (max = 128)
fllast -4 in agf 0 too large (max = 128)
bad length # 65552 for agi 0, should be 262206
reset bad agf for ag 0
reset bag agi for ag 0
freeblk count 1 != flcount -5 in ag 0
bad magic # 0x58414749 in btbno block 0/1
expected level -3 got 0 in btbno block 0/1
bno freespace btree block claimed (state 4), agno 0, bno 1, suspect 0
bad agbno 16318461 for btbcnt root, agno 0
bad magic # 0xbebdabbc in inobt block 0/3
expected level 0 got 65535 in inobt block 0/3
dubious inode btree block header 0/3
badly aligned inode rec (starting inode = 4294967286)
bad starting inode # (4294967286 (0x0 0xfffffff6)) in ino rec, skipping rec
badly alogned inode rec (starting inode = 252)
ir_freecount/free mismatch, inode chunk 0/252, freecount 251658240 nfree 0
badly aligned inode rec (starting inode = 0)
badly starting inode # (0 (0x0 0x0)) in ino rec, skipping rec
badly aligned inode rec (starting inode = 4294967295)
bad starting inode # (4294967295 (0x0 0xffffffff)) in ino rec, skipping rec
zeroing unused portion of secondary superblock 1 sector
bad magic # 0xa7b9acbd in btbno block 1/2
expected level 0 got 65535 in btbno block 1/2
block (1,1) multiply claimed by bno space tree, state - 4
block (1,2) multiply claimed by bno space tree, state - 7
block (1,7129) multiply claimed by bno space tree, state - 1
...
bad starting inode # (4194304 (0x2 0x0)) in ino rec, skipping rec
```

```
badly aligned inode rec (starting inode = 4194304)
bad starting inode # (4194304 (0x2 0x0)) in ino rec, skipping rec
ir_freecount/free mismatch, inode chunk 2/224, freecount -1 nfree 64
badly aligned inode rec (starting inode = 4294967295)
bad starting inode # (4294967295 (0x2 0xffffffff)) in ino rec, skipping rec
badly aligned inode rec (starting inode = 4294967295)
bad starting inode # (4294967295 (0x2 0xffffffff)) in ino rec, skipping rec
zeroing unused portion of secondary superblock 3 sector
zeroing unused portion of secondary superblock 4 sector
zeroing unused portion of secondary superblock 5 sector
zeroing unused portion of secondary superblock 6 sector
zeroing unused portion of secondary superblock 7 sector
root inode chunk not found
Phase 3 - for each AG...
        - scan and clear agi unlinked lists...
error following ag 0 unlinked listopad
found inodes not in the inode allocation tree
found inodes not in the inode allocation tree
found inodes not in the inode allocation tree
        - process known inodes and perform inode discovery...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
        - agno = 4
        - agno = 5
        - agno = 6
        - agno = 7
        - process newly discovered inodes...
Phase 4 - check for duplicate blocks...
        - setting up duplicate extent list...
        - check for inodes claiming duplicate blocks...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
        - agno = 4
        - agno = 5
        - agno = 6
        - agno = 7
Phase 5 - rebuild AG headers and trees...
        - resets superblock...
Phase 6 - check inode connectivity...
        - resetting contents of realtime bitmap and summary inodes
        - ensuring existence of lost+found directory
        - traversing filesystem starting at / ...
        - traversal finished ...
        - traversing all unattached subtrees ...
        - traversal finished ...
        - moving disconnected inodes to lost+found ...
Phase 7 - verify and correct link ocunts...
Note - stripe unit (0) and width (0) fields have been reset.
Please set with mount -o sunit=<value>,swidth=<value>
```

Nie należy się przejmować ilością komunikatów o błędach prezentowanych w przykładzie — specjalnie uszkodziłem system plików ręcznie, aby sprowokować taki wydruk. Proste odłączenie wtyczki zasilania nie spowoduje takich użytkownicy XFS powinni się ucieszyć na wieść, że wprowadzenie takich uszkodzeń nie jest sprawą łatwą.