Archiwizacja i odzyskiwanie danych

Autor: W. Curtis Preston
Tłumaczenie: Piotr Pilch, Marek Pętlicki
ISBN: 978-83-246-1182-9
Tytuł oryginału: Backup & Recovery
Format: B5, stron: 768

Optymalizacja procesu archiwizacji dla administratorów i nie tylko

- Jak archiwizować i odtwarzać system oraz dane?
- Jak wybrać optymalną metodę archiwizacji?
- Jak ograniczyć koszty związane z procesem archiwizacji?

Wdrożenie systemu archiwizacji jest podstawową czynnością, jaką administrator powinien wykonać. Cel tej procedury jest oczywisty, tak jak cena, jaką przyjdzie zapłacić za brak takiego systemu. Utrata danych lub krach systemu wcale nie muszą oznaczać dla firmy długiego postoju i narażać jej na wielkie straty; wtedy to właśnie wysiłek, jaki włożył administrator w system archiwizacji, jest doceniany przez pracodawców i współpracowników. Jakie urządzenia i narzędzia należy zastosować, aby ten proces był sprawny i szybki, a jednocześnie zmieścił się w wyznaczonym do tego celu budżecie? Opisane oprogramowanie i rady udzielone przez autora tej książki pomogą Ci dokonać właściwego wyboru.

„Archiwizacja i odzyskiwanie danych" to przewodnik po darmowych narzędziach do archiwizacji i odzyskiwania danych, przeznaczony głównie dla administratorów. W. Curtis Preston, specjalista w dziedzinie ochrony danych, zwrócił uwagę, że środki przeznaczone na system archiwizacji są często zbyt małe, by zapewnić mu wysoką skuteczność. Przyglądając się bliżej bezpłatnym narzędziom, odkrył ich duże możliwości; swoje spostrzeżenia i uwagi na ich temat zapisał w tej książce. Prędzej czy później każdy administrator staje przed koniecznością odbudowania systemu lub odzyskania danych i właśnie lektura tej książki sprawi, że będzie na to przygotowany!

- Narzędzia do archiwizacji i odtwarzania danych
- Przegląd programów komercyjnych i darmowych
- Urządzenia archiwizujące
- Przywracanie komputera od podstaw z wybranym systemem operacyjnym
- Archiwizacja i odtwarzanie baz danych
- Sposoby zabezpieczania archiwum i magazynów danych
- Wykorzystanie wirtualnych maszyn w procesie archiwizacji

Poszerz swoją wiedzę i archiwizuj dane we właściwy, a także sprawdzony sposób!

Wydawnictwo Helion
ul. Kościuszki 1c
44-100 Gliwice
tel. 032 230 98 63
e-mail: helion@helion.pl

Spis treści

ROZDZIAŁ 3.

Podstawowe narzędzia do archiwizacji i odtwarzania

Podstawowe narzędzia archiwizujące to takie, na których bazują wszystkie niekomercyjne systemy archiwizowania. Narzędzia takie realizują ważne zadanie, którym jest kopiowanie danych z jednego miejsca w drugie, zwykle z użyciem innego formatu docelowego (na przykład program *tar*). Żadne z tych narzędzi nie ma wbudowanych możliwości planowania ani katalogowania sporządzonych kopii zapasowych w celu ich rejestrowania. Aby wykonywać takie zadania, trzeba skorzystać z określonego typu aplikacji zewnętrznej i planującej. Może to być prosty skrypt wsadowy i zaplanowane zadanie (system Windows), skrypt powłoki i wpis narzędzia *cron* (systemy Unix i Mac OS) lub jedno z zaawansowanych narzędzi open source zaprezentowanych w dalszej części książki.

Do podstawowych narzędzi archiwizujących należy zaliczyć wersje programów: *dump*, *cpio*, *tar* i *dd* wbudowane w systemy uniksowe, narzędzia *ntbackup* i *Przywracanie systemu* w systemie Windows, program *ditto* systemu Mac OS, a także wersje GNU narzędzi: *tar*, *cpio* i *rsync*, dostępne dla wszystkich wymienionych platform. Niezależnie od tego, czy ktoś dopiero wkracza w świat archiwizacji czy jest doświadczonym administratorem systemów, musi zaznajomić się z tymi narzędziami.

Przegląd

W rozdziale omówiono zalety i wady kilku narzędzi. W przypadku wszystkich wersji systemu Windows począwszy od wersji NT program *ntbackup* jest jedynym wbudowanym narzędziem pozwalającym archiwizować dane w tradycyjny sposób. Jednak warto się też zapoznać z narzędziem *Przywracanie systemu*. Użytkownicy systemu Mac OS X w wersji nowszej niż 10.4 mają do dyspozycji kilka uniksowych narzędzi archiwizujących, takich jak *cpio*, *tar*, *rsync* i *ditto*. Choć w przypadku komercyjnych systemów Unix dość popularne są narzędzia *dump* i *restore*, w systemie Linux nie są już tak polecane. Narzędzie *dump* jest dostępne w systemie Mac OS, lecz nie obsługuje systemu plików HFS+. Wbudowane narzędzie *cpio* jest kolejnym po *dump* i *restore*, które oferuje większość funkcji. Jednak jest mniej przystępne w obsłudze od swojego kuzyna, czyli programu *tar*. *tar* jest zadziwiająco prosty w użyciu i bardziej przenośny od narzędzi *dump* i *cpio*. Wersje GNU programów *tar* i *cpio* mają znacznie większą funkcjonalność niż wersje wbudowane w system. Jeśli za pomocą narzędzia *tar* lub *cpio* trzeba zarchiwizować

urządzenia o dostępie bezpośrednim lub utworzyć zdalne kopie zapasowe, najlepszym nowym przyjacielem stanie się program *dd*. Narzędzie *rsync* może być użyte do kopiowania danych między systemami plików takich systemów operacyjnych, jak Windows, Mac OS, Linux i Unix.

Niniejszy rozdział rozpoczyna się od przeglądu każdego z tych narzędzi. W dalszej kolejności jest szczegółowo omawiana składnia każdego polecenia zarówno dla procesu archiwizowania, jak i odtwarzania. Na końcu rozdziału zamieszczono bezcenną tabelę, która może pełnić rolę szybkiego przewodnika pozwalającego porównać narzędzia: *tar*, *cpio* i *dump*.

Jak nie używać programu dump

Pewnego razu wybrałem się do klienta, aby usunąć problem z pocztą elektroniczną. Okazało się, że nie miało to nic wspólnego z pocztą, lecz z serwerem DNS. Klient poprosił mnie również, abym przyjrzał się jego kopiom zapasowym. To, co ujrzałem, było przerażające. Kopie były sporządzane przez wykonywanie poleceń uruchamiających program *dump* bez użycia narzędzia *cron*.

- Klient nie utworzył skryptu. Po prostu z różną częstotliwością wykonywał kolejne polecenia ładujące program *dump*. Następne polecenia były uaktywniane, zanim poprzednie zakończyły działanie.
- Klient stosował sterownik przewijający taśmę.
- Klient był zdziwiony, że wszystko mogło zmieścić się na jednej taśmie!

Zróżnicowanie systemów plików systemu Mac OS

Poniższy fragment poświęcony archiwizowaniu w systemie Mac OS powstał przy udziale Leona Towns-von Staubera (autor rozdziału 14.).

To, co może utrudnić archiwizowanie w przypadku systemu Mac OS X, jest domyślnym formatem jego wbudowanego systemu plików HFS+, będącego zaawansowaną wersją starszego systemu plików Macintosh Hierarchical File System. Między systemami plików HFS+ i UFS (*Unix File System*) występują znaczące różnice, dotyczące między innymi obsługi *rozwidleń* (wiele zestawów danych powiązanych z pojedynczym plikiem) i *specjalnych atrybutów pliku* (na przykład typ, twórca i data utworzenia). Choć system Mac OS X może działać z systemem plików UFS, jest on znacznie rzadziej stosowany od powszechnie używanego systemu plików HFS+. Poza tym system plików UFS nie jest dobrze obsługiwany przez firmę Apple i innych producentów oprogramowania.

Narzędzie niezaprojektowane pod kątem obsługi tych unikatowych funkcji systemu plików HFS+ może spowodować, że magazynowane kopie zapasowe będą pozbawione ważnych rozwidleń i atrybutów. Tym samym pełne odtworzenie danych będzie niemożliwe. Największym problemem jest rozwidlenie zasobów, czyli zestaw dodatkowych danych powiązanych z wieloma rodzajami plików systemu Macintosh. Pomimo tego, że od czasu pojawienia się systemu Mac OS X firma Apple odradza stosowanie rozwidleń zasobów, w dalszym ciągu wiele aplikacji używa ich do przechowywania informacji, takich jak ikony miniatur plików obrazów. Nawet sama firma Apple korzysta z takich rozwidleń, aby zapisywać zawartość aliasów, które są graficznymi odpowiednikami dowiązań symbolicznych.

Przed pojawieniem się systemu Tiger (Mac OS X 10.4) nawet standardowe uniksowe narzędzia ignorowały rozwidlenia i atrybuty systemu Macintosh. Jeśli korzysta się z systemu Mac OS X w wersji 10.3 lub starszej bez zewnętrznych narzędzi, najlepiej zastosować program *CpMac* (zgodny z systemem plików HFS+ odpowiednik narzędzia *cp* dodany do pakietu Developer Tools), *ditto* (rekursywne narzędzie kopiujące obsługujące rozwidlenia zasobów i atrybuty systemu plików HFS+, gdy użyje się znacznika `-rsrc`) lub *asr* (*Apple System Restore*; narzędzie powielające woluminy).

Ze względu na trudność wykonywania kopii zapasowych w systemie Mac OS X przed pojawieniem się jego wersji Tiger w internecie udostępniono kilka przeznaczonych dla tego systemu odmian standardowych narzędzi archiwizujących, takich jak *hfstar*, *xtar*, *hfspax*, *rsync_hfs* i *psync*. Dla narzędzi tych utworzono graficzne nakładki (na przykład *RsyncX*, *PsyncX* i *Carbon Copy Cloner*). Aplikacje zgodne z wieloma platformami, takie jak *Amanda* i *BackupPC*, w celu umożliwienia wykonywania kopii zapasowych danych systemu plików HFS+ również korzystały z tych narzędzi.

cpio

cpio może być narzędziem archiwizującym o bardzo dużych możliwościach. Jego najważniejszą funkcją jest możliwość pobierania ze standardowego wejścia listy plików przeznaczonych do archiwizowania. Jest to jedyne wbudowane narzędzie, które na to pozwala. Funkcję tę można wykorzystać jednocześnie z plikami narzędzia *touch* i poleceniem `find`, aby móc utworzyć przyrostowe kopie zapasowe.

Jednak w przeciwieństwie do programu *dump* narzędzie *cpio* nie pozwala wykonać następujących operacji:

- sporządzanie przyrostowych kopii zapasowych bez użycia plików narzędzia *touch* i polecenia `find`;
- pozostawianie bez zmian wartości *atime* i *ctime* po zakończeniu archiwizacji (należy zapoznać się z punktem „Nie należy zapomnieć o uniksowych wartościach mtime, atime i ctime" z rozdziału 2.);
- wykonywanie interaktywnej operacji odtwarzania (na przykład za pomocą opcji `-i` polecenia `restore`).

Dlaczego narzędzie cpio nie jest bardziej popularne?

Jeśli narzędzie *cpio* ma tak duże możliwości, dlaczego program *tar* cieszy się większym powodzeniem? Jednym z powodów jest to, że wykonanie tych samych podstawowych operacji jest znacznie prostsze (i bardziej ustandaryzowane) w przypadku programu *tar*. Przykładowo każda wersja narzędzia *tar* umożliwia wykonanie poleceń `tar cf` *`nazwa_urządzenia`* i `tar xf` *`nazwa_urządzenia`*, a program *cpio* czasami obsługuje opcje `-I` i `-O`, a czasami nie. Jeżeli zsumuje się wszystkie opcje programu *cpio* dostępne dla różnych jego wersji, okaże się, że jest ich ponad 40. Jest też kilka argumentów, które używają identycznych liter, lecz mają zupełnie odmienne przeznaczenie w przypadku różnych wersji systemu Unix. Kolejnym powodem większej popularności narzędzia *tar* jest to, że rozwijana jest jego wersja GNU, która łączy w sobie możliwości programu *cpio* i łatwość obsługi narzędzia *tar*.

ditto

Narzędzie *ditto* jest dostępne tylko w systemie Mac OS i normalnie służy do powielania zawartości jednego dysku na innym. W ten sposób wykorzystuje się je w rozdziale 14. Może też być użyte do utworzenia pliku programu *ZIP* lub *cpio*. Ponieważ narzędzie *ditto* zastosowano w książce i jest powszechnie używane w środowiskach z systemem Mac OS, uwzględniono je w niniejszym rozdziale.

dd

Polecenie `dd` to narzędzie, którego większość osób nie wykorzystuje. Jest to niskopoziomowe polecenie zaprojektowane z myślą o kopiowaniu bitów informacji z jednego miejsca w drugie. Narzędzie w ogóle nie zna struktury kopiowanych danych, ponieważ jej nie wymaga. A zatem, w przeciwieństwie do narzędzi: *dump*, *tar* i *cpio*, nie jest stosowane do kopiowania grupy plików na wolumin archiwizacyjny. Program *dd* może skopiować pojedynczy plik, jego fragment, niesformatowaną partycję lub jej część. Narzędzie to może nawet skopiować dane ze standardowego wejścia `stdin` do standardowego wyjścia `stdout`, modyfikując je po drodze. Choć program *dd* może skopiować plik, podczas wykonywania tej operacji nie zna nazwy pliku lub jego zawartości. Narzędzie po prostu kopiuje bajty z miejsca określonego przez użytkownika, a następnie umieszcza je w ustalonej lokalizacji.

Choć narzędzie *dd* jest raczej proste, jest wyjątkowo elastyczne. Może kopiować pliki lub partycje niezależnie od ich formatu. Jest też w stanie przenieść dane między dwoma różnymi platformami (przykładem translacja EBCDIC na ASCII lub Big Endian na Little Endian). Formaty zapisu danych Big Endian i Little Endian szczegółowo objaśniono w rozdziale 23. Doskonałym przykładem elastyczności narzędzia *dd* jest skrypt archiwizujący bazę danych Oracle zamieszczony w rozdziale 16. Dane bazy Oracle mogą być przechowywane w plikach systemu plików lub na niesformatowanych partycjach dyskowych. Ponieważ skrypt nie może przewidzieć, jaką konfigurację zastosuje każdy administrator baz danych, korzysta z programu *dd*, który jest w stanie skopiować zarówno pliki, jak i niesformatowane partycje. Dzięki temu administrator baz danych może zastosować taką konfigurację, która przedstawia dla niego największą wartość. Skrypt automatycznie zarchiwizuje dowolną konfigurację. Skrypt utworzy nawet kopię zapasową mieszanej konfiguracji, w przypadku której część danych znajduje się w plikach, a część na niesformatowanych partycjach.

dump i restore

Wielu uważa, że narzędzia *dump* i *restore* oferują największe możliwości spośród uniksowych programów archiwizujących. Spośród funkcji wyróżniających oba narzędzia wymieńmy możliwość archiwizowania plików bez modyfikowania ich daty dostępu i zastosowania minipowłoki w celu interaktywnego wybierania plików, które wcześniej zamierza się odtworzyć. Narzędzia *dump* i *restore* są dość zaawansowanymi poleceniami z prostymi interfejsami, których podstawowe opcje są identyczne w przypadku większości systemów uniksowych. Pojawiło się mnóstwo kontrowersji wokół narzędzia *dump* i tego, czy potrafi poprawnie archiwizować aktywny system plików. Więcej na ten temat można przeczytać w dalszej części rozdziału, w podrozdziale poświęconym narzędziu *dump*.

ntbackup

Jest to jedyne wbudowane narzędzie systemu Windows, które może posłużyć do wykonania kopii zapasowej w tradycyjny sposób. Nie zmienia tego to, że niektórzy pobierają z sieci i używają pod systemem Windows wersji GNU narzędzia *tar* lub *rsync*. Podobnie do narzędzi uniksowych zaprezentowanych w niniejszym rozdziale, program *ntbackup* może archiwizować dane na dysku lub taśmie, a także pozwala określić kilka opcji. Można nawet opcje te zapisać w pliku konfiguracyjnym, a następnie nakazać systemowi Windows, aby użył go podczas uruchamiania programu *ntbackup*. Plik konfiguracyjny umożliwia automatyczne sporządzanie kopii zapasowych za pomocą narzędzia *ntbackup*.

rsync

O narzędziu *rsync* można myśleć jak o darmowej i ładniejszej wersji uniksowego polecenia `rcp`, które może posłużyć do synchronizacji dwóch katalogów, nawet wtedy, gdy znajdują się na oddzielnych komputerach. Podstawowa składnia narzędzia jest w zasadzie taka sama jak polecenia `rcp`. W związku z tym osobom zaznajomionym z tym poleceniem powinno być łatwo zrozumieć narzędzie *rsync*. Dwa z produktów archiwizujących open source przedstawionych w książce używają programu *rsync* wraz z innymi narzędziami, aby zapewnić możliwość archiwizowania i odtwarzania. Z tego powodu w tym rozdziale zostaną omówione podstawowe funkcje tego programu.

Narzędzie Przywracanie systemu

Choć narzędzie *Przywracanie systemu* nie przypomina do końca innych narzędzi przedstawionych w rozdziale, zasługuje na wzmiankę. Od czasu pojawienia się systemu Windows 2000 za pomocą programu *Przywracanie systemu* można tworzyć obraz systemu. Narzędzie wykonuje kopię zapasową kilku krytycznych plików i rejestru, dzięki czemu można przywrócić system do stanu z wybranego momentu w przeszłości.

tar

Atrakcyjność narzędzia *tar* w dużej mierze wynika z łatwości obsługi. Niemal każdy wie, jak odczytać wolumin programu *tar*. Jeśli ktoś nie wie, naprawdę łatwo można mu wytłumaczyć, jak to zrobić. Jeżeli na dysku zapisano plik utworzony za pomocą narzędzia *tar*, a nawet skompresowaną wersję pliku, programy takie jak *WinZip*[1] mogą automatycznie go rozpakować i wyświetlić zawartość (*WinZip* nie obsługuje archiwum sporządzonego przy użyciu programu *cpio*). Narzędzie *tar* jest też bardziej przenośne między uniksowymi platformami niż programy *dump* i *cpio*[2].

---

[1] *WinZip* jest zarejestrowanym znakiem towarowym firmy Nico Mak Computing Inc. Wersję demonstracyjną narzędzia można pobrać pod adresem *http://www.winzip.com*.

[2] W projekcie DJGPP, mającym na celu utworzenie wersji kompilatora *gcc* oraz pakietów narzędzi i programów użytkowych GNU dla systemów MS-DOS i Windows, zastosowano program *cpio* w roli standardowego przenośnego narzędzia archiwizującego, a także przeniesiono wersję GNU narzędzi *cpio* i *tar* do systemów DOS i Windows w postaci 32-bitowych plików wykonywalnych.

Jeśli trzeba szybko wykonać kopię zapasową katalogu lub zestawu plików, pod względem łatwości obsługi program *tar* będzie trudny do pobicia. Jeżeli jednak konieczne jest regularne sporządzanie kopii zapasowych, oczekuje się funkcji, których *wbudowana* wersja narzędzia *tar* jest pozbawiona. Użytkownik między innymi będzie chciał utworzyć przyrostowe kopie zapasowe, pozostawić bez zmian wartość *atime* i mieć pewność, że są odtwarzane odpowiednie uprawnienia i prawa własności plików. Aby te wymagania zostały spełnione, można skorzystać z wersji GNU narzędzia *tar* lub przyjrzeć się programowi *cpio*.

Zawarte dalej omówienie podstawowych narzędzi archiwizujących nie ma za zadanie zastępować ich oficjalnej dokumentacji. Zdecydowanie powinno się zapoznać z dokumentacją każdego z tych narzędzi. W takiej dokumentacji można znaleźć wszystko, począwszy od mniej istotnych informacji, a skończywszy na bardzo ważnych, związanych z konkretnym systemem operacyjnym. W określonych sytuacjach producenci dokumentują jedną lub dwie specjalne funkcje. Zawsze należy być na bieżąco z dokumentacją używanego narzędzia archiwizującego, niezależnie od tego, jakie ono jest.

Inne narzędzia

W tym punkcie zamieszczono listę poleceń, których z różnych powodów nie zostały omówione w książce.

asr

asr (*Apple System Restore*) jest narzędziem do wykonywania obrazów dostępnym tylko w przypadku systemu Mac OS. Program służy przed wszystkim do wykonywania masowych operacji powielania, podobnie jak narzędzie *ghost* stosowane przez użytkowników systemu Windows. *asr* jest narzędziem, które można wykorzystać do bezpośredniego kopiowania danych z jednego dysku twardego na inny lub do tworzenia obrazu zawartości dysku twardego, podobnego do pliku ISO stosowanego w innych systemach operacyjnych. Plik obrazu narzędzia *asr* ma rozszerzenie *.dmg*.

pax

Narzędzie *pax* (*Portable Archive Exchange*) generuje przenośne archiwum zgodne z formatem Archive/Interchange File Format określonym w standardzie IEEE Std. 1003.1-1988. Program potrafi też czytać i zapisywać kilka innych formatów plików, takich jak obsługiwane przez narzędzia *tar* i *cpio*. Narzędzie *pax* jest używane przez program instalacyjny systemu Mac OS. Podobnie do wielu rzeczy w uniksowym świecie program *pax* ma grupę wiernych miłośników, którzy przekonują, że jest najlepszy z możliwych. Jednak narzędzie nie będzie omawiane, ponieważ większość osób nie korzysta z niego.

psync, rsyncx, hfstar, xtar i hfspax

Odkąd system Mac OS X bazuje na jądrze systemu Mach Unix, jest wyposażony w kilka narzędzi uniksowych, takich jak *tar*, *cpio*, *pax*, *cp* i *rsync*. Niestety, wczesne wersje tych narzędzi dla systemu Mac OS nie obsługiwały systemu plików z wieloma rozwidleniami, takiego jak HFS+. Podobnie jest w przypadku wersji GNU programu *tar*.

psync, *rsyncx*, *hfstar*, *xtar* i *hfspax* to narzędzia rozprowadzane wśród społeczności użytkowników systemu Mac OS. Zaprojektowano je w celu obejścia ograniczeń wbudowanych narzędzi systemu Mac OS. Programy *psync* i *rsyncx* napisano tak, aby działały podobnie do narzędzia *rsync* i jednocześnie poprawnie obsługiwały rozwidlenia zasobów. Programy *hfstar* i *xtar* przypominają narzędzie *tar*, a ponadto są zgodne z rozwidleniami zasobów. Program *hfspax* działa jak narzędzie *pax* i obsługuje rozwidlenia zasobów.

Od momentu pojawienia się systemu Mac OS w wersji 10.4.x narzędzia: *tar*, *pax*, *cp* i *rsync* poprawnie obsługują rozwidlenia zasobów używających formatu AppleDouble (według firmy Apple obecnie polecenia te korzystają z tego samego interfejsu API co narzędzie wyszukujące *Spotlight* systemu Mac OS). Gdy plik jest kopiowany w formacie, który nie obsługuje wielu rozwidleń, tak jak format narzędzi *tar* i *cpio*, a nawet systemu plików UFS systemu Mac OS, wyżej wymienione programy skonwertują plik na dwa pliki. Pierwszy plik będzie zawierać *rozwidlenie danych* lub rzeczywiste dane pliku. Drugi plik, określany jako *plik nagłówka*, będzie przechowywał rozwidlenie zasobów oraz informacje wyszukiwania. Pierwszy plik ma nazwę oryginalnego pliku, z kolei na początku pliku nagłówka znajduje się łańcuch `._`.

```
dokument.txt
._dokument.txt
```

Gdy plik z wieloma rozwidleniami jest kopiowany lub przywracany przy użyciu formatu bez obsługi wielu rozwidleń (formaty narzędzi *tar* i *cpio* oraz systemu plików UFS) i konwertowany na format zgodny z wieloma rozwidleniami (system plików HFS+), dwa pliki są zamieniane z powrotem na jeden zawierający rozwidlenie danych i zasobów.

Archiwizowanie i odtwarzanie danych za pomocą narzędzia ntbackup

Polecenie `ntbackup` uaktywnia graficzny interfejs narzędzia *ntbackup*. W przeciwieństwie do wszystkich innych poleceń omówionych w niniejszym rozdziale przy użyciu samego polecenia `ntbackup` nie można wybrać, co ma być archiwizowane. Operację tę trzeba wykonać z poziomu graficznego interfejsu użytkownika. Jednakże wystarczy raz załadować graficzny interfejs, a następnie określić pliki przeznaczone do archiwizacji i zapisać wynik operacji w pliku z rozszerzeniem *.bks*, który później można podać w wierszu polecenia `ntbackup`.

Jak w przypadku innych narzędzi przedstawionych w rozdziale, niniejszy podrozdział nie ma zastępować pliku pomocy dołączonego do programu *ntbackup*. Opisano w niej o wiele więcej opcji niż w tej książce.

Oprócz wybrania plików, które znajdą się w kopii zapasowej, można określić wartości dla kilku innych opcji. Oto one:

- typ kopii zapasowej (normalna, kopia, różnicowa lub codzienna),
- typ lokalizacji docelowej kopii zapasowej (dysk lub taśma),
- ścieżka lokalizacji docelowej (na przykład *f:\plik_kopii.bkf*),
- dołączenie lub nadpisanie danych kopii zapasowych istniejących w lokalizacji docelowej,
- poziom rejestrowania (szczegółowy, podsumowanie lub brak).

Wymienione opcje można określać w wierszu polecenia lub w graficznym oknie narzędzia *ntbackup* przez zapisanie w pliku *.bks*. Jednak ze względu na to, że i tak trzeba załadować graficzny interfejs programu *ntbackup* w celu skonfigurowania go (proces obejmuje powyższe opcje), opcje wiersza polecenia nie będą szczegółowo omawiane. Zamiast tego pokażę, jak spowodować, żeby system Windows automatycznie wykonywał wymagane polecenie.

Tworzenie prostej konfiguracji archiwizowania

W celu sporządzenia prostej kopii zapasowej za pomocą graficznego interfejsu narzędzia *ntbackup* trzeba utworzyć plik opcji archiwizacji, zapisać go, a następnie wczytać podczas przeprowadzania archiwizacji przez uruchomienie polecenia `ntbackup`. Aby uaktywnić graficzny interfejs narzędzia *ntbackup*, z poziomu wiersza poleceń należy wykonać polecenie `ntbackup` lub z menu *Start* wybrać pozycję *Wszystkie programy/Akcesoria/Narzędzia systemowe/Kopia zapasowa*. W obrębie karty *Kopia zapasowa* należy wybrać napędy lub katalogi, które będą archiwizowane. Warto zauważyć, że można również wykonać kopię zapasową stanu systemu (pozycja *System State*).

W dalszej kolejności trzeba wybrać różne opcje związane z archiwizowaniem. Dwie podstawowe opcje to typ kopii zapasowej i jej docelowa lokalizacja. Dostępne są następujące typy kopii: normalna, kopia, przyrostowa, różnicowa i codzienna.

Normalna (domyślna)
: Uwzględnia wybrane pliki i oznacza je jako zarchiwizowane.

Kopia
: Uwzględnia wybrane pliki i nie oznacza ich jako zarchiwizowanych.

Przyrostowa
: Uwzględnia wybrane pliki, jeśli je zmodyfikowano od czasu wykonania ostatniej kopii zapasowej, i nie oznacza ich jako zarchiwizowanych.

Różnicowa
: Uwzględnia wybrane pliki, jeśli zostały zmodyfikowane od czasu wykonania ostatniej kopii zapasowej, i nie oznacza ich jako zarchiwizowanych.

Codzienna
: Uwzględnia wyłącznie pliki zmodyfikowane bieżącego dnia.

Aby zastosować coś innego niż normalna kopia zapasowa, z menu *Narzędzia* należy wybrać pozycję *Opcje*, a następnie uaktywnić kartę *Typ kopii zapasowej*. Po otwarciu okna dialogowego *Opcje* należy przejrzeć pozostałe karty i zdecydować, czy ustawić jakieś inne opcje. W celu zamknięcia okna należy kliknąć przycisk *OK*.

Trzeba następnie zdecydować, czy skorzystać z dysku czy z taśmy. Dysk jest prawdopodobnie najlepszą opcją w przypadku prostej kopii zapasowej, zwłaszcza gdy po prostu mają być archiwizowane dane udziału, który będzie archiwizowany przez inny proces. Dalej trzeba określić nazwę pliku kopii zapasowej. Po wykonaniu tych czynności z menu *Zadanie* należy wybrać pozycję *Zapisz wybory jako* i zapisać opcje w pliku (na przykład *c:\moja_kopia.bks*).

Sporządzanie prostej kopii zapasowej

W celu zastosowania ustalonej konfiguracji archiwizacji można wybrać trzy warianty. Pierwszy polega po prostu na kliknięciu przycisku *Rozpocznij* w graficznym oknie narzędzia *ntbackup*. Jeśli opcje zapisano w pliku, można również zainicjować tworzenie kopii zapasowej z poziomu wiersza poleceń. Poniższe polecenie zakłada, że nie wybrano żadnych dodatkowych opcji i określono jedynie, które pliki znajdą się w kopii zapasowej. Wszystkie istotne opcje w postaci argumentów są podane w wierszu polecenia `ntbackup`. Polecenie archiwizuje wybrane pliki, wyszczególnione w pliku *C:\moja_kopia.bks*, nadaje zadaniu nazwę *Codzienna kopia zapasowa* i zapisuje dane w pliku kopii zapasowej *F:\kopia.bkf*.

```
ntbackup backup "@C:\moja_kopia.bks" /M normal /J "Codzienna kopia zapasowa" /F
    "F:\kopia.bkf"
```

Następną czynnością jest zdefiniowanie zaplanowanego zadania zawierającego powyższe polecenie. Jeśli ktoś woli, żeby system Windows sam ustawił wszystkie opcje wiersza poleceń, może po prostu użyć kreatora programu *ntbackup*, który utworzy zaplanowane zadanie. Po uruchomieniu narzędzia *ntbackup* należy uaktywnić kartę *Planowanie zadań*, zaznaczyć w kalendarzu datę i kliknąć przycisk *Dodaj zadanie*. Dalej w oknie dialogowym *Co ma zawierać kopia zapasowa* należy zaznaczyć pozycje, które mają zostać zarchiwizowane. Następne okno prosi o określenie docelowego katalogu i pliku. W kolejnym pojawi się prośba o wybranie typu kopii zapasowej. Dalej zostanie wyświetlonych kilka opcji, w tym opcja pozwalająca zdecydować, czy po zakończeniu archiwizacji dane mają być sprawdzane, czy nie. Można następnie określić, czy kopia zapasowa powinna zostać dołączona do kopii już istniejących w miejscu docelowym czy ma je nadpisać. Na końcu pojawi się prośba o nadanie nazwy zadaniu i ustalenie harmonogramu jego wykonywania. Gdy zostanie to zrobione, system Windows utworzy zaplanowane zadanie zawierające odpowiednie polecenia. W przypadku przytoczonego przykładu utworzyłem następujące zadanie:

```
C:\WINDOWS\system32\ntbackup.exe backup "@C:\moja_kopia.bks" /a /d "Zestaw utworzony
    2007-06-01 o 19:20" /v:no /r:no /rs:no /hc:off /m normal /j "moja_kopia" /l:s /f
    "C:\kopia.bkf"
```

Program *ntbackup* może posłużyć do archiwizowania i odtwarzania serwera Exchange. Więcej na ten temat można znaleźć w rozdziale 20.

Odtwarzanie za pomocą narzędzia ntbackup

Przy użyciu programu *ntbackup* nie można odtwarzać danych z poziomu wiersza poleceń. Trzeba uruchomić jego graficzny interfejs i uaktywnić kartę *Przywróć i zarządzaj nośnikiem*. W obrębie tej karty pojawi się lista kopii zapasowych znanych programowi *ntbackup*. Po wybraniu dowolnej kopii zapasowej pojawi się drzewo zawartych w niej plików. Można następnie zaznaczyć pliki, które mają być odtworzone, zdecydować, czy zostaną umieszczone w swoim pierwotnym miejscu czy w innym, a także przez kliknięcie przycisku *Rozpocznij przywracanie* nakazać programowi *ntbackup* odtworzenie danych. W dalszej kolejności można określić zaawansowane opcje. Operacja odtwarzania rozpoczyna się po kliknięciu przycisku *OK*. Naprawdę dużo prościej się już nie da!

Zastosowanie narzędzia Przywracanie systemu

Każdemu, kto korzystał z systemu Windows przez dłuższy czas, zdarzyło się zainstalować oprogramowanie, które spowodowało, że system stał się bezużyteczny. Wcześniej w takiej sytuacji jedyną opcją była ponowna instalacja systemu Windows i wszystkich aplikacji. Jednak to się zmieniło wraz z pojawieniem się narzędzia *Przywracanie systemu*. Jeśli możliwe jest załadowanie systemu w trybie awaryjnym i uaktywnienie narzędzia *Przywracanie systemu*, prawdopodobnie będzie można znaleźć stabilną wersję systemu nadającą się do odtworzenia. W efekcie od razu można dysponować działającym systemem operacyjnym!

Program *Przywracanie systemu* różni się trochę od innych narzędzi zaprezentowanych w rozdziale, ponieważ nie tworzy kopii zapasowych w tradycyjny sposób i nie można korzystać z niego w obrębie innego narzędzia. Jednak to narzędzie odtwarzające, dołączone do systemu Windows XP i nowszych, jest bardzo ważne i należy się z nim zaznajomić.

Narzędzie *Przywracanie systemu* dodane do systemu Windows XP i jego następców archiwizuje rejestr systemowy i krytyczne pliki, tworząc *punkt przywracania*. System Windows robi to automatycznie, gdy uzna, że użytkownik wykonuje znaczącą operację, taką jak instalacja nowego sterownika lub większa aktualizacja. Ponadto punkty przywracania można tworzyć samemu zawsze, gdy jest to wskazane, lub za pomocą zaplanowanego zadania automatycznie w regularnych odstępach czasu. Można następnie za pomocą dowolnego z punktów przywracania utworzonych przez system przywrócić go do stanu z wybranego momentu w przeszłości.

Tworzenie punktów przywracania

Jak wspomniano, system Windows generuje za użytkownika wiele punktów przywracania, o ile nie wyłączono narzędzia *Przywracanie systemu*. Aby stwierdzić, czy narzędzie jest aktywne, należy zalogować się jako użytkownik należący do grupy *Administratorzy*, a następnie z menu *Start* wybrać pozycję *Mój komputer* i prawym przyciskiem myszy kliknąć w menu podręcznym pozycję *Właściwości*. Po uaktywnieniu karty *Przywracanie systemu* można włączyć lub wyłączyć narzędzie.

Aby móc skorzystać z narzędzia *Przywracanie systemu*, trzeba być zalogowanym jako użytkownik *Administrator* lub inny członek grupy *Administratorzy*.

Każdy użytkownik należący do grupy *Administratorzy* może w dowolnej chwili utworzyć punkt przywracania. W tym celu z menu *Start* należy wybrać pozycję *Wszystkie programy/Akcesoria/Narzędzia systemowe/Przywracanie systemu*, a następnie kliknąć opcję *Utwórz punkt przywracania*. Okno dialogowe poprosi o podanie nazwy punktu przywracania, który zostanie wygenerowany. Może to być dowolna nazwa, taka jak *Tuż przed instalacją gry Doom*. Następnie system utworzy punkt i nada mu nazwę. Później za pomocą narzędzia *Przywracanie systemu* można przywrócić system Windows do poprzedniego stanu.

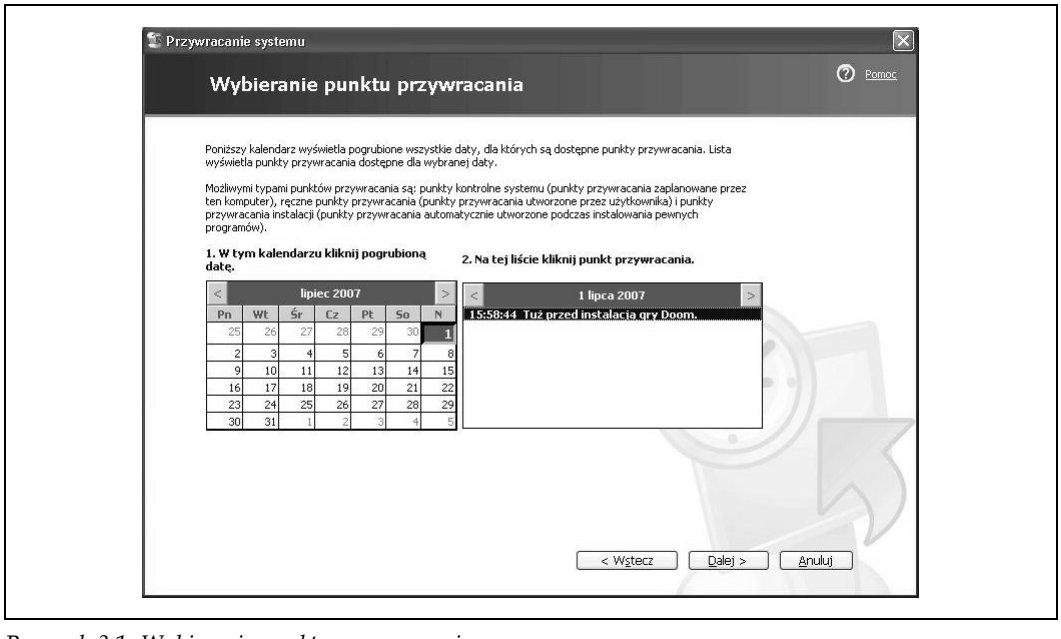
Choć narzędzie *Przywracanie systemu* można również uruchomić przez załadowanie programu *%SystemRoot%\system32\restore\rstrui.exe*, jest raczej mało prawdopodobne, że ktoś to zapamięta.

Jeżeli ktoś nie ufa systemowi Windows, jeśli chodzi o tworzenie przez niego punktów przywracania, ale nie chce mu się ręcznie wykonywać takiej operacji, może utworzyć zaplanowane zadanie, które będzie generować punkt przywracania tak często, jak to będzie potrzebne. W tym celu z menu Start należy wybrać pozycję *Wszystkie programy/Akcesoria/Narzędzia systemowe/Zaplanowane zadania*, a następnie kliknąć *Dodaj zaplanowane zadanie*. W otwartym oknie dialogowym należy kliknąć przycisk *Dalej*, a w następnym — opcję przywracania systemu. Należy określić, jak często zadanie ma być uruchamiane i kiedy, a także podać nazwę i hasło konta użytkownika, będącego członkiem grupy *Administratorzy*. System Windows utworzy następnie punkt przywracania.

Przywracanie systemu Windows za pomocą punktu przywracania

Jeśli wersja używanego systemu Windows stała się niestabilna na skutek ostatniej instalacji aktualizacji lub sterownika, wystarczy uaktywnić narzędzie *Przywracanie systemu*, wybrać poprzedni stan i nakazać systemowi jego przywrócenie. Jeżeli system Windows jest naprawdę niestabilny, najtrudniejsze może być samo jego załadowanie. Najlepsze w tym przypadku może być uruchomienie systemu w trybie awaryjnym i zalogowanie się jako *Administrator*.

Gdy już jakimś sposobem uda się załadować system, z menu *Start* należy wybrać pozycję *Wszystkie programy/Akcesoria/Narzędzia systemowe/Przywracanie systemu*, a następnie kliknąć opcję *Przywróć komputer do wcześniejszego stanu*. Następnie pojawi się okno dialogowe podobne do pokazanego na rysunku 3.1.



Rysunek 3.1. Wybieranie punktu przywracania

Automatycznie jest zaznaczany punkt przywracania z najbardziej aktualną datą widoczną w kalendarzu. Punkty przywracania powiązane z wybraną datą są wyświetlane z prawej strony. Można przywrócić system do stanu z tej chwili lub wybrać wcześniejszą datę, jeśli jest się przekonanym, że również najnowszy punkt przywracania może być podejrzany. Po wybraniu punktu przywracania należy kliknąć przycisk *Dalej*. Oczywiście system Windows poprosi o potwierdzenie wyboru i ostrzeże o konieczności zapisania wszelkich danych i zamknięcia wszystkich otwartych programów, ponieważ operacja odtwarzania wymaga ponownego uruchomienia komputera.

Potem należy klikać jedynie przyciski *Dalej* — do momentu zamknięcia okna dialogowego. Po ponownym załadowaniu systemu należy przetestować odtworzoną wersję systemu Windows, aby się upewnić, czy problemy zostały wyeliminowane. Jeśli tak — to wszystko. W przeciwnym razie po prostu trzeba ponownie przeprowadzić cały proces, aż do znalezienia punktu przywracania, który da pożądane efekty.

Archiwizowanie za pomocą narzędzia dump

W wielu środowiskach program *dump* może być wszystkim, co jest potrzebne do zapewnienia dobrej jakości kopii zapasowych. Istnieje jednak wiele kontrowersji związanych z tym narzędziem, wynikających z tego, że nie uzyskuje dostępu do danych za pośrednictwem systemu plików, przeciwnie niż większość innych narzędzi archiwizujących. *dump* uzyskuje bezpośredni dostęp do urządzenia systemu plików. Właśnie dlatego program może archiwizować pliki bez modyfikowania ich daty dostępu. Również dlatego zawsze w dokumentacji narzędzia *dump* jest zaznaczone, aby odłączyć systemy plików przed rozpoczęciem archiwizowania ich. Oczywiście nikt tego nie robi, stąd kontrowersje.

Narzędzie dump w systemach Mac OS i Linux

Administratorzy Linuksa powinni być świadomi tego, że narzędzie *dump* nie oferuje dobrej metody archiwizowania danych. Program ten nie obsługuje też systemu plików HFS+ systemu Mac OS. Firma RedHat oficjalnie odradza stosowanie narzędzia *dump* w systemie RedHat 9. Poniższy fragment wypowiedzi Linusa Torvaldsa podsumowuje postawę społeczności linuksowej wobec narzędzia *dump*.

„*dump* po prostu nie będzie stabilnie działał nawet w przypadku jądra 2.4.x. Pamięć podręczna buforowania i stronicowania (znajdują się w niej wszystkie rzeczywiste dane) nie są spójne. Sytuacja tylko się pogarsza w przypadku jądra 2.5.x, gdy do pamięci podręcznej buforowania są również przenoszone katalogi. W związku z tym każdy, kto jest zależny od poprawnego sporządzania kopii zapasowych przez narzędzie *dump* (dotyczy to systemu Linux), gra w rosyjską ruletkę, w której stawką są kopie. W ogóle nie ma gwarancji, że uzyska się poprawne kopie zapasowe. Może się okazać, że w pamięci podręcznej buforowania będą nieaktualne dane, które ostatecznie zostaną »zarchiwizowane«… Program *dump* może świetnie zadziałać tysiące razy. Jednak zawiedzie w prostych okolicznościach. I nie ma się na to żadnego wpływu".

Choć trzeba będzie samemu zdecydować, czy narzędzie *dump* jest dobre, czy nie, zdaje się, że nie jest najlepszą opcją archiwizowania danych w systemie Linux.

Choć narzędzia *dump* i *restore* są dostępne dla systemu Mac OS, współpracują jedynie z systemem plików UFS. Nie istnieje żadne narzędzie takie jak *hfsdump* zgodne z systemem plików HFS+. Z tego, co wiem, nie ma planów stworzenia takiego narzędzia.

Aby użyć narzędzi *dump* i *restore* do regularnego wykonywania kopii zapasowych danych systemowych, trzeba rozumieć następujące zagadnienia:

- zastosowanie narzędzia *dump* do archiwizowania systemu plików (przy wykorzystaniu odpowiednich opcji);
- sposób finalizowania kopii zapasowej na woluminie;
- uzyskiwanie tabeli zawartości woluminu narzędzia *dump*;
- obsługę woluminu i odtwarzanie danych z kopii zapasowej utworzonej za pomocą narzędzia *dump*;
- ograniczenia narzędzi *dump* i *restore*;
- operacje, które powinny być wykonane, gdy zamierza się regularnie korzystać z programu *dump*.

W pierwszej kolejności trzeba zapoznać się z opcjami i przeznaczeniem polecenia `dump`. W tabeli 3.1 zebrano polecenia `dump` stosowane w różnych wersjach systemu Unix. Poniższy punkt właściwie pełni rolę zunifikowanej dokumentacji tych poleceń, przeznaczonych dla konkretnych systemów operacyjnych.

Tabela 3.1. Polecenia dump stosowane w przypadku różnych wersji systemu Unix

Wersja systemu Unix	Polecenie
HP-UX 9.x/HP-UX 10/SunOS/IRIX	`(r)dump`
Solaris	`ufsdump`
SCO	`xdump`
Network Appliance	`dump`
AIX	`backup` i `rdump`
Linux	`dump`
SGI	`dump` i `xfsdump`
Tru64 Unix	`dump` i `vdump`
Linux/Mac OS	Należy zapoznać się z treścią ramki „Narzędzie dump w systemach Mac OS i Linux".

Choć w systemie Mac OS jest dostępne polecenie `dump`, nie obsługuje ono systemu plików HFS+, który jest najczęściej stosowany.

Składnia polecenia dump

Rozpocznijmy od podstawowej składni polecenia `dump`:

```
# dump poziomunbdsf współczynnik_bloków gęstość rozmiar nazwa_urządzenia system_plików
```

Oto praktyczne przykłady powyższego polecenia:

- Aby utworzyć pełną kopię zapasową katalogu */home* i zapisać ją na lokalnym napędzie taśmowym */dev/rmt/0cbn*, należy zastosować następujące polecenie:

```
# dump 0unbdsf 126 141000 11500 /dev/rmt/0cbn /home
```

- W celu sporządzenia pełnej kopii zapasowej */backup/home.dump* katalogu */home* i umieszczenia jej na urządzeniu optycznym lub dysku CD należy wykonać poniższe polecenie:

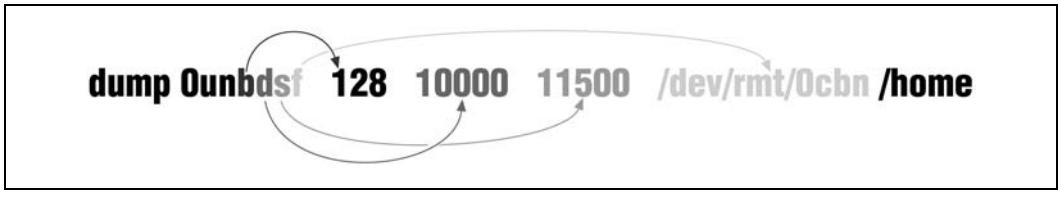
```
# dump 0unbdsf 126 141000 11500 /backup/home.dump /home
```

- Aby utworzyć pełną kopię zapasową katalogu */home* na zdalnym napędzie taśmowym */dev/rmt/0cbn* komputera *elvis*, należy użyć następującego polecenia:

```
# (r)dump 0unbdsf 126 141000 11500 elvis:/dev/rmt/0cbn /home
```

W powyższych poleceniach zastosowano trzy opcje (`0`, `u` i `n`), które nie wymagają argumentów, i cztery opcje (`b`, `d`, `s` i `f`) wymagające dołączenia argumentu.

Jako swój pierwszy argument polecenie `dump` pobiera listę opcji. Argument każdej z tych opcji jest wstawiany w wierszu polecenia w takiej samej kolejności, w jakiej wymieniono opcje. Rysunek 3.2 ilustruje, w jaki sposób opcje polecenia `dump` są powiązane ze swoimi argumentami.



Rysunek 3.2. Proste polecenie dump

Opcje polecenia dump

Narzędzie *dump* ma siedem głównych opcji dostępnych w przypadku większości platform. Oto one:

`0 – 9`
: Określa poziom operacji archiwizacji, która powinna zostać wykonana przez program *dump*.

`b`
: Określa współczynnik bloków, który powinien zostać użyty przez narzędzie *dump*.

`u`
: Opcja nakazuje programowi *dump* uaktualnić plik *dumpdates*.

`n`
: Nakazuje narzędziu *dump* powiadomić członków grupy operatorów, gdy zakończy działanie.

`d` *i* `s`
: Opcja informuje program *dump* o rozmiarze woluminu archiwizacyjnego. Na podstawie wartości tych opcji *dump* szacuje dostępną pojemność „taśmy".

`f`
: Opcja przekazuje programowi *dump* dane na temat urządzenia, które ma być użyte.

`W, w`
: Opcje nakazują programowi *dump* przeprowadzenie próbnej archiwizacji mającej na celu stwierdzenie, jakie systemy plików wymagają utworzenia kopii zapasowej (opcje są rzadko stosowane).

Jeśli przy użyciu narzędzia *dump* regularnie tworzy się systemowe kopie zapasowe, powinno się korzystać z większości wymienionych opcji. Godne uwagi jest to, że wiele z tych opcji ma domyślne wartości. Dzięki temu eliminuje się konieczność określania w wierszu polecenia `dump` opcji i jej argumentu. Przykładowo domyślny poziom archiwizacji zwykle wynosi `9`. Problem z domyślnymi wartościami jest taki, że są inne w różnych systemach operacyjnych, a nawet w obrębie tego samego systemu (zależnie od takich czynników jak typ nośnika). W celu późniejszego uproszczenia sobie procesu odtwarzania lepiej każdą z opcji określać w taki sam sposób w przypadku wszystkich kopii zapasowych tworzonych przy użyciu narzędzia *dump*.

Definiowanie pełnej lub przyrostowej kopii zapasowej (0 – 9)

Pierwszym argumentem, który można określić jest poziom archiwizacji. Można użyć dowolnej wartości z przedziału od 0 do 9 (w rozdziale 2. objaśniono poziomy archiwizacji). Przyrostowe kopie zapasowe narzędzia *dump* odwołują się do pliku *dumpdates* w celu określenia daty sporządzenia ostatniej kopii niższego poziomu (plik omówiono w dalszej części rozdziału w podpunkcie „Aktualizowanie pliku dumpdates (u)"). Jeśli na przykład wykonuje się kopię zapasową poziomu 5, program *dump* archiwizuje wszystkie pliki, które zostały zmodyfikowane od czasu sporządzenia ostatniej kopii poziomu 4 lub niższego. Narzędzie pobiera datę utworzenia tej kopii zapasowej z pliku *dumpdates* (zwykle znajduje się w katalogu */etc*). Ponieważ plik *dumpdates* jest wymagany przez przyrostowe kopie zapasowe, trzeba użyć opcji `u` w celu jego aktualizacji.

Narzędzia dump i restore zapobiegły katastrofie

Był to długi i ciężki tydzień. Próbowaliśmy zakończyć kilka spraw, tak aby móc jechać do domu. Właśnie wtedy odebraliśmy telefon. *Zawsze* w takich chwilach dzwoni telefon. Okazało się, że zniknął z systemu bardzo ważny katalog zawierający rzadko używane, lecz istotne narzędzie. „Żaden problem. Mamy to na taśmie" — odpowiedziałem. Gdy rozpocząłem przygotowania do odtworzenia plików, zdałem sobie sprawę, że katalog nie istniał już od jakiegoś czasu. W rzeczywistości nie było go w systemie na tyle długo, że nie został zarchiwizowany przez stosowane przez nas komercyjne narzędzie. Można sobie wyobrazić, jak się wtedy poczułem.

Przyjrzałem się starej szafie na dokumenty, w której przechowywano kiepsko uporządkowane, nieodpowiednio oznaczone i niemal zapomniane taśmy z kopiami zapasowymi wykonanymi za pomocą narzędzia *ufsdump*. W tamtej chwili taśmy te były najważniejsze na świecie, ponieważ zostały zapisane, zanim zaczęto korzystać z komercyjnego oprogramowania archiwizującego. Umieściłem kolejno taśmy w napędzie i zastosowałem opcję `table of contents` narzędzia *ufsrestore*, mając nadzieję, że jedna z taśm okaże się tą właściwą. Stos taśm stawał się coraz niższy. W końcu jedna z taśm wydała się przydatna. Przełączyłem tryby, korzystając z opcji `interactive`, i okazało się, że mam szczęście. Zaznaczyłem katalog i odtworzyłem go. Katalog został zapisany w systemie, a klient nigdy się nie dowiedział, że niemal nie byliśmy w stanie odtworzyć danych. Był to ten dzień, w którym byłem naprawdę zadowolony, że zaznajomiłem się z narzędziami *dump* i *restore* (dowiedziałem się również, jak ważne jest comiesięczne archiwizowanie pełnych kopii zapasowych).

Określanie współczynnika bloków (b)

Opcja `b` określa liczbę bloków, które zostaną zapisane w ramach jednej operacji na wyjściu. Opcja odwołuje się do *liczby* fizycznych bloków. *Rozmiar* całego bloku zapisywanego przez narzędzie *dump* zależy od wielkości fizycznego bloku pomnożonej przez współczynnik bloków. W przypadku większości wersji systemu Unix rozmiar fizycznego bloku używanego przez narzędzie *dump* wynosi 1024 bajty. A zatem, jeśli określi się współczynnik bloków o wartości 10, rozmiar rzeczywistego bloku zapisywanego przez program *dump* wyniesie 10 240 bajtów lub 10 kilobajtów. Opcja `b` nie jest dostępna w systemie SCO.

Przynajmniej jedna odmiana systemu Unix umożliwia zmianę współczynnika bloków na potrzeby narzędzia *dump*, lecz nie *restore*. Oznacza to, że za pomocą narzędzia *dump* można tworzyć woluminy, których nie da się odczytać! Trzeba się upewnić, że używana wersja narzędzia *restore* pozwala zmodyfikować współczynnik bloków.

Aktualizowanie pliku dumpdates (u)

Opcja `u` powoduje, że narzędzie *dump* aktualizuje plik *dumpdates* dla archiwizowanego systemu plików (choć plik ten zwykle znajduje się w katalogu */etc*, w systemie HP-UX 10.x umieszczono go w katalogu */var/adm*). Jest to zwykły plik tekstowy wyszczególniający urządzenie każdego systemu plików i datę sporządzenia dla tego urządzenia ostatniej kopii zapasowej na każdym poziomie. Oto przykładowa zawartość pliku */etc/dumpdates* z systemu Solaris:

```
/dev/rdsk/c0t1d0s0          0  Sun  Apr  30  23:07:22 2006
/dev/rdsk/c0t1d0s0          1  Wed  May  3   02:49:51 2006
/dev/rdsk/c0t3d0s0          0  Sat  May  20  00:31:49 2006
/dev/rdsk/c0t3d0s0          1  Mon  May  29  01:33:33 2006
/dev/rdsk/c0t3d0s0          5  Wed  May  31  00:28:14 2006
```

Jak widać, dla urządzenia `c0t1d0s0` 30 kwietnia 2006 r. wykonano kopię zapasową poziomu 0, a 3 maja — poziomu 1 Dla urządzenia `c0t3d0s0` 20 maja sporządzono kopię zapasową poziomu 0, 29 maja — poziomu 1, a 31 maja — poziomu 5.

W związku z plikiem *dumpdates* trzeba wspomnieć o kilku ważnych rzeczach. Gdy narzędzie *dump* uruchamiane jest w systemie po raz pierwszy, konieczne jest utworzenie pustego pliku *dumpdates* i ustanowienie jego właścicielem superużytkownika. Jeśli plik nie istnieje lub jego właścicielem nie jest superużytkownik, narzędzie *dump* nie utworzy pliku. Program będzie kontynuował działanie, lecz poinformuje o problemie z plikiem *dumpdates*. Warto zauważyć, że plik ten jest aktualizowany tylko wtedy, gdy narzędzie *dump* całkowicie z powodzeniem ukończy tworzenie kopii zapasowej. Jeżeli jakieś błędy spowodują przerwanie działania narzędzia *dump*, plik *dumpdates* nie zostanie uaktualniony. Oznacza to, że plik ten dobrze nadaje się do zastosowania przez zautomatyzowany skrypt sprawdzający, czy narzędzie *dump* utworzyło kopie zapasowe. Poniższa lista prezentuje różne nazwy i lokalizacje pliku *dumpdates*.

- HP-UX 9.x, SunOS, Solaris, AIX, Linux i IRIX: */etc/dumpdates*.
- HP-UX 10.0: */var/adm/dumpdates*.
- SCO: */etc/ddate*.

Można nie chcieć używać opcji `u`, gdy jednorazowo tworzy się specjalną kopię zapasową. Wynika to stąd, że zastosowanie opcji `u` ma wpływ na inne kopie zapasowe. Jeśli na przykład sporządza się jednorazowo kopię poziomu 0 z użyciem opcji `u`, automatycznie tworzone kopie zapasowe poziomu 1 będą odwoływać się do kopii poziomu 0, którą przekazano komuś innemu i która nie wchodzi w skład normalnej puli archiwizacyjnej.

Plik *dumpdates*, niezależnie od swojej nazwy, może być przeglądany lub modyfikowany za pomocą standardowego edytora tekstu. Przykładowo można coś takiego zrobić, gdy wiadomo, że taśma z kopią zapasową poziomu 0 z bieżącego tygodnia została wciągnięta przez napęd. Nawet gdy brakuje czasu potrzebnego do ponownego utworzenia kopii tego poziomu, jakąś kopię zapasową chcemy mieć. Jeśli jednak wykonamy kopię zapasową poziomu 1, będzie ona odwoływać się do kopii poziomu 0 z tego tygodnia, która nie jest dostępna. W takiej sytuacji dla wymaganych systemów plików można poddać edycji wiersz kopii zapasowej poziomu 0, zmieniając datę na datę dostępnej kopii tego samego poziomu, sporządzonej w zeszłym tygodniu. Dzięki temu kopia poziomu 1 będzie odwoływać się do kopii poziomu 0 z poprzedniego tygodnia, a nie do kopii z bieżącego tygodnia, która został zniszczona. W ten sposób mimo uszkodzenia kopii poziomu 0 można spać trochę spokojniej bez konieczności ponownego tworzenia pełnej kopii zapasowej poziomu 0.

Powiadamianie operatorów archiwizacji (n)

Opcja `n` powoduje, że narzędzie *dump* powiadamia każdego członka grupy operatorów (określona w pliku */etc/group*), gdy konieczna okaże się ich interwencja. Powiadomienie wygląda podobnie do komunikatu polecenia `wall` (powiadomienia nie są dostępne w przypadku systemu SCO). Kopia zapasowa tworzona przez narzędzie *dump* może wymagać interwencji, gdy wystąpi jedno z następujących zdarzeń:

- kopia zapasowa osiągnie koniec taśmy lub dojdzie do zapełnienia dysku CD;
- napęd archiwizujący przestanie poprawnie działać, powodując błędy zapisu;
- pojawią się trudności z odczytem przez napęd dyskowy.

Określanie gęstości i rozmiaru (opcje d i s)

Gęstość (opcja `d`) i rozmiar (opcja `s`) nie mają wpływu na to, *jak* dane są zapisywane na nośniku archiwizacyjnym. Polecenie `dump` używa tych opcji tylko do określenia ilości danych, które mogą się zmieścić na konkretnym woluminie, i do identyfikacji logicznego końca taśmy (miejsce, po którego osiągnięciu narzędzie *dump* sądzi, że wolumin jest pełny), występującego przed fizycznym końcem taśmy. Gdy zostaje wykryty logiczny koniec taśmy, program *dump* prosi operatora o zmianę woluminów. Identyfikowanie logicznego końca taśmy ma na celu niedopuszczenie do osiągnięcia przez wolumin fizycznego końca taśmy. Starsze wersje narzędzia *dump* nie radzą sobie dobrze z taką sytuacją. Poniżej w skrócie omówiono opcje `d` i `s`.

d (gęstość)
: Określając gęstość, informuje się program *dump* o ilości danych, które mogą się zmieścić na jednym calu taśmy (choć w rzeczywistości wartość opcji `d` nawiązuje do czasów 9-ścieżkowych taśm, narzędzie *dump* używa jej w połączeniu z opcją `s` do określenia pojemności woluminu archiwizacyjnego). Aby mieć pewność, że program *dump* wykorzysta cały wolumin, należy zastosować dużą wartość, taką jak 80 000.

s (rozmiar „taśmy" wyrażony w stopach)
: Opcja `s` informuje narzędzie *dump* o długości taśmy. Dysponując wartościami opcji `d` i `s`, program oblicza ilość danych, które zmieszczą się na taśmie. Aby mieć pewność, że narzędzie *dump* wykorzysta cały wolumin, należy użyć dużej wartości, takiej jak 500 000. Podając wartość 80 000 jako gęstość i 500 000 jako rozmiar, skutecznie informuje się program *dump*, że wolumin jest w stanie pomieścić 480 GB! Choć opcje `d` i `s` wydają się bezsensowne, jeśli

dane archiwizuje się na dysku lub płycie CD, pełnią istotną rolę. W celu uzyskania dodatkowych informacji należy zapoznać się z podpunktem „Czy trzeba używać opcji d i s?".

W praktyce opcje te są bardzo trudne w użyciu i dają znikome korzyści. Większość osób oszukuje narzędzie *dump*, stosując wartości, które sprawiają, że dla programu taśma ma nieograniczoną pojemność. W efekcie narzędzie używa całego woluminu i jest w stanie wykryć fizyczny koniec taśmy, jeśli aż tyle jej zapisze. Można wymienić wiele powodów takiego stanu rzeczy. Oto one:

- Polecenie `dump` może obecnie wykrywać fizyczny koniec taśmy i odpowiednio reagować (wcześniej program po osiągnięciu fizycznego końca taśmy przerywał działanie). W systemie Solaris jest nawet dostępna opcja powodująca wysunięcie taśmy i włożenie następnej (jeśli stosuje się automatyczną zmieniarkę). A zatem w przypadku systemu Solaris narzędzie *dump* może kontynuować działanie bez potrzeby interwencji operatora.
- Obliczenia sprawdzają się tylko wtedy, gdy narzędzie *dump* umieściło na woluminie dopiero pierwszą kopię zapasową. Przykładowo za każdym razem, gdy stosuje się narzędzie *dump*, informuje się je, że taśma ma długość wynoszącą 10 000 stóp. Jeśli na woluminie zapisano wcześniej przynajmniej jedną kopię, jej długość nie będzie miała już takiej wartości.
- Jeżeli użyje się „rzeczywistych" wartości, prawdopodobnie poda się niewielką gęstość i bardzo duży rozmiar. Wiele wersji systemu Unix informuje, że coś takiego może spowodować problemy (trzeba to potraktować poważnie!).
- Jeśli narzędzie *dump* ma przerwać działanie przed osiągnięciem fizycznego końca taśmy, trzeba podać mniejsze wartości. W efekcie zostanie wykorzystana przestrzeń woluminu mniejsza niż rzeczywista (niekiedy budżet obliguje do zapisywania każdego centymetra wszystkich kupionych woluminów).

Uwzględnienie w obliczeniach kompresji naprawdę komplikuje proces, ponieważ jest ona tym obszarem, w przypadku którego naprawdę ma zastosowanie stwierdzenie: „długość może się zmieniać".

Unikanie tworzenia za pomocą narzędzia dump kopii zapasowej na wielu woluminach

Przez stwierdzenie „na wielu woluminach" rozumiem to, że pojedyncza kopia zapasowa narzędzia *dump* rozpoczyna się na jednym woluminie i kończy w miejscu osiągnięcia logicznego lub fizycznego końca taśmy, a następnie jest kontynuowana na następnym woluminie. Jeśli na przykład dysponuje się napędem taśmowym DDS o pojemności 4 GB oraz archiwizuje 2- i 3-gigabajtowy system plików, pierwsza kopia zapasowa narzędzia *dump* powinna się zmieścić na jednej taśmie. Druga kopia powinna zapełnić resztę pierwszej taśmy i wymagać włożenia drugiej taśmy, aby program *dump* mógł dokończyć archiwizowanie (rysunek 3.3).

Według mnie tworzenie kopii zapasowej w ten sposób jest proszeniem się o kłopoty. Jeśli nie ma wyboru, trzeba tak postąpić, lecz wtedy pojawi się kilka nowych kwestii i zwiększy się trudność operacji odtwarzania. Przykładowo przed załadowaniem taśmy nr 2 trzeba włożyć taśmę nr 1 i rozpocząć jej odczyt. W przypadku odtwarzania danych już to jest wystarczającym utrudnieniem! Ponadto zaczynam się zastanawiać nad poziomem bezpieczeństwa plików znajdujących się przy końcu pierwszej taśmy. Czy Czytelnik uważa, że są bezpieczne? Polecenie `dump` może być czasami zabawne.

Rysunek 3.3. Przykład kopii zapasowej narzędzia dump zapisanej na wielu woluminach

Czy trzeba używać opcji d i s?

W kilku nowszych wersjach narzędzia *dump* zrezygnowano z tych opcji i zaoferowano nową, `size in kilobytes`, która może posłużyć do określenia rozmiaru woluminu wyrażonego w kilobajtach. Pomimo tego sam w wierszu każdego wykonywanego polecenia `dump` używam opcji `s` i `d`, żebym nie musiał pamiętać o różnicach poszczególnych wersji programu *dump*. Czytelnik zauważy, że w książce jest to często przeze mnie praktykowane. Im więcej zadań można wykonać w taki sam sposób, tym mniejszą liczbą rzeczy trzeba się martwić. Im więcej operacji dostosowywania wykonuje się dla każdego komputera i systemu operacyjnego, w tym większe kłopoty można wpaść (przykładowo opcja `size in kilobytes` używa innej litery dla każdej obsługiwanej wersji systemu Unix!). W tym przypadku zastosowanie archaicznych opcji gęstości i rozmiaru w rzeczywistości znacznie ułatwia tworzenie skryptów powłoki, ponieważ w większości wersji systemu Unix można korzystać z tych samych opcji.

Co się stanie, gdy nie użyje się opcji `s`, `d` lub `size in kilobytes`? W przypadku niektórych odmian systemu Unix narzędzie *dump* korzysta z domyślnych wartości opcji gęstości i rozmiaru (z wyjątkiem systemu AIX, w którym całkowicie zrezygnowano z tych opcji). Niestety, domyślne wartości są zwykle ustawiane z myślą o obsłudze 9-ścieżkowych taśm (w systemie Solaris zmieniono domyślne wartości, aby były bardziej rozsądne). W takiej sytuacji narzędzie *dump* będzie przekonane, że potrzebuje kilku woluminów. Wynik wygenerowany przez program *dump* wygląda podobnie do poniższego:

```
DUMP: Estimated 5860 blocks (3006KB) on 39.00 tapes.
```

Warto zauważyć, że narzędzie uważa, że potrzebuje 39 taśm. Właśnie do czegoś takiego może dojść, jeśli do określenia pojemności woluminu nie użyje się opcji gęstości i rozmiaru. Jak wspomniano, z łatwością można wyeliminować taką sytuację przez ustawienie dla tych opcji absurdalnie dużych wartości, tak aby narzędzie *dump* nigdy nie przewidziało, że braknie taśmy (lubię dla obu opcji podawać takie wartości jak 1 000 000).

Określanie pliku urządzenia archiwizującego (f)

Opcja `f` określa nazwę pliku urządzenia archiwizującego, do którego zostaną przesłane dane (oczywiście tym „urządzeniem" może być rzeczywisty napęd taśmowy lub plik znajdujący się na dysku lub nośniku optycznym). Jeśli dla posiadanego napędu taśmowego zamierza się zastosować sprzętową kompresję, trzeba wybrać urządzenie, które ją obsługuje. Jeżeli dane chce się przesłać do napędu innego komputera, należy użyć formatu *nazwa_zdalnego_komputera:urządzenie*. Większość wersji systemu Unix pozwala na współpracę narzędzia *dump* ze zdalnymi urządzeniami, pod warunkiem że poprawnie korzysta się z programu *rsh* w roli mechanizmu uwierzytelniania.

Zastosowanie narzędzia *rsh* i plików */.rhosts* powoduje powstanie poważnej luki w zabezpieczeniach. Wiele organizacji nie zezwala na ich użycie! Nie należy wszędzie tworzyć plików */.rhosts* i później mnie za to obwiniać. Zanim zacznie się korzystać z narzędzia *rsh*, trzeba się dokładnie zorientować, czy organizacja zezwala na to, czy nie. Jeśli nie, można wziąć pod uwagę wykorzystanie narzędzia *ssh* zamiast *rsh*. Więcej informacji można znaleźć w podrozdziale „Zastosowanie programu ssh lub rsh w roli kanału między systemami", zamieszczonym na końcu rozdziału.

Zdalne urządzenia wymagają, aby ich host ufał lokalnemu hostowi wykorzystującemu plik */.rhosts*. Jeśli spróbuje się użyć zdalnego urządzenia z poziomu niezaufanego komputera, będzie można zobaczyć następujący przerażający komunikat, informujący o braku uprawnień:

```
Permission Denied
```

Aby sprawdzić, czy dysponuje się zaufanym hostem, należy jako superużytkownik spróbować wykonać poniższe polecenie:

```
# rsh zdalny_system uname -a
```

Jeśli polecenie nie da oczekiwanych rezultatów, wiersz z nazwą lokalnego komputera trzeba umieścić w pliku */.rhosts* zdalnego hosta.

Niestety, w przypadku obecnych mieszanych środowisk nie zawsze wiadomo, jaka jest znana innym komputerom nazwa konkretnego hosta. Zdalny komputer może używać serwera DNS, NIS lub lokalnego pliku *hosts*. Gdy za pomocą programu *rsh* nawiąże się połączenie z hostem, początkowo będzie on widział adres IP komputera, który zainicjował połączenie. W dalszej kolejności zdalny host użyje funkcji `gethostbyaddr()` i spróbuje zamienić adres IP na nazwę. Zależnie od konfiguracji konkretnego komputera host może skomunikować się z serwerem DNS, NIS lub skorzystać z lokalnego pliku */etc/hosts*. Kolejność sprawdzania wymienionych źródeł również zmienia się w zależności od przeprowadzonej konfiguracji. Jeżeli do translacji adresu zostanie użyty lokalny plik *hosts* lub serwer NIS, adres może mieć lub nie postać w pełni kwalifikowanej nazwy domenowej, takiej jak *apollo.domena.com*. Jeśli zostanie zastosowany serwer DNS, nazwa komputera będzie w pełni kwalifikowaną nazwą domenową. Ważne jest, aby o tym wiedzieć, ponieważ taka nazwa musi zostać umieszczona w pliku */.rhosts*. Załóżmy, że lokalny komputer nosi nazwę *apollo*, a zdalny — *elvis*. Jeśli za pomocą narzędzia *rsh* z komputera *apollo* zamierza się nawiązać połączenie z hostem *elvis*, powinno się najpierw spróbować wykonać prosty krok. Na komputerze *elvis* należy wykonać następujące polecenie:

```
$ echo apollo >>/.rhosts
```

Jeśli polecenie nie zadziała, będzie to prawdopodobnie oznaczać, że host *apollo* jest dla komputera *elvis* czymś innym (na przykładem hostem *apollo.domena.com*). Aby się upewnić, że tak jest, za pomocą programu *telnet* z poziomu komputera *apollo* należy połączyć się z hostem *elvis*, a następnie użyć takich poleceń, jak `last`, `who`, `tty` i `netstat`, w celu sprawdzenia pola zawierającego nazwę zdalnego hosta. Jeśli wartością pola okaże się nazwa *apollo.domena.com*, należy umieścić ją w pliku */.rhosts* na komputerze *elvis* (przykładowo po stronie jednego klienta nazwa miała postać *apollo.DOMENA.COM*). Po wstawieniu poprawnej nazwy do pliku */.rhosts* narzędzie *rsh* powinno działać.

Wyświetlanie systemów plików, które muszą zostać zarchiwizowane (opcje W i w)

Opcje `W` i `w` są dostępne w większości systemów uniksowych i wyświetlają informacje na temat systemów plików, które muszą zostać zarchiwizowane. Opcja `w` zwykle prezentuje dane dotyczące wszystkich systemów plików, natomiast opcja `W` wyszczególnia jedynie te systemy

plików, dla których trzeba utworzyć kopię zapasową, na podstawie wybranego poziomu archiwizacji. Ponieważ opcje w przypadku różnych odmian systemu Unix nieznacznie się różnią, należy zapoznać się z odpowiednią dokumentacją.

Interesujące opcje narzędzia ufsdump systemu Solaris

Program *ufsdump* systemu Solaris oferuje kilka opcji niespotykanych w innych wersjach systemu Unix. Narzędzie obsługuje opcje: `l` (`autoloader`), `o` (`offline`), `a` (`archive file`) i `v` (`verify`). Oto opis opcji:

`l`
: Opcja automatycznego zmieniania powoduje wysunięcie taśmy, gdy przed zakończeniem działania narzędzia *dump* zostanie osiągnięty fizyczny koniec taśmy. W takiej sytuacji program *dump* czeka maksymalnie dwie minuty na włożenie następnej taśmy. Opcja działa dobrze z sekwencyjnymi automatycznymi zmieniarkami.

`o`
: Opcja trybu offline po prostu powoduje wysunięcie taśmy na zakończenie archiwizacji w celu ochrony taśmy przed nadpisaniem przez inny proces.

`a`
: Opcja pliku archiwum powoduje zapisanie tabeli zawartości narzędzia *dump* w pliku *plik_archiwum* (jest zapisywany na woluminie przez wszystkie polecenia `dump`). Plik może być użyty przez narzędzie *ufsrestore* do sprawdzenia, czy plik znajduje się na określonym woluminie (bez konieczności podłączania jego nośnika).

`v`
: Opcja weryfikacji porównuje kopię zapasową z rzeczywistym systemem plików. Choć może to dobrze wyglądać w teorii, operacja wymaga odłączenia systemu plików, co w przypadku wielu zastosowań nie jest zbyt praktyczne.

Wygląd kopii zapasowej narzędzia dump

W tym punkcie objaśniono podstawową różnicę między narzędziem *dump* a jego krewnymi, czyli programami *tar* i *cpio*. *dump* zapisuje tabelę zawartości na początku każdego woluminu, natomiast programy *tar* i *cpio* — nie.

Narzędzie dump zapisuje na woluminie indeks

Indeks jest wczytywany podczas interaktywnego procesu odtwarzania, co umożliwia przetwarzanie tabeli zawartości za pomocą takich poleceń, jak `cd` i `ls`, oraz przeglądanie i zaznaczanie plików, które mają być przywrócone (narzędzie *restore* omówiono w dalszej części rozdziału). Funkcja interaktywnego odtwarzania jest jedną z największych zalet narzędzia *restore* (w porównaniu z programami *tar* i *cpio*). Warto zwrócić uwagę na jedną ważną kwestię dotyczącą indeksu: jest tworzony na początku archiwizacji, zanim faktycznie zostanie podjęta próba zapisania czegokolwiek w kopii zapasowej. Obecność indeksu sprawia, że proces interaktywnego odtwarzania jest efektywny, ponieważ nie trzeba wczytywać całego woluminu, aby sprawdzić, co na nim jest. Jednak to, że indeks jest generowany przed zapisaniem zarchiwizowanych danych i być może minuty lub godziny przed umieszczeniem danych na taśmie, powoduje, że w kopii zapasowej nie są uwzględniane pliki utworzone podczas archiwizacji. Ponadto pliki usunięte w czasie tworzenia kopii zapasowej wprawdzie są wyszczególnione w indeksie, lecz w rzeczywistości nie ma ich na woluminie.

Użycie indeksu do utworzenia tabeli zawartości

Można utworzyć tabelę zawartości woluminu narzędzia *dump* przez fizyczne odczytanie zawartości indeksu utworzonego przez program i sprawdzenie, co zamierzał zapisać na woluminie. Warte uwagi jest również to, że odczytanie woluminu nie gwarantuje spójności pliku znajdującego się na woluminie w większym stopniu niż polecenie `ls -l` sprawdzające integralność pliku zawartego w katalogu. Można się zastanawiać, dlaczego mowa o tym w podrozdziale poświęconym narzędziu *dump*. Wynika to stąd, że tworzenie tabeli zawartości powinno być częścią każdej operacji archiwizowania realizowanej za pomocą narzędzia *dump*. Jak w takim razie wygenerować tabelę zawartości pliku programu *dump*? Przede wszystkim co tak naprawdę należy rozumieć przez plik programu *dump*? Być może pomocny będzie rysunek 3.4.

Rysunek 3.4. Format taśmy z danymi zapisanymi przy użyciu narzędzia dump

Wolumin tworzony przez narzędzie *dump* może mieć wiele plików programu *dump*, czasami nazywanych *partycjami*. Każdy plik jest zakończony znacznikiem końca pliku EOF (*End-Of-File*), symbolizowanym na rysunku 3.4 przez ciemniejsze pola.

Jeśli zamierza się uzyskać tabelę zawartości pliku 3 programu *dump* (rysunek 3.4), ma się do wyboru dwie opcje:

- Przy użyciu opcji `s` można nakazać programowi *restore* odczytanie z taśmy trzeciego pliku. W efekcie narzędzie pominie pliki pierwszy i drugi, a następnie odczyta trzeci plik (wariant ten nie dotyczy kopii zapasowych zapisywanych przez narzędzie *dump* na dysku).
- Można ręcznie wykonać pozycjonowanie taśmy (za pomocą narzędzia `mt` lub `tpctl`), tak aby zlokalizować początek pliku, a następnie nakazać programowi *restore* odczytanie go, tak jakby był pierwszym plikiem na taśmie.

Trzeba znać współczynnik bloków, przy użyciu którego zapisano wolumin. Jeśli nie ma się pewności, należy spróbować zastosować domyślną wartość współczynnika. Jeżeli to nie pomoże, należy zapoznać się z zawartością podrozdziału „Jak odczytać ten wolumin?" z rozdziału 23.

Pierwsza metoda jest najprostsza, ponieważ uwzględnia tylko jeden krok. Składnia polecenia jest następująca:

```
$ restore tsbfy plik współczynnik_bloków urządzenie
```

W celu odczytania z taśmy trzeciego pliku narzędzia *dump*, gdy współczynnik bloków ma wartość 32, należy posłużyć się poniższym poleceniem:

```
$ restore tsbfy 3 32 /dev/rmt/0cbn
```

Oto lista zastosowanych opcji i ich przeznaczenie:

- Opcja `t` nakazuje programowi *restore* odczytanie indeksu woluminu i przekazanie tabeli zawartości.
- Opcja `s` i powiązany z nią argument `3` powodują, że narzędzie *restore* odczytuje z taśmy trzeci plik narzędzia *dump*.

- Opcja `b`, mająca argument `32`, informuje program *restore*, że podczas zapisywania pliku narzędzia *dump* użyto współczynnika bloków o wartości 32.
- Opcja `f` i jej argument `dev` identyfikują lokalizację pliku narzędzia *dump* zapisanego na urządzeniu.
- Opcja `y` nakazuje programowi *restore* kontynuowanie działania w przypadku wystąpienia błędów zamiast pytania operatora, czy tego chce.

Jeżeli wybierze się drugą metodę, polegającą na ręcznym użytkowaniu taśmy, trzeba znać uniksową wersję polecenia obsługującego taśmę magnetyczną. Zwykle nosi ono nazwę `mt`. Polecenie ma pięć opcji: `status`, `rewind`, `offline`, `fsf` i `fsr`. Spośród nich cztery mogą być wykorzystywane podczas obsługi taśm zapisanych za pomocą narzędzia *dump*. Format polecenia jest następujący:

```
$ mt -t urządzenie argument
```

Jeśli planuje się pozycjonować taśmę, trzeba się upewnić, że korzysta się z urządzenia *bez przewijania*, takiego jak */dev/rmt/0n*. W przeciwnym razie urządzenie zostanie przewinięte od razu po zakończeniu pozycjonowania!

Niektóre wersje programu *mt* zamiast opcji `-t` używają opcji `-f`. Argument *`urządzenie`* identyfikuje stosowany napęd taśmowy bez przewijania (na przykład */dev/rmt/0n*). Pora w miejsce słowa *`argument`* wstawić jedną z następujących opcji:

`status`
: Opcja pozwala uzyskać status napędu taśmowego zwrócony przez wywołanie systemowe `ioctl`. Opcja nie wymaga dołączenia argumentu.

`rewind`
: Opcja powoduje przewinięcie taśmy do początku. W przypadku niektórych wersji systemu Unix opcja nosi nazwę `rew`. Opcja nie wymaga dodania argumentu.

`offline`
: Opcja powoduje wysunięcie taśmy z napędu. W przypadku niektórych wersji systemu Unix opcja ma nazwę `offl`. Opcja nie wymaga podania argumentu.

`fsf` *`x`*
: `fsf` jest skrótem od *forward space file*. Opcja pozycjonuje taśmę do momentu osiągnięcia *x* znaczników plików, gdzie *x* jest wartością większą od 0 (jeśli nie poda się dla *x* wartości, domyślnie będzie to 1). Jeżeli osiągnięto początek taśmy, oznacza to zlokalizowanie pliku 1. Aby przejść do pliku 3, trzeba przewinąć taśmę do przodu o dwa pliki. Wymaga to użycia opcji `fsf 2` wstawionej do polecenia `mt -t` *`urządzenie`* `fsf 2`.

`fsr` *`x`*
: `fsr` jest skrótem od *forward space record*. Opcja nie jest wymagana podczas obsługi taśm zapisanych za pomocą narzędzia *dump* (jeśli nie określi się wartości dla *x*, zostanie użyta domyślna, wynosząca 1).

Poniżej podano przykłady zastosowania polecenia `mt`. W celu przewinięcia napędu taśmowego */dev/rmt/0cbn* należy wykonać polecenie:

```
# mt -t /dev/rmt/0cbn rewind
```

Aby szybko przewinąć taśmę urządzenia */dev/rmt/0cbn* do drugiego pliku, należy zastosować polecenie:

```
# mt -t /dev/rmt/0cbn fsf 1
```

W celu wysunięcia taśmy z napędu */dev/rmt/0cbn* należy wykonać polecenie:

```
# mt -t /dev/rmt/0cbn offline
```

Aby uzyskać status taśmy napędu */dev/rmt/0cbn*, należy użyć polecenia:

```
# mt -t /dev/rmt/0cbn status
```

Po zlokalizowaniu na taśmie właściwego pliku wystarczy użyć tego samego polecenia `restore` co wcześniej, pomijając opcję `s` i jej argument.

```
$ restore tbfy 32 /dev/rmt/0cbn
```

Niezależnie od zastosowanej metody tabela zawartości jest przesyłana do standardowego wyjścia, które powinno się przekierować do pliku. W przypadku tego wyjścia godne uwagi jest to, że nie ma w nim nazwy systemu plików zarchiwizowanego na woluminie. Tabela zawartości jest względna w stosunku do tego systemu plików, niezależnie od jego nazwy. Jeśli na przykład zarchiwizowano katalog */var* i szukano by pliku */var/adm/messages*, uzyskany rezultat wyglądałby podobnie do poniższego:

```
345353  ./adm/messages
```

Namawiam do generowania tabeli zawartości dla każdego woluminu w momencie tworzenia go za pomocą narzędzia *dump* i zapisywania wyniku w pliku o nazwie zgodnej z nazwą woluminu. Oczywiście należy użyć unikatowej nazwy, takiej jak poniższa:

```
./dump.system.system_plików.poziom0.19Maj.2006
```

Zapisywanie tabel zawartości w ten sposób bardzo się przydaje, gdy szuka się pliku i początkowo nie można go znaleźć na żadnym woluminie. Szybkie przeszukanie za pomocą narzędzia *grep* wszystkich plików programu *dump* pozwoli stwierdzić, który wolumin jest potrzebny.

Odtwarzanie danych za pomocą narzędzia restore

Gdy pisałem ten podrozdział, do głowy przyszło mi zdanie pochodzące z reklamy leku na choroby lokomocyjne, który w Stanach Zjednoczonych nazywa się Dramamine. Brzmiało ono tak: „*Pora* na zażycie Dramaminy jest za *późna* na to, aby to zrobić" (a zatem mowa o chwili, gdy jest się już chorym). To samo dotyczy nauki korzystania z narzędzia *restore*. Trzeba bardzo dobrze zapoznać się z różnymi metodami odtwarzania danych (z kopii zapasowej utworzonej przy użyciu narzędzia *dump*) za pomocą programu *restore*. Jeśli Czytelnik przeprowadza ważną operację odtwarzania akurat w czasie, gdy czyta ten rozdział, nie ma powodu do obaw. Podrozdział zorganizowano z myślą o takim scenariuszu i w związku z tym uwzględnia każdy szczegół dotyczący narzędzia *restore*.

W poniższym punkcie przyjęto, że wiadomo na pewno, iż wolumin zapisano za pomocą programu *dump*, i znany jest rozmiar bloku woluminu. Jeśli nie dysponuje się takimi informacjami, należy zapoznać się z zawartością podrozdziału „Jak odczytać ten wolumin?" z rozdziału 23.

Niewystarczające, aby było przydatne

Pewnego razu zlecono mi odtworzenie danych komputera wycofanego 10 lat wcześniej. Po otrzymaniu nazwy komputera i orientacyjnej lokalizacji taśm rozpocząłem poszukiwania.

Gdy znalazłem taśmy i pożyczyłem napęd taśmowy o wystarczająco niewielkiej gęstości, aby można było je odczytać, stwierdziłem, że zostały utworzone za pomocą narzędzia *dump* w formacie już nieobsługiwanym! Znalazłem kod źródłowy oryginalnego programu *restore* (w tym przypadku wchodzącego w skład systemu BSD 4.1), pobrałem go na komputer (z systemem SunOS 4.0.1, podobnym do systemu BSD 4.3) i zacząłem prace nad przeniesieniem starego oprogramowania. Bez powodzenia. Wkrótce zdałem sobie sprawę z tego, że zadanie to zajęłoby mi całe tygodnie, ponieważ formaty systemu plików i narzędzia *dump* zmieniły się tak znacząco.

Ponieważ *istniało* inne rozwiązanie, przeszukałem magazyny w celu znalezienia dodatkowych taśm. Szczęśliwie udało mi się odszukać następny stos taśm, oznaczonych jako przechowujące kopie zapasowe utworzone za pomocą programu *tar*. Udało mi się! Większość z tych taśm nadal można było odczytać. Dane uzyskałem już przy pierwszym podejściu.

Morał z tej historii jest taki, że gdy wycofuje się komputer, należy sporządzić archiwalną kopię zapasową danych w każdym możliwym formacie i zapisać ją na wszystkich dostępnych typach nośników. Część narzędzi, jak *dump*, jest bardzo skuteczna, lecz pewnego dnia mogą nie być już obsługiwane. Inne narzędzia, jak *tar* i *cpio*, były wykorzystywane przez lata. Ponieważ zmieniają się czasy, nośniki i formaty, należy wykonać tak wiele odmian kopii zapasowych, jak to możliwe, aby dane można było odtworzyć przez jak najdłuższy okres.

Powyższa historia sprawiła, że stałem się wielkim zwolennikiem używania narzędzia *tar* do sporządzania archiwalnych kopii zapasowych. W końcu słowo *tar* jest skrótem od *Tape ARchiver* (archiwizator danych na taśmie).

— Doug Freyburger

Czy można odczytać wolumin z kopią zapasową?

Aby mieć pewność, że znany jest format kopii zapasowej zapisanej na taśmie i rozmiar bloku, należy spróbować wyświetlić tabelę zawartości. Poniższe polecenie generuje tabelę zawartości woluminu utworzonego przy użyciu narzędzia *dump*.

```
$ restore tbfy rozmiar_bloku nazwa_urządzenia
```

Przykładowo, aby odczytać tabelę zawartości taśmy zapisanej za pomocą programu *dump* (z zastosowaniem współczynnika bloków o wartości 32) i umieszczonej w urządzeniu */dev/rmt/0cbn*, należy wykonać następujące polecenie:

```
$ restore tbfy 32 /dev/rmt/0cbn
```

Jeżeli polecenie zadziała, reszta będzie już prosta (w przeciwnym razie należy zapoznać się z zawartością podrozdziału „Jak odczytać ten wolumin?" z rozdziału 23).

Współczynnik bloków

Czasami narzędzie *dump* może zapisać kopię zapasową przy użyciu współczynnika bloków, z którym nie może sobie poradzić program *restore*. Zwykle tego typu problem można bardzo łatwo rozwiązać. Również w tym przypadku konieczna jest znajomość rozmiaru bloku, przy

użyciu którego zapisano wolumin. Zgodnie z tym, co napisano w rozdziale 23., należy określić rozmiar bloku woluminu. Załóżmy, że wielkość bloku woluminu wynosi 65 536. Przy użyciu narzędzia *dd* należy odczytać wolumin i jego wynik przekazać programowi *restore*, podając - jako argument plikowy. Poniższe polecenie nakaże narzędziu *restore* wczytanie swoich danych ze standardowego wejścia.

```
# dd if=nazwa_urządzenia bs=64k|restore tfy -
```

Dlaczego to działa? Zapisywanie danych na woluminie w postaci bloków w rzeczywistości zmienia sposób fizycznego rozmieszczenia danych na woluminie. Dla narzędzia *restore* musi być zrozumiały format bloków, aby mogło odczytać zawartość woluminu. Jeśli jednak dane z woluminu odczyta się za pomocą programu *dd*, zostaną umieszczone w potoku. Polecenie `dd` jako rozmiar bloku w potoku ustawia wartość 1, pozwalając programowi *restore* użyć podczas odczytu bloku danych dowolnego rozmiaru bloku.

Różnice w uporządkowaniu bajtów

Format kopii zapasowej tworzonej przez narzędzie *dump* jest bardzo ściśle powiązany z konkretnym systemem plików. Jeżeli występują różnice w kolejności bajtów, prawdopodobnie różne są też wersje programów *dump* i *restore*. Najprostsze i chyba jedyne, co należy zrobić, jest poszukanie komputera z identycznym systemem operacyjnym jak ten użyty do zapisania woluminu archiwizacyjnego. Wynika to stąd, że odwrotne uporządkowanie bajtów może umożliwić odczytanie nagłówka kopii zapasowej, lecz odtworzone pliki, zależnie od formatu stosowanego przez program *dump*, mogą okazać się bezużyteczne.

Różne wersje narzędzia dump

Niestety, z upływem czasu problem różnych wersji narzędzia *dump* tylko się pogłębia. W przeciwieństwie do innych narzędzi omówionych w rozdziale program *dump* jest mocno powiązany z systemem plików i zwykle współpracuje tylko z jednym typem systemu plików. Problem polega na tym, że twórcy systemów uniksowych cały czas próbują ulepszać system plików. W efekcie wielu dostawców systemu Unix oferuje więcej niż jeden typ systemu plików. Jeśli narzędzie *dump* w ogóle występuje w wykorzystywanej wersji systemu Unix, może obsługiwać systemy plików tylko starszego typu. Niekiedy istnieje wiele wersji programu *dump*. Przykładowo system IRIX ma zarówno narzędzie *dump*, jak i *xfsdump*. Każda wersja programu *dump* dysponuje też własną wersją narzędzia *restore*. Różne wersje programu *restore* mogą być w stanie odczytać kopię zapasową utworzoną przy użyciu innej wersji narzędzia *dump* lub nie. Jest to obszar, w przypadku którego zdecydowanie ma zastosowanie stwierdzenie, że „zakres może się zmieniać".

Prawdopodobnie najlepszym przykładem zmiennej natury narzędzia *dump* jest system plików XFS systemu SGI i jego narzędzie *xfsdump*. Początkowo można odnieść wrażenie, że jest to stare polecenie `efsdump` z kilkoma nowymi opcjami. Jednak jest to bardzo dalekie od prawdy. Załóżmy, że korzysta się z utworzonego we własnym zakresie programu, który używa narzędzia *dump*. Dodatkowo przyjmijmy, że do listy dołączeń programu *xfsdump* dodano właśnie zainstalowany nowy system plików XFS. W pierwszej kolejności narzędzie *xfsdump przewija taśmę*, niezależnie od tego, czy wybrano urządzenie z przewijaniem, czy nie. Następnie narzędzie próbuje odczytać z taśmy pierwszy blok danych. Zależnie od złożoności skryptu wywołującego program *xfsdump* pierwszym plikiem na taśmie może być elektroniczna etykieta zapisana przez

skrypt lub pierwsza kopia zapasowa sporządzona przez narzędzie *dump*. Gdy będzie to pierwsza kopia, narzędzie *xfsdump* poinformuje, że nie znalazło swojej kopii zapasowej i nadpisze dane. Jeśli program *xfsdump* napotka na własną kopię zapasową, nie nadpisze jej, lecz dołączy do niej dane.

Ponadto narzędzie *xfsdump* — i to jest być może „najciekawsze" — w ramach tworzenia jednej kopii zapasowej zapisuje na taśmie wiele plików. Zwykle każda kopia zapasowa narzędzia *dump* ma postać jednego pliku zapisanego na taśmie. Z kolei program *xfsdump* używa algorytmu określającego liczbę plików, które powinny zostać umieszczone na taśmie. Choć przypuszczalnie dzięki temu proces odtwarzania jest szybszy, dodatkowo staje się całkowicie zgodny z niemal wszystkimi możliwymi skryptami powłoki.

Najlepszym rozwiązaniem jest bycie przygotowanym. Trzeba wiedzieć, jakich używa się wersji narzędzi *dump* i *restore*, a także poeksperymentować z nimi, aby stwierdzić, czy są w stanie odczytać różne woluminy. Jeśli chodzi o dwie wersje narzędzia *dump* tego samego systemu, prawdopodobnie będą ze sobą współpracować zawsze lub nigdy. Trzeba pamiętać, aby testować, testować i jeszcze raz testować.

Składnia polecenia restore

Gdy można odczytać wolumin z kopią zapasową narzędzia *dump*, trzeba zdecydować, jakie dane muszą zostać odczytane i w jaki sposób. W niniejszym punkcie omówiono powszechnie używane argumenty polecenia `restore` i wyjaśniono, kiedy je stosować.

Zasadniczo należy wyróżnić cztery operacje, które można wykonać w przypadku woluminu zapisanego za pomocą narzędzia *dump*. Oto one:

- odczytanie tabeli zawartości w celu jej przejrzenia,
- odtworzenie całego systemu plików,
- odtworzenie wybranych plików,
- przeprowadzenie interaktywnego procesu odtwarzania.

W przypadku trzech pierwszych zastosowań narzędzie *restore* może pobrać dane ze standardowego wejścia. Są to właściwe warianty użycia polecenia `restore`, gdy trzeba do niego w potoku przekazać dane, tak jak we wcześniej przedstawionym przykładzie polecenia `dd`. Interaktywny proces odtwarzania sprawdza się dobrze tylko wtedy, gdy ma dostęp do całego pliku kopii zapasowej narzędzia *dump* lub taśmy. Składnia standardowego polecenia `restore` jest następująca:

```
$ restore [trxi]vbsfy współczynnik_bloków numer_pliku nazwa_urządzenia
```

Opcje polecenia restore

Sposób działania narzędzia *restore* zależy od typu przekazanych mu argumentów.

Określanie typu operacji odtwarzania

Pierwszy argument polecenia `restore` identyfikuje *typ* operacji odtwarzania, która zostanie wykonana. Można określić tylko jeden z czterech następujących argumentów:

- `t` — nakazuje narzędziu *restore* wyświetlenie tabeli zawartości woluminu;
- `r` — powoduje, że cała zawartość woluminu powinna zostać odtworzona do bieżącego katalogu roboczego;
- `x` — nakazuje narzędziu *restore* wyodrębnienie z kopii zapasowej tylko plików wyszczególnionych na końcu polecenia;
- `i` — umożliwia przeprowadzenie interaktywnego procesu odtwarzania.

Określanie sposobu przebiegu procesu odtwarzania

Reszta argumentów jest opcjonalna i decyduje o sposobie działania narzędzia *restore* podczas procesu odtwarzania.

- `v` — włącza wyświetlanie szczegółowych wyników;
- `s` — nakazuje programowi *restore* przed rozpoczęciem odczytu taśmy pominięcie kilku plików;
- `b` — umożliwia podanie współczynnika bloków dla odczytywanego woluminu;
- `f` — określa ścieżkę pliku urządzenia używanego napędu archiwizującego (lub pliku na dysku);
- `y` — nakazuje programowi *restore*, aby spróbował kontynuować działanie mimo wystąpienia błędów odczytu.

W następnych podpunktach bardziej szczegółowo omówiono powyższe opcje.

Tworzenie tabeli zawartości woluminu narzędzia dump (t)

Opcja `t` służy do sprawdzenia, jakie pliki znajdują się na woluminie utworzonym za pomocą narzędzia *dump*. Jest to opcja dobrze nadająca się do tego, aby uwzględnić ją w każdym zautomatyzowanym skrypcie powłoki kontrolującym kopie zapasowe sporządzone przy użyciu narzędzia *dump*. Opcja jest też przydatna, gdy nie jest się pewnym takich rzeczy, jak wielkość znaków w nazwach plików lub ich lokalizacja. Można umieścić w pliku listę plików wchodzących w skład dowolnego woluminu, a następnie za pomocą narzędzi takich jak *grep* znaleźć szukane pliki. Oto przykład:

```
# restore tfy urządzenie >/tmp/dump.list
```

Powyższe polecenie wczytuje z *urządzenia* tabelę zawartości kopii zapasowej narzędzia *dump* i wynik umieszcza w pliku */tmp/dump.list*. Poniższe polecenie szuka w pliku */tmp/dump.list* łańcucha *nazwa_pliku*.

```
# grep nazwa_pliku /tmp/dump.list
3455                    ./jakiś_katalog/nazwa_pliku
```

Przeprowadzanie procesu całkowitego odtworzenia (rekursywnego) systemu plików (r)

Opcję `r` zaprojektowano z myślą o odtwarzaniu całego systemu plików przez odczytanie pełnej zawartości woluminu zapisanego za pomocą programu *dump*. Opcja powinna być stosowana tylko wtedy, gdy ma się absolutną pewność, że zamierza się odtworzyć cały system plików. Proces wymaga, aby zacząć od pliku kopii zapasowej poziomu 0. Później opcjonalnie można wczytać wszystkie przyrostowe kopie zapasowe. Proces zapisuje plik *restoresymtable* (w niektórych wersjach systemu Unix nosi nazwę *restoresmtable*) i odwołuje się do niego podczas

wczytywania odtwarzanych danych przyrostowych kopii zapasowych. Przyrostowa kopia zapasowa narzędzia *dump* rejestruje czas kopii niższego poziomu, na której bazuje. Ponieważ opcję `r` dodano w celu odtwarzania całego systemu plików, nie pozwala na odczytanie przyrostowej kopii zapasowej narzędzia *dump* bazującej na woluminie, którego jeszcze nie wczytano. Załóżmy, że są trzy kopie zapasowe programu *dump*: poziomu 0 z poniedziałku, poziomu 1 z wtorku i poziomu 2 ze środy. Jeśli za pomocą opcji `r` wczyta się kopię poziomu 0, a następnie spróbuje odczytać kopię poziomu 2 bez uprzedniego wczytania kopii poziomu 1, narzędzie *restore* zaprotestuje.

Po ukończeniu odtwarzania całego systemu plików plik *restoresymtable* powinno się usunąć (jednak nie należy tego robić do momentu wczytania taśm z kopiami zapasowymi wszystkich poziomów).

W celu użycia opcji `r` najpierw należy za pomocą polecenia `cd` przejść do systemu plików, który zamierza się odtworzyć, a następnie wczytać kopię zapasową poziomu 0 i wykonać następujące polecenie:

```
# restore rbvsfy współczynnik_bloków numer_pliku nazwa_urządzenia
```

Przykładowo, aby odtworzyć całą zawartość taśmy z kopiami zapasowymi narzędzia *dump* zapisanymi z wykorzystaniem współczynnika bloków o wartości 32 na urządzeniu */dev/rmt/0cbn*, należy zastosować poniższe polecenie.

```
$ restore rvbfy 32 /dev/rmt/Ocbn
```

Gdy polecenie zakończy działanie, należy wczytać dowolne przyrostowe kopie zapasowe, począwszy od kopii najniższego poziomu, i ponownie wykonać to samo polecenie. Proces należy powtarzać do momentu załadowania najbardziej aktualnej przyrostowej kopii zapasowej. Jeżeli dysponuje się więcej niż jedną kopią zapasową narzędzia *dump* tego samego poziomu, trzeba załadować tylko najnowszą. Jeśli na przykład raz w miesiącu tworzy się kopię zapasową poziomu 0, a w pozostałe dni miesiąca sporządza się kopie zapasowe poziomu 1, w celu odtworzenia całego systemu plików trzeba będzie załadować jedynie kopię poziomu 0, a następnie najnowszą kopię zapasową poziomu 1.

Odtwarzanie plików według nazwy (x)

Z opcji `x` można skorzystać, gdy znana jest dokładna nazwa i ścieżka pliku lub plików, które zamierza się odtworzyć (ponieważ nie wszystkie testowane przeze mnie wersje narzędzia *restore* obsługują w obrębie listy dołączeń symbole wieloznaczne, trzeba znać *dokładną* nazwę plików). Opcja powoduje, że program *restore* działa tak jak narzędzie *tar*, umożliwiając podanie w wierszu poleceń plików, które mają być wczytane z kopii zapasowej. Pamiętając o tym, że wszystkie kopie zapasowe są tworzone przez narzędzie *dump* z zastosowaniem względnych ścieżek, za pomocą polecenia `cd` trzeba przejść do systemu plików, w którym mają się znaleźć odtwarzane pliki. W dalszej kolejności należy wykonać poniższe polecenie, wyodrębniające pliki z kopii zapasowej.

```
# restore xbvsfy współczynnik_bloków numer_pliku nazwa_urządzenia ./katalog/plik1
    ./katalog/plik2
```

Przykładowo, aby odtworzyć pliki */etc/hosts* i */etc/passwd* z taśmy zapisanej za pomocą narzędzia *dump* przy wykorzystaniu współczynnika bloków o wartości 32 i urządzenia */dev/rmt/0cbn*, należy wykonać następujące polecenie:

```
$ restore xvbfy 32 /dev/rmt/Ocbn ./etc/hosts ./etc/passwd
```

Interaktywne odtwarzanie plików (i)

Opcja `i` odróżnia narzędzie *restore* od programów *tar* i *cpio*. Gdy program *dump* tworzy kopię zapasową, na jej początku zapisuje indeks z tym, co zostanie zarchiwizowane (podobnie jak w przypadku innych trybów działania narzędzia *restore*, przed jego uruchomieniem za pomocą polecenia `cd` powinno się przejść do systemu plików, w którym mają znaleźć się odtworzone pliki). Opcja `i` symuluje podłączenie woluminu narzędzia *dump* i inicjuje fikcyjną powłokę umożliwiającą wykonanie takich poleceń, jak `cd`, `ls`, `pwd`, `add`, `delete` i `extract`. Przy użyciu tych poleceń można przemieszczać się między katalogami woluminu narzędzia *dump* bardzo podobnie jak w przypadku katalogów systemu plików.

Po natrafieniu na plik, który ma być uwzględniony w procesie odtwarzania, wystarczy wykonać polecenie `add` *`nazwa_pliku`*. Ponieważ większość wersji narzędzia *restore* obsługuje również symbole wieloznaczne powłoki, można także zastosować polecenie `add` *`*wzorzec*`*. Po wybraniu pliku do odtworzenia, gdy ponownie wykonując polecenie `ls`, zażąda się listy plików, obok nazwy pliku pojawi się znak gwiazdki. Jeśli zauważy się, że dodano plik, który nie ma być odtworzony, wystarczy wykonać polecenie `delete` *`nazwa_pliku`* lub `delete` *`*wzorzec*`*. Oczywiście nie spowoduje to usunięcia pliku z woluminu, a jedynie z listy plików przeznaczonych do wczytania z kopii zapasowej. Po określeniu plików, które mają zostać odtworzone, wystarczy wykonać polecenie `extract`.

Narzędzie *restore* zadaje następnie pytanie dotyczące woluminu, od którego ma zacząć. Pytanie jest istotne tylko wtedy, gdy odtwarza się kilka plików rozmieszczonych na wielu taśmach. Ponieważ pliki są zapisywane zgodnie z kolejnością i-węzłów, można najpierw włożyć do napędu ostatnią taśmę i narzędzie *restore* będzie w stanie odczytać numer i-węzła pierwszego pliku, a następnie od razu stwierdzić, czy musi odczytać cokolwiek z tej taśmy. Jeśli tak, narzędzie musi jedynie odczytać dane aż do ostatniego i-węzła na taśmie. Jeżeli narzędzie *restore* nadal musi odczytać pliki z innych taśm, należy je umieszczać w napędzie numerami w odwróconej kolejności. Również w tym przypadku narzędzie wie, czy musi odczytać taśmy i w jakim zakresie. Jeśli najpierw włoży się do napędu taśmę 1, urządzenie po prostu odczyta kolejne taśmy. Przy odtwarzaniu systemu plików działa to po prostu świetnie.

Jeśli kilka plików odtwarza się z kopii zapasowej utworzonej za pomocą narzędzia *dump* na wielu taśmach, należy je umieszczać w napędzie w odwrotnej kolejności i wprowadzać odpowiednie numery taśm. Jeżeli jest tylko jedna taśma lub po prostu zamierza się kolejno odczytać taśmy, wystarczy wprowadzić wartość 1.

Wybrane pliki są następnie odtwarzane w katalogu uaktywnionym podczas wykonywania polecenia `restore` (tworzy ono wszystkie katalogi wymagane do zapisania odtworzonych plików). Po ukończeniu odtwarzania narzędzie zadaje pytanie `set owner/mode for '.'?`. Wiele osób nie rozumie jego znaczenia. Załóżmy, że zarchiwizowano katalog */home/jnowak* należący do użytkownika *jnowak*. Jeśli podczas odtwarzania zawartości tego katalogu do katalogu */tmp* na pytanie odpowie się twierdząco, właścicielem katalogu */tmp* stanie się użytkownik *jnowak*! A zatem trzeba zachować ostrożność podczas odtwarzania plików do alternatywnych lokalizacji i odpowiadania na powyższe pytanie. Negatywna odpowiedź powoduje pozostawienie uprawnień katalogu bez zmian.

Przykład 3.1 prezentuje prostą sesję narzędzia *restore*. Większość widocznych poniżej dodatkowych komentarzy, takich jak rozmiar bloku, data utworzenia woluminu przez program *dump* i inne komunikaty, jest wynikiem użycia opcji `v` (omówiono ją w dalszej części rozdziału).

W ramach sesji plik */etc/passwd* jest zaznaczany i odtwarzany do katalogu */tmp/etc* (właśnie z tego powodu przed rozpoczęciem procesu odtwarzania przeszedłem do katalogu */tmp*).

Przykład 3.1. Prosta sesja narzędzia restore

```
# cd /tmp
# ufsrestore ifvy /tmp/dump
Verify volume and initialize maps
Media block size is 126
Dump date:      Sun Apr 30 23:07:22 2006
Dumped from: Sun Apr 30 22:15:37 2006
Level 9 dump of / on apollo:/dev/dsk/c0t0d0s0
Label: none
Extract directories from tape
Initialize symbol table.
ufsrestore > ls
.:
    2  *./       2  *../      11395  devices/    28480  etc/

ufsrestore > cd etc
ufsrestore > ls
./etc:
 28480  ./      2  *../        28562  dumpdates   28486  passwd
ufsrestore > add passwd
Make node ./etc
ufsrestore > ls
./etc:
 28480  *./      2  *../        28562  dumpdates   28486  *passwd

ufsrestore > extract
Extract requested files
You have not read any volumes yet.
Unless you know which volume your file(s) are on you should start
with the last volume and work towards the first.
Specify next volume #: 1
extract file ./etc/passwd
Add links
Set directory mode, owner, and times.
set owner/mode for '.'? [yn] n
ufsrestore > q
# ls -lt /tmp/etc/passwd
-rw-r--r--   1 root     sys     34983 Apr 28 23:54 /tmp/etc/passwd
```

Odtwarzanie plików do innej lokalizacji

Wszystkie nazwy plików zawarte w woluminie archiwizacyjnym narzędzia *dump* mają względną ścieżkę. Inaczej mówiąc, jeśli archiwizuje się katalog */home* zawierający katalogi */home/kaczor* i */home/donald*, lista plików będzie wyglądać tak:

```
15643   ./kaczor
12456   ./donald
```

W związku z tym odtworzenie plików w innym miejscu jest bardzo łatwe. Wystarczy przejść do katalogów innych niż oryginalny (na przykład do katalogu */home1*) i rozpocząć proces odtwarzania. Narzędzie utworzy wymagane katalogi. Jeżeli w poprzednim przykładzie katalog */home* zmieni się na */tmp*, zostaną utworzone katalogi */tmp/kaczor* i */tmp/donald*.

Żądanie wyświetlania szczegółowych komunikatów (v)

Opcja `v` nie wymaga argumentu i powoduje pokazywanie szczegółowych informacji. Opcja generuje wiele dodatkowych informacji, takich jak data i poziom kopii zapasowej, a także nazwa każdego odtwarzanego pliku.

Opcje: `s`, `b` i `f` wymagają argumentu. Działają one dokładnie tak jak ich odpowiedniki z polecenia `dump` (nie oznacza to jednak, że opcja `s` w obu poleceniach pełni identyczną rolę). Wszystkie konieczne opcje wystarczy wstawić za poleceniem `restore`, a następnie podać argument powiązany z każdą opcją w kolejności zgodnej z ustawieniem opcji. Przykładowo, aby użyć opcji: `b`, `f` i `s`, należy wykonać następujące polecenie:

```
# restore tbfsy współczynnik_bloków plik_urządzenia numer_pliku
```

Pomijanie plików (s)

Opcja `s` służy do odczytu kopii zapasowej narzędzia *dump* innej niż pierwsza na taśmie. Gdy dla napędu taśmowego bez przewijania wykona się wiele poleceń `dump`, każde spowoduje utworzenie oddzielnego pliku. Pliki są od siebie oddzielone znakiem końca pliku. Za pomocą jednego polecenia nie można odczytać wszystkich takich plików (jeśli ktoś miałby przeprowadzać operację odtwarzania, raczej by z niej zrezygnował, ponieważ każdy plik prawdopodobnie byłby kopią zapasową oddzielnego systemu plików). Każdą kopię zapasową trzeba odczytać za pomocą niezależnego polecenia `restore`. Można w tym miejscu wyróżnić dwa scenariusze. Oto one:

- sukcesywne odczytywanie z taśmy każdego systemu plików, tak jak wtedy, gdy chce się uzyskać tabelę zawartości całej taśmy;
- wczytanie z taśmy wybranego systemu plików.

Wiele systemów plików można kolejno odczytać, po prostu wykonując kilka poleceń `restore` dla napędu taśmowego bez przewijania. To, czy coś takiego się uda, zależy od sposobu działania systemowego sterownika używanego napędu taśmowego. Po udanym wykonaniu polecenia `restore` taśma może zatrzymać się przy końcu pliku tuż za znakiem końca pliku. Jeśli urządzenie jest zgodne ze standardem Berkeley, może zatrzymać się przy końcu pliku tuż *przed* znakiem końca pliku. W tym przypadku następne polecenie `restore` nie zadziała. Czasami można sobie z tym poradzić przez wykonanie polecenia `mt -t urządzenie fsf 1`, które ustawia taśmę bezpośrednio za znakiem końca pliku. Można wtedy wykonać następne polecenie `restore`.

Operację odczytania kopii zapasowej narzędzia *dump* z konkretnym systemem plików można wykonać na dwa sposoby:

- Przewinięcie za pomocą polecenia `mt` lub `tctl` taśmy do odpowiedniego pliku kopii zapasowej, a następnie wykonanie polecenia `restore` bez argumentu `s`.
- Przewinięcie taśmy i poinformowanie narzędzia *restore* za pomocą opcji `s`, który plik ma wczytać. W dalszej kolejności taśma jest przewijana do tego pliku, który jest odczytywany. Opcja `s` wymaga argumentu o wartości z przedziału od `1` do *`n`*. W miejsce *`n`* należy wstawić numer pliku, który ma być odczytany z taśmy. Ponieważ pierwsza kopia zapasowa zapisana na taśmie ma numer 1, wykonanie polecenia `restore tsf 1 urządzenie` pod względem funkcjonalnym odpowiada poleceniu `restore tf urządzenie`.

Warto zwrócić uwagę na różnice między programami *mt* i *restore*. Oba narzędzia w odmienny sposób numerują pliki taśmy. Mówiąc dokładniej, numeracja jest przesunięta o 1. Jeśli program *mt* ma przewinąć taśmę do drugiego pliku, należy wykonać polecenie `mt -t urządzenie fsf 1`. Aby narzędzie *restore* odczytało z taśmy drugą kopię zapasową, należy zastosować polecenie `restore [irtx]s 2`. Tego typu różnica zrobiła zamieszanie niejednemu administratorowi!

Określanie współczynnika bloków (b)

Opcja `b` pozwala przekazać narzędziu *restore* wartość współczynnika bloków, którego użyto podczas zapisywania woluminu. Opcja wymaga argumentu będącego wartością liczbową, zwykle z przedziału od 1 do 126 (lub maksymalną wartością obsługiwaną przez używaną wersję programu *restore*). Współczynnik bloków jest mnożony przez minimalny rozmiar bloku obsługiwany przez używaną wersję narzędzia *restore*.

Choć minimalny rozmiar bloku zwykle wynosi 1024, może mieć wartość 512 (należy zajrzeć do dokumentacji dołączonej do wykorzystywanej wersji programu *restore*). Ponieważ obecnie wiele wersji programu *restore* może automatycznie wykrywać najbardziej typowe współczynniki bloków, może nawet nie być potrzeby używania opcji `b`. Jeśli stwierdzi się, że stosuje się współczynnik, który może nie zostać automatycznie zidentyfikowany przez używaną wersję programu *restore*, za pomocą opcji `b` należy poinformować go o wartości użytego współczynnika. Jeżeli do odczytu danych stosuje się polecenie `dd`, a następnie w ramach potoku przekazuje je narzędziu *restore*, nie trzeba używać opcji `b`.

Określanie napędu archiwizacyjnego lub pliku (f)

Opcja `f` jest używana dość często. Nakazuje programowi *restore* odczyt z urządzenia określonego w powiązanym argumencie zamiast z domyślnego napędu taśmowego używanej wersji systemu Unix. Argument może zawierać dowolny z następujących łańcuchów:

/dev/rmt/0
: Nazwa lokalnego urządzenia (na przykład */dev/rmt0*, */dev/rmt/8500compressed*)

/kopie_zapasowe/plik_kopii_dump
: Dowolny plik kopii zapasowej utworzony przez narzędzie *dump*.

zdalny_host:/dev/rmt/0
: Zdalne urządzenie, którego ścieżka rozpoczyna się nazwą hosta (nie wszystkie wersje programu *restore* obsługują zdalne hosty).

W celu uzyskania informacji na temat bezpieczniejszej metody korzystania ze zdalnych urządzeń trzeba zapoznać się z treścią ostatniego podrozdziału niniejszego rozdziału.

"-"
: Standardowe wejście używane na przykład podczas pobierania danych z programu *dd* lub wysyłania ich przez narzędzie *dump* do standardowego wyjścia.

Określanie nieprzerwanego działania podczas odtwarzania (y)

Normalnie gdy program *restore* napotka w pliku błąd, zatrzyma się i zapyta operatora, czy kontynuować działanie. Jeśli zastosuje się opcję `y`, narzędzie nie zada żadnego pytania i po wystąpieniu błędu spróbuje wykonać zadanie najlepiej, jak może.

Ograniczenia narzędzi dump i restore

Programy *dump* i *restore* oferują wiele możliwości. Dobry skrypt powłoki może zautomatyzować korzystanie z nich i zapewnić wysoki poziom bezpieczeństwa, gdy dysk się zepsuje. Jednak narzędzia te mają następujące ograniczenia:

- W przypadku narzędzia *dump* nie istnieje metoda uzyskania spójnego obrazu całego systemu plików w dowolnym wybranym momencie.
- Polecenie `dump` czasami nie informuje o otwartych plikach i innych problemach. Wyświetla komunikaty o błędzie, gdy wszystko naprawdę się skomplikuje.
- Choć pliki zostały pominięte, w rzeczywistości narzędzie *restore* może sprawić, że użytkownik będzie przekonany, iż znajdują się na woluminie.
- Trzeba pisać skrypty współpracujące z programem *dump*. Skrypty te mogą zawierać błędy.
- Istnieje wiele wersji narzędzia *dump*, z których nie wszystkie dobrze ze sobą współpracują.
- Podobnie do wszystkich innych wbudowanych narzędzi programy *dump* i *tar* są pozbawione indeksów dołączanych do komercyjnego oprogramowania (choć wersja programu *dump* dla systemu Solaris ma opcję wykonującą w pewnym stopniu indeksowanie, zdecydowanie nie jest tym, czego można oczekiwać po komercyjnym produkcie).

Dopóki pamięta się o powyższych ograniczeniach, można przez długi czas używać narzędzi *dump* i *restore* bez potrzeby wydawania dodatkowych pieniędzy na komercyjne oprogramowanie. Dobrej zabawy!

Funkcje godne sprawdzenia

Jeśli zamierza się napisać własny skrypt archiwizujący współpracujący z programem *dump* lub dowolnym innym poleceniem opisanym w niniejszym rozdziale, trzeba zadbać o to, aby wykonywał niżej wymienione zadania.

Kontrolowanie błędów w dużym zakresie
: Przez lata miałem do czynienia ze zbyt wieloma skryptami powłoki, które zakładały różne rzeczy. Nie należy przyjmować, że proste polecenie zadziałało tylko dlatego, że zawsze jest tak, jak się założyło. Gdy automatyzuje się różne rzeczy, *zawsze* należy sprawdzać zwrócony kod. Jeśli można przewidzieć, co powoduje określony błąd, należy spróbować utworzyć skrypt, który wyeliminuje go.

Powiadamianie, powiadamianie i jeszcze raz powiadamianie
: Nie sposób napisać wystarczająco dobitnie, jakie to ważne. Jeżeli skrypt stwierdzi coś, z czym się nie spotkał, powinien powiadomić o tym użytkownika. Powinny być też rejestrowane wszystkie poprawne działania, tak aby można było przejrzeć dzienniki w celu upewnienia się, że wszystko zadziałało. Zbyt wiele operacji odtwarzania nie powiodło się, ponieważ

ktoś nie zapoznał się z dziennikiem archiwizacji. Jeśli dysponuje się skryptem powiadamiającym, gdy coś nie idzie zgodnie z planem, nie należy zakładać, że wszystko jest w porządku, jeśli nie otrzymało się żadnej wiadomości pocztowej. Co się stanie, jeśli program *cron* będzie wyłączony? Co będzie, gdy jakaś mniej istotna zmiana dokonana w skrypcie spowoduje przerwanie jego działania bez powiadomienia? Co się stanie, gdy program *sendmail* był lub jest wyłączony? *Nigdy nie należy niczego przyjmować za pewnik*.

Właściwe sprawdzanie polecenia `rsh` *lub* `ssh`

Zbyt wiele skryptów sprawdza kod zwracany przez polecenie `rsh/ssh`, lecz już nie kod wygenerowany przez te polecenia na zdalnym komputerze. Czasami warto spróbować wykonać jedno z poniższych poleceń.

```
$ rsh zdalny_komputer operacja_do_zrealizowania ; echo $?
$ ssh zdalny_komputer operacja_do_zrealizowania ; echo $?
```

Łańcuch *`zdalny_komputer`* identyfikuje komputer, z którym można się połączyć za pomocą polecenia `rsh` lub `ssh`, a *`operacja_do_zrealizowania`* — polecenie niedostępne na tym hoście. Okaże się, że wykonane polecenie nie uda się na zdalnym komputerze, lecz program *rsh* lub *ssh* zwróci kod 0, oznaczający pomyślne zrealizowanie operacji. Wynika to stąd, że polecenie `rsh` lub `ssh` powiodło się, niezależnie od tego, czy wykonane przez nie polecenie było udane, czy nie. Właśnie z tego powodu wymagane jest użycie składni podobnej do poniższej (w miejsce programu *rsh* można też zastosować *ssh*).

```
rsh apollo "ls -l /tmp/* ; echo \$?>/tmp/ls.success"
SUCCESS='rsh apollo cat /tmp/ls.success ; rm /tmp/ls.success'
if [ $SUCCESS -eq 0 ] ; then
  # wszystko zadziałało
  echo "Wszystko zadziałało."
else
  echo "Coś poszło nie tak!"
fi
```

Powyższa składnia zwróci kod zdalnego polecenia, a nie tylko polecenia `rsh` lub `ssh`.

Przedstawiona składnia nie zadziała w przypadku narzędzia *csh*, ponieważ nie pozwala ono na przekierowywanie wyjścia. Jeden ze sposobów poradzenia sobie z tym problemem polega na utworzeniu niewielkiego skryptu, który można będzie skopiować za pomocą programu *rcp*. Skrypt może wywoływać program */bin/sh*, aby użytkownik miał pewność, że używa tej powłoki.

Uzyskanie tabeli zawartości z woluminu archiwizacyjnego

Powody, aby zawsze ponownie odczytywać woluminy archiwizacyjne, są dwa. Pierwszy jest taki, że postępując w ten sposób, można najlepiej sprawdzić poprawność działania kopii zapasowej (uproszczenie rzeczywistego procesu odtwarzania danych). Drugim powodem jest to, że tabele zawartości można zapisać w pliku i użyć go podczas procesu odtwarzania, aby sprawdzić, który wolumin zawiera szukany plik.

Najlepszą metodą sprawdzenia, czy wolumin narzędzia *dump* jest nienaruszony, jest wyświetlenie tabeli zawartości z włączoną opcją szczegółowej informacji, posortowanie według numeru i-węzła i odtworzenie ostatniego pliku. W efekcie jest wczytywany cały wolumin i uzyskuje się pewność, że kopie zapasowe narzędzia *dump* są nienaruszone aż do ostatniego pliku zapisanego na taśmie.

Archiwizowanie i odtwarzanie danych za pomocą narzędzia cpio

cpio jest narzędziem o dużych możliwościach. W przeciwieństwie do programu *dump* operuje na poziomie plików. Dlatego radzi sobie ze zmieniającymi się systemami plików trochę lepiej niż program *dump*. Jednak podczas archiwizowania plików narzędzie *cpio* modyfikuje ich czas dostępu (wartość *atime*). Choć narzędzie oferuje opcję resetującą wartość *atime*, powoduje ona zmianę wartości *ctime*. Jeśli nie używa się wersji GNU narzędzia *cpio*, jedną z jego największych zalet jest zgodność z różnymi systemami operacyjnymi. Ponadto program *cpio* wymaga określenia plików, które zostaną mu przekazane za pośrednictwem standardowego wejścia. Powoduje to, że program ten różni się trochę od innych narzędzi archiwizujących.

Korzystanie z narzędzia *cpio* jest bardziej pracochłonne niż z programu *dump*. Chodzi o to, że trzeba wiedzieć trochę więcej na temat działania narzędzia *cpio*, jeśli zamierza się stosować je do regularnego sporządzania systemowych kopii zapasowych. Trzeba znać odpowiedzi na następujące pytania:

- Jak używać programu *find* razem z *cpio* do tworzenia pełnych i przyrostowych kopii zapasowych systemu plików, bez modyfikowania czasu dostępu plików (wartość *atime*)?
- Jakie argumenty dają najlepsze wyniki?
- Jak za pomocą polecenia `rsh` lub `ssh` przesłać kopię zapasową narzędzia *cpio* do zdalnego napędu archiwizującego?
- Jak uzyskać tabelę zawartości woluminu?
- Jak obsługiwać napęd taśmowy i odtwarzać dane z kopii zapasowej utworzonej przy użyciu programu *cpio*?

W przypadku narzędzia *cpio* dobre jest to, że zwykle jego nazwa nie zmienia się (duża zaleta w porównaniu z programem *dump*!).

Użytkownicy systemu MacOS muszą pamiętać o stosowaniu wbudowanego narzędzia *cpio*, jeśli korzystają z wersji systemu nowszej niż 10.4. W przeciwnym razie trzeba użyć narzędzia *ditto*, gdy wymagany jest format programu *cpio*.

Najpierw Czytelnik zapozna się z podstawową składnią programu *cpio*, a następnie z kilkoma przykładowymi poleceniami.

Składnia narzędzia *cpio* w przypadku archiwizowania danych jest następująca:

```
cpio -o [aBcv]
```

Składnia narzędzia *cpio* w przypadku odtwarzania danych jest następująca:

```
cpio -i [Btv] [wzorce]
```

Oto przykładowe polecenia tworzące pełną kopię zapasową katalogu */home* i zapisujące ją na lokalnym napędzie taśmowym:

```
$ cd /home
$ touch level.0.cpio.timestamp
```

Choć polecenie `touch` jest opcjonalne, umożliwia sporządzanie przyrostowych kopii zapasowych.

```
$ find . -print|cpio -oacvB > urządzenie
```

Oczywiście urządzeniem w powyższym poleceniu może też być lokalny plik, jeśli dane archiwizuje się na nośniku optycznym. Polecenia tworzą na lokalnym napędzie taśmowym przyrostową kopię zapasową katalogu */home*.

```
$ cd /home
$ touch level.1.cpio.timestamp
$ find . -newer level.0.cpio.timestamp -print \|cpio -oacvB > urządzenie
```

Polecenia tworzą na zdalnym napędzie taśmowym pełną kopię zapasową katalogu */home*.

```
$ cd /home
$ find . -print|cpio -oacvB \|(rsh zdalny_host dd of=urządzenie bs=5120)
```

Oto bardziej bezpieczna metoda, wykorzystująca program *ssh*:

```
$ find . -print|cpio -oacvB \|(ssh zdalny_host dd of=urządzenie bs=5120)
```

Składnia narzędzia cpio podczas archiwizowania danych

Polecenie `cpio` pobiera swoją listę plików ze standardowego wejścia (`stdin`) i domyślnie przesyła strumień danych do standardowego wyjścia (`stdout`). W celu zapewnienia listy plików przeznaczonych do archiwizacji należy podjąć odpowiednie działania. Oto one:

- Zastosowanie programu *ls* lub *find* (należy na przykład wykonać polecenie `ls | cpio -oacvB`).
- Utworzenie pliku dołączeń, a następnie przesłanie go do standardowego wejścia (`stdin`) narzędzia *cpio* (przykładowo należy zastosować polecenie `cat /tmp/include | cpio -oacvB` lub `cpio -oacvB </tmp/include`).

Wszystkie powyższe operacje tworzą listę dołączeń ze ścieżką *względną* w stosunku do bieżącego katalogu roboczego. Jest to automatycznie wykonywane w przypadku narzędzia *dump*. Z kolei program *cpio* pozwala użyć ścieżek względnych (na przykład `cd /home ; find .`) lub bezwzględnych (na przykład `find /home1`). Jednak zastosowanie bezwzględnych ścieżek poważnie ogranicza możliwości odtwarzania. Jeśli tabela zawartości pliku kopii zapasowej narzędzia *cpio* zawiera ścieżkę */home1/katalog/jakiś_plik*, można go odtworzyć tylko w miejscu identyfikowanym przez ścieżkę */home1/katalog/jakiś_plik* (choć czasami można sobie z tym poradzić za pomocą narzędzia *chroot*, jest to bardzo trudne!). Z kolei jeśli tabela zawartości pokazuje ścieżkę *./home1/katalog/jakiś_plik* lub *home1/katalog/jakiś_plik*, można ją odtworzyć w dowolnym miejscu, przechodząc do innego katalogu, a następnie rozpoczynając proces odtwarzania. A zatem zawsze powinno się używać względnych ścieżek, gdy tworzy się listy dołączeń dla programu *cpio* lub *tar* (choć wersja GNU narzędzia *tar* usuwa bezwzględne ścieżki podczas odtwarzania danych, prawdopodobnie lepiej wyrobić w sobie nawyk stosowania względnych ścieżek, gdy tworzy się listy dołączeń dla jednego z wyżej wymienionych narzędzi archiwizujących).

Użycie narzędzia *find* jest typową metodą tworzenia regularnych systemowych kopii zapasowych, ponieważ umożliwia wykonywanie za pomocą programu *cpio* kopii przyrostowych. Przed rozpoczęciem sporządzania pełnej kopii zapasowej systemu plików lub katalogu w najwyżej położonym katalogu należy utworzyć plik znacznika czasu. Przykładowo, aby w przypadku wersji narzędzia *cpio* wbudowanej w system wykonać przyrostowe kopie zapasowe katalogu */home1*, należy utworzyć plik */home1/level.0.cpio.timestamp*. Następnie można sporządzić

pełną kopię zapasową, korzystając z polecenia `find` (na przykład `find . -print`, wyświetlającego całą zawartość katalogu lub systemu plików). Gdy nadejdzie pora na wykonanie kopii zapasowej poziomu 1, należy utworzyć plik */home1/level.1.cpio.timestamp* i ponownie użyć polecenia `find` (na przykład `find . -newer level.0.cpio.timestamp`, szukającego plików nowszych od pliku `/home1/level.0.cpio.timestamp`). Plik */home1/level.1.cpio.timestamp* może następnie posłużyć do sporządzenia kopii zapasowej poziomu 2 za pomocą polecenia `find` szukającego plików nowszych od tego pliku. Stosując taką metodę, można wygenerować wybraną liczbę poziomów kopii zapasowych.

Opcje polecenia cpio

Można wyróżnić sześć opcji, które powinny być użyte podczas wykonywania regularnych kopii zapasowych za pomocą narzędzia *cpio*. Pierwszych pięć opcji zwykle jest podawanych razem (`-oacvB`), natomiast szósta zazwyczaj jest określana oddzielnie (na przykład `-C 5120`). Warto zauważyć, że opcje `-B` i `-C` wzajemnie się wykluczają. Nie można ich stosować jednocześnie.

- `o` — określa, że powinna zostać utworzona kopia zapasowa;
- `a` — resetuje wartość *atime* przed rozpoczęciem archiwizowania;
- `c` — nakazuje programowi *cpio* użycie formatu nagłówka ASCII;
- `v` — uaktywnia tryb szczegółowych informacji;
- `B`, `C` — umożliwiają określenie rozmiaru bloku;

Ponadto można zidentyfikować urządzenie lub plik, do których program *cpio* może wysyłać swoje dane wyjściowe zamiast do standardowego wyjścia `stdout`. Wszystkie z powyższych opcji (i nie tylko te) są dostępne w wersji GNU programu *cpio*. Wersja ta oferuje również możliwość zastosowania zdalnych urządzeń.

Jeśli to możliwe, należy korzystać z wersji GNU narzędzia cpio!

Wersja GNU programu *cpio* oferuje wiele funkcji. Są trzy bardzo dobre powody, dla których warto z niej korzystać (jeśli tylko jest to możliwe):

- Wersja narzędzia *cpio* wbudowana w system nie jest zbyt przenośna, nawet jeśli w dokumentacji jest stwierdzone inaczej. Jeśli utworzy się kopię zapasową za pomocą wersji GNU programu *cpio*, zawsze będzie można ją odczytać, pod warunkiem że w systemie jest dostępna taka wersja narzędzia *cpio* (niezależnie od rodzaju platformy).
- Przenośny format ASCII też ma ograniczenia. Przykładowo nie radzi sobie z systemem plików mającym więcej niż 65 536 i-węzłów. Ograniczenia tego jest pozbawiony format nagłówka newc dostępny w przypadku wersji GNU programu *cpio*.
- Tak jak program *dump* wersja GNU narzędzia *cpio* obsługuje zdalne urządzenia! Jeśli tylko nie ma problemu z zastosowaniem uwierzytelniania bazującego na programie *rsh*, trzeba jedynie wykonać następujące polecenie:

```
$ -O zdalny_host:/nazwa_urządzenia
```

Wersja GNU narzędzia *cpio* jest dostępna pod adresem *http://www.gnu.org*.

Określanie trybu wyjściowego (o)

Opcja `o` reprezentuje jeden z trzech trybów narzędzia *cpio* (`o`, `i` i `p`) i służy do tworzenia kopii zapasowych. Opcja jest podawana jako pierwsza w grupie kilku argumentów polecenia `cpio`.

Przywracanie czasów dostępu (a)

Jedną z różnic między narzędziami *dump* i *cpio* jest to, że pierwsze archiwizuje dane bezpośrednio na urządzeniu dyskowym, natomiast drugie za pośrednictwem systemu plików. A zatem gdy program *cpio* odczytuje plik w celu jego archiwizacji, zmienia czas dostępu (wartość *atime*). Administratorzy systemów zwykle używają tej wartości, aby sprawdzić, kiedy użytkownik po raz ostatni korzystał z pliku. Pliki, które od dłuższego czasu nie były używane, zwykle są usuwane z komputera w ramach procesu czyszczenia. Jeśli program archiwizujący zmienia czas dostępu pliku, wygląda to tak, jakby wszystkie pliki były przetwarzane każdej nocy. Opcja `a` narzędzia *cpio* pozwala przywrócić oryginalną wartość *atime*.

Przywracanie czasów dostępu powoduje modyfikację wartości *ctime*. Może to wygenerować ostrzeżenia dotyczące włamań, jeśli pod tym względem dokładnie monitoruje się system.

Określanie formatu ASCII (c)

Gdy program *cpio* archiwizuje dane, może je przesyłać do urządzenia archiwizującego, używając kilku formatów nagłówka. Formaty te mogą być w dużym stopniu zależne od platformy i w związku z tym być mało przenośne między systemami. Najbardziej przenośny format (jednak nie całkowicie) to ASCII. Opcja `c` nakazuje programowi *cpio* zastosowanie tego formatu. Jak wspomniano w ramce „Jeśli to możliwe, należy korzystać z wersji GNU narzędzia cpio!", format ASCII może nie być tak przenośny, jak można by pomyśleć. Jeśli naprawdę przenośność jest istotną kwestią, pod uwagę powinno się wziąć użycie wersji GNU narzędzia *cpio*. Jeżeli nie jest to możliwe, powinno się spróbować przenieść pliki narzędzia *cpio* między różnymi wersjami systemu Unix, z których się korzysta. Przynajmniej będzie wiadomo, na czym się stoi. W każdym razie użycie opcji `c` jest nieszkodliwe.

Żądanie wyświetlania szczegółowych informacji (v)

Opcja `v` powoduje, że narzędzie *cpio* kieruje listę archiwizowanych plików do `stderr`. Faktyczne dane kopii zapasowej narzędzia trafiają do standardowego wyjścia `stdout` (archiwizowane dane zawsze są kierowane do standardowego wyjścia, jeśli używana wersja programu *cpio* nie obsługuje opcji `-O`, umożliwiającej określenie wyjściowego pliku lub urządzenia).

Określanie współczynnika bloków o wartości 5120 (B)

Opcja `B` po prostu nakazuje programowi *cpio* przesyłanie swoich danych do standardowego wyjścia `stdout` w postaci bloków o rozmiarze 5120, a nie domyślnym rozmiarze 512. Może to być pomocne w przyspieszeniu procesu archiwizowania. Jednak taki rozmiar ani trochę nie jest bliski wartościom współczynnika bloków preferowanym przez wiele nowoczesnych napędów archiwizujących. W związku z tym powinno się wstawić niżej przedstawioną opcję `C`, jeśli tylko jest dostępna w systemie. Te dwie opcje wzajemnie się wykluczają.

Określanie rozmiaru bloku I/O (C)

Opcja `C` wymaga argumentu i pozwala określić rzeczywisty rozmiar bloku. Jeśli używa się systemu AIX, wartość opcji jest wartością *współczynnika* bloków pomnożonego przez minimalny rozmiar bloku, wynoszący 512. Większość innych odmian systemu Unix pozwala na podanie wartości w bajtach[3].

W każdym razie dla opcji `C` można ustawić dość dużą wartość, dzięki czemu program *cpio* będzie znacznie lepiej współpracował z nowszymi napędami archiwizującymi. Należy powtórzyć, że opcja ta wyklucza się z opcją `B` i zwykle jest podawana oddzielnie wraz ze swoim argumentem. Oto przykład:

```
$ find . -print|cpio -oacv -C 129024 >urządzenie
```

Określanie wyjściowego urządzenia lub pliku (0)

Niektóre wersje narzędzia *cpio* umożliwiają użycie argumentu `-O` *`urządzenie`*, który powoduje przesłanie danych wyjściowych do urządzenia (opcja nie zawsze jest dostępna). Jednakże wszystkie odmiany programu *cpio* domyślnie wysyłają archiwizowane dane do standardowego wyjścia `stdout`. Aby sobie uprościć sprawę, nie trzeba stosować opcji `-O` nawet wtedy, gdy jest dostępna. W celu określenia urządzenia archiwizującego wystarczy przekierować standardowe wyjście `stdout` do pliku lub urządzenia. Metoda ta zawsze się sprawdza, niezależnie od używanej wersji systemu Unix.

Archiwizowanie danych na zdalnym urządzeniu (potokowanie do programu rsh lub ssh)

Wersja narzędzia *cpio* wbudowana w system nie obsługuje automatycznie zdalnych urządzeń tak jak program *dump* (inaczej jest w przypadku wersji GNU narzędzia *cpio*). A zatem aby dane zarchiwizować na zdalnym napędzie, w miejsce łańcucha *`>urządzenie`* trzeba wstawić potok do polecenia `rsh` lub `ssh`.

```
$ find . -print|cpio -oacv \ | rsh zdalny_system dd of=urządzenie bs=5k
```

Oto bardziej bezpieczna wersja polecenia:

```
$ find . -print|cpio -oacv \ | ssh zdalny_system dd of=urządzenie bs=5k
```

Warto zauważyć, że dane są potokowane do programu *dd* uruchamianego na zdalnym komputerze. Ponieważ plikiem wejściowym jest standardowe wejście `stdin`, trzeba jedynie określić plik wyjściowy (`of=`) i rozmiar bloku. Konieczne jest podanie rozmiaru bloku równego 5 kB, ze względu na to, że może być odczytany przez dowolną wersję narzędzia *cpio*.

Odtwarzanie za pomocą programu cpio

Dla programu *cpio* obowiązują te same reguły co w przypadku dowolnego polecenia `restore`. Mam nadzieję, że Czytelnik nie ma w ręku woluminu narzędzia *cpio* zawierającego bardzo ważną systemową kopię zapasową i nigdy wcześniej nie odtwarzał danych za pomocą tego programu. Trzeba pamiętać, aby wielokrotnie testować i ćwiczyć wykonywanie operacji! Choć

[3] Tym razem dziwny okazał się system Unix firmy HP! Nie oferuje podobnej metody ustawiania rozmiaru bloku, a ponadto opcja `-C` służy do czegoś zupełnie innego, a mianowicie do stosowania punktów kontrolnych, które nie mają nic wspólnego ze współczynnikiem bloków (punkty kontrolne nie są złym pomysłem, ale czy nie można było przypisać im innej litery opcji?).

jasno wyraziłem własną opinię, nie ma powodu do obaw. Co prawda odtwarzanie danych z woluminu narzędzia *cpio* nie jest trudne, jednak jest z tym związanych kilka wyzwań, z którymi można mieć do czynienia podczas próby odczytania woluminu programu *cpio*.

W poniższym podpunkcie przyjęto, że dla Czytelnika wolumin utworzony przy użyciu narzędzia *cpio* nie jest tajemnicą i wie on, jaki zastosował rozmiar bloku. W przeciwnym razie należy zapoznać się z zawartością podrozdziału „Jak odczytać ten wolumin?" z rozdziału 23.

Różne wersje narzędzia cpio

To, że wiadomo, w jakim formacie narzędzia *cpio* zapisano wolumin archiwizacyjny, nie oznacza jeszcze, że będzie można go z łatwością odczytać. Wynika to stąd, że choć większość wersji programu *cpio* ma właśnie taką *nazwę*, nie zawsze generują identyczny format. Nawet nagłówek ASCII mający na celu zapewnienie przenośności nie jest rozpoznawany przez wszystkie platformy. Jeżeli po prostu chce się sprawdzić, czy wolumin można odczytać, należy spróbować wykonać proste polecenie `cpio -itv < urządzenie`. Jeśli zadziała — świetnie! W przeciwnym razie można zobaczyć błędy podobne do następujących:

```
Not a cpio file, bad header
```

lub

```
Impossible header type
```

Wersja GNU programu *cpio* pozwoli zaoszczędzić wiele godzin. Jeśli Czytelnik nią dysponuje, może pominąć resztę niniejszego punktu. Oto cytat z dokumentacji wersji GNU narzędzia *cpio*: „Domyślnie program *cpio* tworzy pliki kopii zapasowych w formacie binarnym, aby zachować zgodność ze starszymi wersjami narzędzia. Gdy wyodrębnia się dane z kopii zapasowej, program *cpio* automatycznie rozpoznaje typ pliku kopii i może wczytywać pliki kopii utworzonych w systemach z innym uporządkowaniem bajtów".

Problemy z uporządkowaniem bajtów

Jeśli wolumin odczytuje się na platformie różniącej się od tej, na której go zapisano, może wystąpić problem z uporządkowaniem bajtów. W efekcie prawdopodobnie pojawi się pierwszy z dwóch wyżej wymienionych błędów. Opcje: `b`, `s` i `S` narzędzia *cpio* są po to, aby łatwiej było radzić sobie z problemami z uporządkowaniem bajtów.

```
$ cpio -itbv < urządzenie
# Odwrócenie kolejności bajtów w obrębie każdego słowa.
$ cpio -itsv < urządzenie
# Odwrócenie kolejności bajtów w obrębie każdego półsłowa.
$ cpio -itSv < urządzenie
# Zamiana półsłowa w obrębie każdego słowa.
```

Choć odwrócenie kolejności bajtów może umożliwić odczytanie nagłówka narzędzia *cpio*, może spowodować, że odtworzone pliki staną się bezużyteczne. Jeżeli przy tworzeniu woluminu nie użyto opcji `c`, najlepiej będzie odtworzyć jego zawartość w systemie z takim samym uporządkowaniem bajtów (w celu uzyskania dodatkowych informacji na temat uporządkowania bajtów należy zapoznać się z podrozdziałem „Jak odczytać ten wolumin?" z rozdziału 23.).

Niepoprawny typ nagłówka

Jeśli nie wystąpił problem z uporządkowaniem bajtów, oznacza to, że dane kopii zapasowej narzędzia *cpio* mogły zostać zapisane przy użyciu innego typu nagłówka. Niektóre wersje narzędzia *cpio* mogą automatycznie wykryć część nagłówków, lecz nie wszystkie. Część wersji programu *cpio* jest w stanie zidentyfikować automatycznie tylko jeden typ nagłówka. Może być konieczne poeksperymentowanie z różnymi nagłówkami, aby stwierdzić, którego użyto podczas zapisywania woluminu. Jeśli ma się do czynienia z takim problemem, prawdopodobnie pojawił się komunikat o błędzie `Impossible header type` (również w tym przypadku wersja GNU narzędzia *cpio* może być w stanie wykryć automatycznie dowolny typ nagłówka). Należy spróbować wykonać następujące polecenia:

```
$ cpio -ictv < urządzenie
# Należy spróbować odczytać dane wejściowe w formacie ASCII.
$ cpio -itv -H nagłówek < urządzenie
# Należy spróbować odczytać dane z użyciem nagłówka podanego za opcją H.
```

W miejsce wartości *nagłówek* można wstawić łańcuch `crc`, `tar`, `ustar`, `odc` itp. Należy zajrzeć do dokumentacji, ponieważ opcja `H` nie jest wszędzie dostępna.

```
$ cpio -ictv -H nagłówek < urządzenie
# Połączenie opcji formatu ASCII i nagłówka.
```

Dziwny rozmiar bloku

Wolumin narzędzia *cpio* może być też zapisany z użyciem rozmiaru bloku innego niż oczekiwany przez program. Jeśli rozmiar bloku kopii zapasowej narzędzia *cpio* wynosi 5 kB, można spróbować nakazać programowi zastosowanie takiego rozmiaru przez dodanie opcji `B` do wszystkich wcześniej podanych poleceń (`cpio -itBv`). Jeżeli rozmiar bloku nie jest równy 5 kB, aby program *cpio* zastosował taki rozmiar, na końcu polecenia należy wstawić opcję `-C` *`rozmiar_bloku`* (`cpio -itv -C 5120`).

Czy wykonać pełne czy częściowe odtwarzanie danych, a może tylko wczytać tabelę zawartości?

Po stwierdzeniu, że można odczytać wolumin narzędzia *cpio*, do wyboru będzie kilka operacji:

- odtworzenie zawartości kopii do bieżącego katalogu lub systemu plików;
- odtworzenie plików zgodnych z podanym wzorcem (wzorcem mogą być dane wynikowe polecenia);
- wykonanie jednej z powyższych operacji w ramach interaktywnego procesu zmieniania nazw plików;
- wczytanie tabeli zawartości.

Opcje narzędzia cpio stosowane podczas odtwarzania

Zanim wykona się dowolną z wyżej przedstawionych czynności, należy zapoznać się z kilkoma opcjami odtwarzania danych z woluminu narzędzia *cpio*. Wiele spośród tych opcji jest takich samych co w przypadku tworzenia woluminu programu *cpio*. To takie opcje, jak `B` (pozwala ustawić blok o rozmiarze 5 kB), `c` (umożliwia odczytanie nagłówka ASCII) i `v` (uaktywnia tryb wyświetlania szczegółowych informacji). Ponadto dostępne są następujące opcje:

- `i` — rozpoczyna łańcuch opcji odtwarzania i nakazuje narzędziu *cpio* włączenie trybu wejściowego;
- `t` — jeśli za opcją `i` znajduje się opcja `t`, program *cpio* generuje tabelę zawartości. Opcja `t` nie powoduje odtwarzania żadnych danych z woluminu.
- `k` — nakazuje programowi *cpio* podjęcie próby pominięcia niepoprawnych danych woluminu[4];
- `d` — sprawia, że w razie potrzeby program *cpio* tworzy katalogi;
- `m` — nakazuje narzędziu *cpio* odtworzenie dla plików oryginalnych czasów modyfikacji z chwili, gdy były archiwizowane. Jeśli nie użyje się tej opcji, domyślnie program *cpio* dla odtwarzanego pliku jako czas modyfikacji ustawi czas odtworzenia.

Warto zauważyć, że w tym przypadku domyślne działanie programu *cpio* jest przeciwieństwem domyślnej operacji wykonywanej przez narzędzie *tar*.

- `u` — nakazuje programowi *cpio* bezwarunkowe nadpisanie wszystkich plików;
- `"* wzorzec*"` — odtwarza pliki zgodne z wzorcem;
- `f "* wzorzec*"` — powoduje odtworzenie wszystkich plików z wyjątkiem tych, które są zgodne ze wzorcem;
- `r` — nakazuje narzędziu *cpio* przeprowadzenie interaktywnego procesu zmiany nazw plików. Jeśli pliki są odtwarzane, użytkownik w trakcie trwania operacji jest proszony o zmianę nazwy każdego pliku. Jeżeli użytkownik wstawi pustą wartość, plik nie zostanie odtworzony.

Informowanie narzędzia cpio o urządzeniu, którego ma użyć

W przeciwieństwie do programu *tar* lub *dump* narzędzie *cpio* nie pobiera jako argumentu nazwy urządzenia archiwizującego[5].

Dane narzędziu *cpio* trzeba przekazać za pośrednictwem standardowego wejścia `stdin`. Można to zrobić, używając programu *dd* lub *cat*.

```
$ dd if=urządzenie bs=rozmiar_bloku | cpio -opcje
```

Alternatywnie można po prostu przekierować standardowe wejście `stdin`, tak aby program *cpio* odczytywał dane z urządzenia.

```
$ cpio -opcje < urządzenie
```

[4] Choć opcja ta jest również oferowana przez wersję GNU narzędzia *cpio* w celu zachowania zgodności ze starszymi skryptami powłoki, w rzeczywistości jest ignorowana. Wersja GNU programu *cpio* zawsze próbuje pomijać niepoprawne fragmenty taśmy. A zatem jeśli korzysta się z programu *gcpio*, można zrezygnować z opcji `k`. Niektóre inne wersje narzędzia *cpio* w ogóle nie mają tej opcji.

[5] Będzie tak, jeśli nie postanowi się zastosować opcji `-I` obsługiwanej przez niektóre wersje narzędzia *cpio*. Trzeba w tym miejscu kolejny raz przypomnieć, że w książce skoncentrowano się na opcjach, które można wykorzystać niemal wszędzie.

Przykłady odtwarzania danych przy użyciu narzędzia cpio

Na tym etapie jedyną kwestią jest to, jakie są wymagane opcje. Najprościej poradzić sobie z tym przez prezentację przykładowych poleceń demonstrujących operacje, które można wykonać na woluminie narzędzia *cpio*. W tych poleceniach umieszczono kilka opcji, które nie są obligatoryjne. Choć nie są wymagane, wiele z nich ułatwia operację lub sprawia, że przebiega bardziej stabilnie. Ponieważ część opcji może nie być przydatna w przypadku określonego zastosowania, można bez obaw z nich zrezygnować.

Generowanie listy plików woluminu narzędzia cpio

Poniższe polecenie odczytuje wolumin narzędzia *cpio* w blokach liczących 5120 bajtów (opcja `B`), podczas odczytu nagłówka używa formatu ASCII (opcja `c`), w miarę możliwości pomija niepoprawne fragmenty woluminu (opcja `k`) i wyświetla jedynie tabelę zawartości (opcja `t`) w trybie szczegółowej informacji (opcja `v`), z uwzględnieniem stylów listy (`ls -l`).

```
$ cpio -iBcktv < urządzenie
```

Przeprowadzanie procesu odtwarzania całego systemu plików

Poniższe polecenie odczytuje wolumin narzędzia *cpio* w blokach liczących 5120 bajtów (opcja `B`), podczas odczytu nagłówka używa formatu ASCII (opcja `c`) i w razie potrzeby tworzy katalogi (opcja `d`). Gdy to możliwe, polecenie pomija niepoprawne fragmenty woluminu (opcja `k`), przywraca oryginalny czas modyfikacji pliku (opcja `m`), bezwarunkowo nadpisuje pliki (opcja `u`) i generuje listę nazw plików odtwarzanych po wczytaniu.

```
$ cpio -iBcdkmuv < urządzenie
```

Oczywiście można zrobić to samo bez bezwarunkowego nadpisywania plików (opcja `u`).

```
$ cpio -iBcdkmv < urządzenie
```

Odtwarzanie danych zgodnych ze wzorcem

W celu odtworzenia plików zgodnych z określonym wzorcem wystarczy w wierszu polecenia podać szukane wzorce.

```
$ cpio -iBcdkmuv "wzorzec1" "wzorzec2" "wzorzec3" < urządzenie
```

Wzorzec używa znaków wieloznacznych nazw plików, lecz nie wyrażeń regularnych[6].

Znaki wieloznaczne nazw plików funkcjonują podobnie jak znaki zastosowane w wierszu poleceń (na przykład `*ome*` dopasowuje zarówno nazwę *home1*, jak i *rome*). Program *cpio* jest jedynym wbudowanym narzędziem odtwarzającym dane, które obsługuje w ten sposób znaki wieloznaczne. Jeśli na przykład zamierza się odtworzyć wszystkie pliki znajdujące się w katalogu domowym (*/home1/jnowak*), należy wykonać następujące polecenie:

```
$ cpio -iBcdkmuv "*jnowak*"
```

[6] Czytelnik może dowiedzieć się więcej na temat wyrażeń regularnych, niż mógłby sobie wyobrazić, z książki *Wyrażenia regularne* napisanej przez Jeffreya Friedla (wydawnictwo Helion), którą gorąco polecam. Zrozumienie, czym są wyrażenia regularne i do czego służą, jest wartościowym doświadczeniem, pozwalającym znacznie bardziej owocnie korzystać z takich narzędzi, jak *grep*, *sed*, *awk* i *vi*.

Zastosowanie wzorca w powyższej postaci spowoduje użycie rozwinięcia nazwy dla plików znajdujących się w kopii zapasowej. Jeśli wzorca nie umieści się w znakach cudzysłowu, powłoka rozwinie znaki wieloznaczne i program *cpio* uzyska listę nazw plików istniejących w systemie i zgodnych ze wzorcem `*jnowak*`. Jeżeli usunie się część plików lub przejdzie do innego katalogu, wyniki nie będą zgodne z oczekiwanymi!

Aby odtworzyć wszystkie pliki z wyjątkiem pasujących do określonego wzorca, należy zastosować opcję `f` i podać wzorce wykluczające.

```
$ cpio -iBcfdkmuv "wzorzec1" "wzorzec2" "wzorzec3" < urządzenie
```

Interaktywny proces zmiany nazw plików

Choć poniżej podano takie samo polecenie jak we wcześniejszym podpunkcie „Odtwarzanie danych zgodnych ze wzorcem", prosi ono użytkownika o wykonanie interaktywnego procesu (opcja `r`) zmiany nazwy odtwarzanych plików.

```
$ cpio -iBcdkmruv < urządzenie
```

Choć poniżej zamieszczono identyczne polecenie jak we wcześniejszym podpunkcie „Przeprowadzanie procesu odtwarzania całego systemu plików", prosi ono użytkownika o wykonanie interaktywnego procesu (opcja `r`) zmiany nazwy wszystkich odtwarzanych plików.

```
$ cpio -iBcdkmruv "wzorzec" < urządzenie
```

Inne przydatne opcje

`b`, `s` i `S`
: Opcje te służą do zamieniania miejscami bajtów, gdy występują problemy z uporządkowaniem bajtów. Z opcji należy korzystać w ostateczności, ponieważ do tej pory nie widziałem, żeby zakończyło się to zupełnym sukcesem. Jest jedna sytuacja, w której użycie opcji może okazać się przydatne. Ma ona miejsce wtedy, gdy wolumin utworzony na komputerze z formatem Little Endian próbuje się odczytać przy wykorzystaniu systemu z formatem zapisu danych Big Endian (w celu uzyskania dodatkowych informacji należy zajrzeć do podrozdziału „Jak odczytać ten wolumin?" z rozdziału 23.). Osoba wykonująca kopię zapasową za pomocą narzędzia *cpio* nie zastosowała opcji `-c`. W efekcie jedyną możliwością odczytania woluminu jest zamiana bajtów miejscami.

  ```
  $ dd if=urządzenie bs=10240 conv=swab | cpio -opcje
  ```

  Później okazuje się, że słowa zawarte w kopii zapasowej mają kolejność odwrotną do wymaganej. W związku z tym uzyskano odtworzone pliki, których nie można odczytać. Przypuszczalnie podczas odtwarzania danych program *cpio* mógł zamienić słowa miejscami. Warto zauważyć, że do zwykłego polecenia `cpio` dodano opcję `b`.

  ```
  $ dd if=urządzenie bs=10240 conv=swab | cpio -iBcdkmubv < urządzenie
  ```

  Użycie opcji `b` odpowiada jednoczesnemu zastosowaniu opcji `s` i `S`. Problem polega na tym, że wszystkie operacje zamiany bajtów miejscami zachodzą, gdy program *dd* lub *cpio* nie zna formatu pliku. Co będzie, gdy spodziewane 8-bajtowe słowa nie będą wcale 8-bajtowe? Co się stanie, gdy słowa te będą 10-bajtowe? Nie spotkałem nikogo, komu do końca udało się skorzystać z tych opcji. A zatem jeśli Czytelnik będzie miał więcej szczęścia, niech mi wyśle wiadomość pocztową!

6
: Opcja pozwala odczytać archiwa szóstej edycji systemu Unix. Z opcji należy skorzystać, aby odczytać *naprawdę* stare kopie zapasowe narzędzia *cpio*.

Odtwarzanie danych do innego katalogu

Jeśli woluminy archiwizacyjne utworzono przy użyciu względnych ścieżek, nie stanowi to problemu. Wystarczy za pomocą polecenia `cd` przejść do katalogu, w którym mają się znaleźć odtwarzane dane, a następnie wykonać polecenia `cpio`. Jeżeli nie wiadomo, czy wolumin zapisano z uwzględnieniem względnych ścieżek, należy wykonać polecenie `cpio -itv < urządzenie` i poszukać ścieżek plików. Jeśli rozpoczynają się od znaku `/`, oznacza to, że wolumin utworzono z zastosowaniem bezwzględnych ścieżek. W tym przypadku można zrobić jedną z dwóch rzeczy:

Użycie dowiązania symbolicznego
: Jeśli korzysta się z systemu Unix, powinno być dostępne polecenie `chroot`. Jeżeli używa się platformy innej niż uniksowa lub nie jest dostępne polecenie `chroot`, trzeba będzie być bardziej kreatywnym. Gdy dane trzeba odtworzyć do innego katalogu i kopię zapasową utworzono przy użyciu bezwzględnych ścieżek, można zdefiniować dowiązanie symboliczne między katalogami */home2* i */home1* (`ln -s /home2 /home1`). W ten sposób wszystkie pliki, które mają trafić do katalogu */home1*, w rzeczywistości znajdą się w katalogu */home2*. Coś takiego zadziała tylko wtedy, gdy w systemie nie podłączono katalogu */home1*. Jeśli taki katalog już istnieje, trzeba go *odłączyć*. Oczywiście, jest to niepożądana sytuacja, dlatego też woluminy archiwizacyjne powinno się zapisywać z zastosowaniem względnych ścieżek.

Zastosowanie wersji GNU narzędzia cpio
: Naprawdę jest to najlepsze rozwiązanie. Wersja GNU programu *cpio* ma opcję „żadnych bezwzględnych ścieżek", która z wszystkich bezwzględnych ścieżek usuwa znak `/` i odtwarza pliki ze ścieżką względną w stosunku do bieżącego katalogu.

Użycie funkcji narzędzia cpio kopiującej katalogi

Jeśli trzeba przenieść katalog z jednego miejsca w drugie, można spróbować użyć tej rzadko stosowanej funkcji narzędzia *cpio*. W tym celu należy wykonać następujące polecenie:

```
$ cd stary_katalog ; find . -print | cpio -padlmuv nowy_katalog
```

Polecenie przenosi *stary_katalog* w miejsce *nowego_katalogu*, resetuje czasy dostępu (opcja `a`), w razie potrzeby tworzy katalogi (opcja `d`), w miarę możliwości łączy pliki (opcja `l`), przywraca oryginalne czasy modyfikacji (opcja `m`), bezwarunkowo nadpisuje wszystkie pliki (opcja `u`) i wyświetla szczegółowe informacje na temat kopiowanych plików (opcja `v`).

Niektóre wersje systemu Unix mają też opcję `-L`, która powoduje, że program *cpio* uwzględnia dowiązania symboliczne, kopiując zamiast samych dowiązań katalogi i pliki, na które wskazują. Jeśli użyje się tej opcji, trzeba się upewnić, że polecenie `find` przekazujące programowi *cpio* plik korzysta z opcji `-follow`. Jeżeli tak nie będzie, uzyska się nieprzewidywalne rezultaty.

Gdyby skompilować listę wszystkich opcji dostępnych w przypadku każdej platformy uniksowej, byłaby bardzo długa. Zależnie od platformy może istnieć mnóstwo innych przydatnych opcji, które sprawiają, że narzędzie *cpio* jest jeszcze bardziej pożyteczne. Trzeba też wspomnieć

o kilku dodatkowych funkcjach wersji GNU narzędzia *cpio*. Warto zapoznać się z dokumentacją dołączoną do posiadanej wersji programu *cpio*. Trzeba być świadomym tego, że jeśli zastosuje się dowolną z opcji mających wpływ na sposób tworzenia kopii zapasowej przez program *cpio*, może zmniejszyć się jej przenośność.

Archiwizowanie i odtwarzanie danych za pomocą narzędzia tar

tar jest najbardziej popularnym narzędziem archiwizującym omawianym w niniejszym rozdziale. Wiele spośród plików pobieranych z internetu ma postać zwykłych lub skompresowanych plików *tar*. Jedno godne uwagi ograniczenie narzędzia *tar* jest takie, że zawsze miało problemy z wyjątkowo długimi ścieżkami. Choć samo narzędzie *tar* nie jest zwykle wykorzystywane do codziennych operacji archiwizowania i odtwarzania danych, jego wersja GNU często jest używana przez inne programy open source, takie jak *Amanda* (rozdział 4.).

Jak wspomniano, wbudowana wersja programu *tar* nie zachowuje czasów dostępu archiwizowanych plików. Jeśli jest to istotna kwestia, należy korzystać z wersji GNU narzędzia *tar*, która umożliwia pozostawienie czasu dostępu bez zmian.

Składnia polecenia tar w przypadku archiwizowania danych

Podstawowe polecenie `tar` wygląda tak:

```
$ tar [cx]vf urządzenie wzorzec
```

Przyjrzyjmy się kilku przykładowym poleceniom. W celu utworzenia kopii zapasowej katalogu o nazwie *wzorzec* należy wykonać poniższe polecenie.

```
$ tar cvf urządzenie wzorzec
```

Aby zrobić to samo, lecz przy współczynniku bloków o wartości 20, należy zastosować następujące polecenie:

```
$ tar cvbf 20 urządzenie wzorzec
```

Aby dodatkowo narzędzie *tar* sprawdzało zapisywane dane (opcja dostępna tylko w wersji GNU programu *tar*)[7], należy użyć poniższego polecenia.

```
$ gtar cvWbf 20 urządzenie wzorzec
```

Aby utworzyć kopię zapasową wszystkich plików i katalogów w bieżącym katalogu, których nazwy zaczynają się od litery `a`, należy wykonać następujące polecenie:

```
$ tar cvf urządzenie a*
```

Jeśli korzysta się z wersji 10.4 systemu Mac OS lub nowszej, trzeba pamiętać, aby użyć narzędzia *tar* wbudowanego w system. W przypadku starszych wersji systemu konieczne będzie zastosowanie narzędzia *hfstar*.

[7] Jest to kolejny powód, dla którego powinno się korzystać z narzędzia *gtar*, gdy regularnie sporządza się systemowe kopie zapasowe.

Opcje polecenia tar

Narzędzie *tar* ma dwie wielkie zalety. Pierwszą jest poziom uzyskanej akceptacji, drugą — naprawdę krótka lista opcji. Oto one:

`c`
: Opcja nakazuje programowi *tar utworzenie* archiwum (kopii zapasowej).

`v`
: Opcja uaktywnia tryb narzędzia *tar*, w którym są wyświetlane *szczegółowe* informacje. Program pokazuje nazwę i rozmiar każdego archiwizowanego pliku.

`W`
: Opcja dostępna wyłącznie w wersji GNU narzędzia *tar* nakazuje mu podjęcie próby sprawdzenia plików po ich umieszczeniu w kopii zapasowej.

`b` *współczynnik_bloków*
: Opcja nakazuje narzędziu *tar* odczytywanie i zapisywanie w blokach o rozmiarze *n* bajtów, gdzie *n* jest wartością użytego współczynnika bloków pomnożonego przez minimalny rozmiar bloku (dla określonego systemu operacyjnego). Choć zwykle wartość ta wynosi 512, może być to 1024. Wynikowa wartość, nazywana *rozmiarem bloku*, może zawierać się w przedziale od 512 do 10 240. Blok o rozmiarze 10 240 normalnie wskazywałby na *współczynnik* bloków wynoszący 20, ponieważ 20 pomnożone przez 512 daje 10 240. Jeśli nie określi się wartości opcji `b`, zostanie użyta domyślna. Choć zazwyczaj wartość ta wynosi 20, może być nawet równa 1.

`f` *urządzenie*
: Opcja nakazuje programowi *tar* zapisywanie danych na urządzeniu określonym za pomocą argumentu *urządzenie*, a nie domyślnym napędzie taśmowym używanej platformy. *Urządzeniem* może być plik na dysku, nośnik optyczny, napęd taśmowy lub standardowe wyjście (`stdout`). Jeśli korzysta się z wersji GNU narzędzia *tar*, urządzeniem może być też napęd taśmowy zdalnego komputera (należy zapoznać się z poniższą ramką). Aby przesłać dane do standardowego wyjścia `stdout`, w miejsce nazwy urządzenia należy wstawić znak `-` (nie jest to możliwe w przypadku wszystkich platform).

wzorzec
: Argument ten generuje listę dołączeń narzędzia *tar*. Ponieważ argument bazuje na składni rozwinięcia nazwy pliku, aby zarchiwizować wszystko, czego nazwa rozpoczyna się od litery `a`, jako argument należy wstawić `a*`. W miejsce argumentu można wstawić dowolną nazwę pliku lub katalogu. W efekcie zostanie zarchiwizowane wszystko, co znajduje się w tym katalogu.

Choć wersja GNU narzędzia *tar* może odczytać kopię zapasową utworzoną za pomocą dowolnej innej wersji programu, odwrotna sytuacja niekoniecznie jest możliwa. Niektóre wbudowane wersje programu *tar* nie są w stanie odczytać kopii zapasowych sporządzonych przy użyciu wersji GNU narzędzia.

Wyświetlanie listy plików standardowego wejścia

Podobnie do programu *cpio* większość wersji narzędzia *tar* nie pozwala na generowanie listy plików znajdujących się w standardowym wejściu, które mają być zarchiwizowane. Jednak wersja GNU narzędzia *tar* dzięki opcji `-T` oferuje taką możliwość. Opcja ta pozwala określić

Jeśli to możliwe, należy korzystać z wersji GNU narzędzia tar

Wersja GNU narzędzia *tar* jest wyjątkowo popularna. Poza możliwością odczytu kopii zapasowej utworzonej przez dowolną inną wersję programu *tar* zapewnia wiele innych funkcji. Oto kilka z najpopularniejszych rozszerzeń:

- Opcja `-d`, wykorzystując narzędzie *diff*, dokonuje porównania kopii zapasowej i systemu plików. Polega to na odczytaniu taśmy i porównaniu jej zawartości z plikami wchodzącymi w skład systemu plików. Zgłaszane są wszelkie różnice.
- Opcja `-a` resetuje czasy dostępu (wartość *atime*).
- Opcja `-F` uruchamia skrypt po osiągnięciu przez narzędzie *tar* końca woluminu. Opcja może być użyta do automatycznego wymieniania woluminów za pomocą zmieniarki nośników.
- Opcje `-Z` i `-z` automatycznie przetwarzają kopię zapasową odpowiednio za pomocą programu *compress* lub *gzip*.
- Opcja `-f` obsługuje nazwy zdalnych urządzeń.
- Domyślnie wersja GNU narzędzia *tar* usuwa znak `/` z bezwzględnych ścieżek podczas tworzenia lub odczytywania kopii zapasowej (można temu zapobiec, używając opcji `-p`).
- Niektórzy preferują styl argumentów oferowany przez wersję GNU programu *tar*. Zamiast polecenia `tar cvf` można zastosować polecenie `tar -create -verbose -file`.

Wersja GNU narzędzia *tar* jest dostępna pod adresem *http://www.gnu.org*.

plik zawierający listę plików przeznaczonych do archiwizacji. Jeśli chce się podać nazwy plików do zarchiwizowania za pośrednictwem standardowego wejścia, należy zastosować wersję GNU narzędzia *tar* i wstawić znak `-` jako plik dołączeń. Zwykle znak ten nakazuje programowi *tar* sprawdzenie standardowego wejścia zamiast podanej nazwy pliku. Załóżmy, że postanowiono z poziomu katalogu */home/jnowak* uruchomić program *find* i zarchiwizować wszystkie znalezione tam pliki.

```
# cd /home/jnowak ; find . -print | tar cvf /dev/rmt/0cbn -T -
```

Powyższe polecenie spowoduje, że narzędzie *tar* otrzyma wyniki działania programu *find* w postaci listy plików, które należy uwzględnić w kopii zapasowej.

W tabeli 3.2 wymieniono wbudowane wersje narzędzia *tar* obsługujące listę dołączeń.

Tabela 3.2. Wersje programu tar obsługujące listę dołączeń

Wersja programu tar	Opcja
AIX	`-L`
DG-UX, SunOS, Solaris	`-I`
FreeBSD, Linux, GNU *tar*	`-T`

Składnia narzędzia tar w przypadku odtwarzania danych

Kopię zapasową narzędzia *tar* można bardzo łatwo odczytać. Jeśli nawet podczas tworzenia takiej kopii zastosowano współczynnik bloków, w czasie odtwarzania danych nie będzie potrzebny. Narzędzie *tar* automatycznie się tym zajmuje (czy może czytając to, Czytelnik pomyślał:

„Jak cudownie…"?). Aby odczytać kopię zapasową utworzoną za pomocą programu *tar*, należy wykonać jedno z poniższych poleceń.

```
$ tar xvf urządzenie
```

lub

```
$ tar xvf urządzenie wzorzec
```

Opcja `x` nakazuje programowi *tar* przeprowadzenie operacji wyodrębniania (odtwarzania) danych z kopii zapasowej. Argumenty: `v`, `f` i *`urządzenie`* działają tak samo jak w przypadku tworzenia kopii zapasowej.

Odtwarzanie wybranych elementów kopii zapasowej

Podczas procesu odtwarzania można określić nazwy plików, które mają być uwzględnione. W tym celu za nazwą urządzenia należy podać jedną lub więcej *ścieżek*. Jednak ważne jest, aby zwrócić uwagę, że ścieżka musi być *idealnie* zgodna ze ścieżką zawartą w kopii zapasowej programu *tar*. Gdy tak nie będzie, dane nie zostaną odtworzone. W przeciwieństwie do narzędzia *cpio*, w przypadku programu *tar* nie są obsługiwane znaki wieloznaczne. Jeśli jednak poda się nazwę katalogu, zarchiwizowana zostanie cała jego zawartość. Trzeba pamiętać o tym, że podana nazwa katalogu musi być całkowicie zgodna z nazwą katalogu zawartego w kopii.

Rozważmy następujący przykład. Istnieje podkatalog *home* i za pomocą narzędzia *tar* tworzymy dla niego kopię zapasową o nazwie *plik.tar*. W tym celu można wykonać polecenie `tar cvf plik.tar home` lub `tar cvf plik.tar ./home`. Przyjrzymy się, jak to wpływa na to, co trzeba zrobić, aby odtworzyć dane.

```
$ tar cvf home.tar ./home
a ./home/ OK
a ./home/plik OK
a ./home/plik.2 OK
```

Jeśli dane zarchiwizuje się z zastosowaniem ścieżki *./home*, trzeba je odtwarzać z uwzględnieniem tej samej ścieżki.

```
$ tar xvf home.tar home
tar: blocksize = 5
$ tar xvf home.tar ./home
tar: blocksize = 5

x ./home, 0 bytes, 0 tape blocks
x ./home/plik, 0 bytes, 0 tape blocks
x ./home/plik.2, 0 bytes, 0 tape blocks
```

W poniższym poleceniu nazwę archiwizowanego katalogu *home* podano jako wzorzec.

```
$ tar cvf home.tar home
a home/ OK
a home/plik OK
a home/plik.2 OK
```

Warto zauważyć, że jeśli dane zarchiwizuje się z zastosowaniem wzorca *home*, trzeba je odtwarzać z uwzględnieniem tego samego wzorca. Wzorzec *./home* nie zadziała.

```
$ tar xvf home.tar ./home
tar: blocksize = 5
$ tar xvf home.tar home
tar: blocksize = 5
x home, 0 bytes, 0 tape blocks
x home/plik, 0 bytes, 0 tape blocks
x home/plik.2, 0 bytes, 0 tape blocks
```

Jeżeli użytkownik nie zna nazwy pliku, który zamierza odtworzyć, i nie chce mu się odtwarzać całej kopii zapasowej, może utworzyć tabelę zawartości i poszukać w niej tej nazwy. Wykonując poniższe polecenie, musi najpierw wygenerować tabelę zawartości kopii zapasowej.

```
tar tf urządzenie > jakiś_plik
```

Jeśli operację taką przeprowadzi się dla kopii zapasowej z powyższego przykładu, uzyska się plik z następującą tabelą zawartości:

```
home/
home/plik
home/plik.2
```

Jeżeli zna się nazwę pliku, na przykład *plik*, można go zlokalizować za pomocą programu *grep*.

```
# grep plik jakiś_plik
home/
home/plik
home/plik.2
```

W takim przypadku powinno się następnie wykonać następujące polecenie:

```
$ tar xvf urządzenie home/plik
```

Sztuczka powodująca, że narzędzie tar używa znaków wieloznacznych podczas procesu odtwarzania

Istnieje zabieg sprawdzający się zwykle w przypadku taśmy, a także plików *.tar* zapisanych na dysku. Sztuczka polega na jednoczesnym wykonaniu dwóch poleceń `tar`.

```
$ tar xvf urządzenie `tar tf urządzenie | grep 'wzorzec'`
```

Jeśli sztuczki tej użyje się w przypadku napędu taśmowego, trzeba sprawdzić, czy oferuje on możliwość przewijania. W przeciwnym razie sztuczka nie zadziała! Można też dodać polecenie `sleep`, aby ustalić czas przewinięcia taśmy.

```
$ tar xvf urządzenie `tar tf urządzenie | grep 'wzorzec' ; sleep 60`
```

Zmiana podczas odtwarzania danych właściciela, uprawnień i atrybutów

Choć domyślne działania narzędzia *tar* mogą być inne w różnych systemach, większość jego wersji umożliwia zastosowanie podczas odtwarzania trzech opcji:

`m`

Standardowo odtwarzane pliki zachowują czasy modyfikacji, które miały w czasie ich archiwizowania. Opcja zmienia czas modyfikacji na czas odtwarzania. W przypadku narzędzia *cpio* jest odwrotnie.

Domyślny sposób traktowania czasów modyfikacji przez narzędzie *tar* podczas procesu odtwarzania jest przeciwieństwem postępowania programu *cpio*.

`o`

Opcja nakazuje narzędziu *tar*, aby użytkownika ustanowił właścicielem wszystkich odtwarzanych plików. Jest to domyślne postępowanie w przypadku innych użytkowników niż *root*. Jeśli nie użyje się tej opcji, pliki wyodrębnione przez użytkownika *root* będą miały identyfikatory użytkowników i grup zapisane w pliku kopii zapasowej *.tar*.

p
Domyślnie narzędzie *tar* nie odtwarza wszystkich atrybutów plików. Uprawnienia plików są określane przy użyciu bieżącej konfiguracji *umask* zamiast uprawnień oryginalnych plików. Ponadto dla wszystkich plików, których użytkownik nie jest właścicielem, nie jest odtwarzany bit `setuid` i bit „lepkości" (*sticky bit*). Opcja nakazuje narzędziu *tar* zastosowanie uprawnień oryginalnych plików, włącznie z wszystkimi specjalnymi atrybutami, takimi jak bit *setuid* (aby ustawić dla plików innych użytkowników bity *setuid* i „lepkości", trzeba mieć uprawnienia użytkownika *root*).

Kilka innych fajnych rzeczy na temat programu tar

Narzędzie *tar* oferuje wiele opcji. Powinno się przeczytać dokumentację, aby zapoznać się z nimi wszystkimi, ponieważ mogą okazać się bardzo przydatne.

Szukanie wszystkiego, co jest zawarte w katalogu

Czasami rzeczy znajdujące się poniżej katalogu nie są tym, na co wyglądają. Jeśli przed usunięciem katalogu tworzy się jego „ostatnie archiwum", można chcieć sprawdzić wszystkie istniejące dowiązania symboliczne. Do tego służy opcja `-h`. Trzeba zadbać o to, aby miejsca dostępnego na taśmie było dużo!

Zastosowanie narzędzia tar do przenoszenia katalogu

Jak wspomniano, program *cpio* oferuje wbudowane polecenie służące do przenoszenia katalogów. Problem polega na tym, że wiele osób nie pamięta składni tego polecenia, gdy trzeba z niego skorzystać. Jednak katalog można przenieść również za pomocą narzędzia *tar*. W tym celu najpierw przy użyciu polecenia `cd` należy przejść o jeden poziom powyżej katalogu, który zamierza się przenieść.

```
$ cd stary_katalog ; cd ..
```

W dalszej kolejności za pomocą narzędzia *tar* i zestawu nawiasów należy utworzyć powłokę podrzędną, która rozpakuje zawartość katalogu w nowym miejscu (warto zwrócić uwagę na użycie opcji `p`, aby zapewnić, że program *tar* utworzy nowy katalog z tymi samymi uprawnieniami co stary).

```
$ tar cf - stary_katalog | (cd nowy_katalog ; cd .. ; tar xvpf -)
```

Znak `-` widoczny na końcu polecenia `tar cf` nakazuje przesłanie jego danych do standardowego wyjścia `stdout` (pominięto opcję `v`, aby uniknąć dwukrotnego wyświetlania nazw plików). Z kolei znak `-` wstawiony na końcu polecenia `tar xvf` nakazuje mu poszukanie danych w standardowym wejściu `stdin`. Przez umieszczenie w nawiasach poleceń `cd nowy_katalog ; tar xvpf -` utworzono powłokę podrzędną, aby zawartość katalogu *stary_katalog* została wyodrębniona w katalogu *nowy_katalog*.

Choć składnia polecenia może wydać się trochę złożona, cechuje się dużą przenośnością. Polecenie będzie trochę krótsze, gdy zastosuje się następujący zapis:

```
$ cd nadrzędny_katalog ; tar cf - stary_katalog | (cd nadrzędny_katalog_nowego_katalogu ;
    tar xvpf -)
```

Miałem do czynienia z osobami próbującymi przenieść katalog domowy użytkownika przez przejście do tego katalogu przy użyciu polecenia `cd`, a następnie utworzenie pliku *.tar* z wykorzystaniem wzorca *. Problem w tym przypadku polega na tym, że nie zostaną uwzględnione pliki z kropką w nazwie, takie jak *.profile*, *.cshrc* i *.emacs*. Słyszałem, jak jedna z tych osób powiedziała: „Muszę użyć `.*`, a nie *!". Trzeba zawsze pamiętać, że z wyrażeniem `.*` jest zgodny łańcuch `..` (nadrzędny katalog). *Oznacza to, że kopia zapasowa uwzględni też nadrzędny katalog*. Właśnie z tego powodu znacznie prostsze będzie przejście o jeden poziom wyżej w hierarchii katalogów, a następnie zarchiwizowanie za pomocą narzędzia *tar* zawartości katalogu. Można też utworzyć plik kopii zapasowej z zastosowaniem łańcucha `...` Preferuję wcześniejszą metodę, ponieważ w jej przypadku są widoczne katalogi, w których znajdują się archiwizowane pliki.

W przytoczonym przykładzie łańcuch *nadrzędny_katalog* identyfikuje katalog nadrzędny w stosunku do katalogu *stary_katalog*. Z kolei w miejsce łańcucha *nadrzędny_katalog_nowego_katalogu* należy wstawić nazwę nadrzędnego katalogu nowego katalogu. Jeśli na przykład przeniesiono by katalog */home1/fred* do katalogu */home2*, w miejsce łańcuchów: *nadrzędny_katalog*, *stary_katalog* i *nadrzędny_katalog_nowego_katalogu* należałoby wstawić odpowiednio katalogi: */home1*, *fred* i */home2*. Trzeba wiedzieć, co się podaje. Jeden z problemów związanych z narzędziem *tar* polega na tym, że zbyt często wykonuje się polecenie `tar cvf`. W efekcie pewnego dnia trzeba będzie zastosować polecenie `tar xvf` i zamiast opcji `x` omyłkowo wpisze się `c`. Można się domyślić, co się wydarzy. Plik archiwum *.tar* uszkodzi się bezpowrotnie. Jest to jedna z najczęściej poruszanych kwestii na grupach dyskusyjnych — nigdy nie było dobrej odpowiedzi.

Odtwarzanie do alternatywnej lokalizacji

Jeśli podczas tworzenia plików archiwum narzędzia *tar* zastosuje się względne ścieżki, odtworzenie danych do alternatywnej lokalizacji będzie bardzo łatwe. Wystarczy zmienić ścieżkę katalogów na coś innego niż ścieżka oryginalnego punktu podłączenia (na przykład */home1*), a następnie rozpocząć proces odtwarzania. W razie potrzeby program *tar* utworzy katalogi.

Jeżeli nie utworzono pliku archiwum programu *tar* z użyciem względnych ścieżek, w celu usunięcia znaków / można skorzystać z wersji GNU narzędzia *tar*.

W części rozdziału poświęconej programowi *cpio* można przeczytać na temat względnych ścieżek i powodów, dla których są tak ważne.

Archiwizowanie i odtwarzanie danych za pomocą narzędzia dd

Program *dd* jest niemal taki sam jak reszta narzędzi archiwizujących. Jednakże do pewnych zastosowań nadaje się w szczególny sposób.

Podstawowe opcje programu dd

```
# dd if=urządzenie of=urządzenie bs=rozmiar_bloku
```

Powyższe opcje są stosowane prawie zawsze podczas uruchamiania programu *dd*. Opcje programu omówiono w dalszej części rozdziału.

Określanie pliku wejściowego

Argument `if=` pozwala określić plik wejściowy lub plik, z którego program *dd* skopiuje dane. Archiwizowany może być plik lub niesformatowana partycja (na przykład `dd if=/dev/dsk/c0t0d0s0` lub `dd if=/home/plik`). Jeśli swoich danych program ma poszukać w standardowym wejściu `stdin`, nie trzeba używać tego argumentu.

Określanie pliku wyjściowego

Argument `of=` pozwala określić plik wyjściowy lub plik, w którym program *dd* umieści dane. Może to być plik na dysku, nośnik optyczny, inna niesformatowana partycja lub napęd taśmowy[8] (na przykład `dd of=/kopia/plik` lub `dd of=/dev/rmt/0n`). Jeżeli dane wysyła się do standardowego wyjścia `stdout`, nie trzeba używać tego argumentu.

Ustalanie rozmiaru bloku

Argument `bs=` określa rozmiar bloku lub ilość danych, które zostaną przesłane w ramach jednej operacji wejścia-wyjścia. Choć wartość argumentu standardowo jest wyrażona w bajtach, w przypadku większości wersji programu *dd* może być też określona w kilobajtach, przez dodanie litery `K` na końcu liczby (na przykład `10 K`). Rozmiar bloku różni się od współczynnika bloków używanego przez narzędzia *dump* i *tar* i mnożonego przez stałą wartość nazywaną minimalnym rozmiarem bloku. Pomnożenie współczynnika bloków wynoszącego 20 przez minimalny rozmiar bloku o wartości 512 daje rzeczywisty rozmiar bloku równy 10 240 bajtów lub 10 kilobajtów. Warto zauważyć, że w przypadku odczytywania lub zapisywania danych za pośrednictwem potoku program *dd* domyślnie używa rozmiaru bloku równego 1.

Zmiana rozmiaru bloku nie ma wpływu na sposób fizycznego zapisywania danych na *urządzeniu dyskowym*, takim jak dysk lub nośnik optyczny. Używając bloków o dużym rozmiarze, po prostu poprawia się transfer danych. Jednak w przypadku zapisywania danych na *urządzeniu taśmowym* każdy blok staje się *rekordem*. Każdy rekord jest rozdzielony przerwą międzyrekordową. Gdy taśmę zapisze się z zastosowaniem bloku o określonym rozmiarze, musi zostać odczytana z uwzględnieniem takiego rozmiaru bloku lub jego wielokrotności (jeśli na przykład na taśmie zapisze się dane z wykorzystaniem bloku o rozmiarze 1024, podczas odczytu trzeba będzie użyć identycznego rozmiaru lub wielokrotności liczby 1024, czyli wartości 2048 lub 10 240). Dotyczy to jedynie urządzeń taśmowych, a nie dyskowych.

Niezależne określanie rozmiaru bloków wejściowych i wyjściowych

Ustalając rozmiar bloku za pomocą opcji `bs=`, określa się zarówno rozmiar bloku wejściowego, jak i wyjściowego. Czasami może być konieczne zastosowanie różnych rozmiarów dla tych dwóch typów bloków. Umożliwiają to opcje `ibs=` i `obs=`. Przykładowo, aby odczytać taśmę

[8] Oczywiście napęd taśmowy jest kolejnym niesformatowanym urządzeniem.

z użyciem jednego rozmiaru bloku i zapisać ją z użyciem innego, należy wykonać polecenie podobne do następującego:

```
# dd if=/dev/rmt/0 ibs=10k of=/dev/rmt/1 obs=64k
```

Określanie liczby odczytywanych rekordów

Opcja `count=n` informuje program *dd* o liczbie odczytywanych rekordów (bloków). Za jej pomocą można na przykład odczytać kilka pierwszych bloków pliku lub taśmy, aby sprawdzić, jakiego typu dane przechowują (więcej informacji zamieszczono poniżej). Korzystając z tej opcji, można również nakazać programowi *dd*, żeby podał, jaki rozmiar bloku zastosował podczas zapisywania taśmy.

Użycie programu dd do kopiowania zawartości pliku lub niesformatowanego urządzenia

Programu *dd* można użyć w roli narzędzia archiwizującego, ponieważ umożliwia kopiowanie w inne miejsce bitów pliku lub niesformatowanego urządzenia. Można nawet w ramach potoku przekazać programowi *compress* strumień bitów i uzyskać kopię danych w skompresowanej postaci (narzędzia: *dump*, *tar* i *cpio* nie oferują takiej możliwości, natomiast wersja GNU programu *tar* — tak). Najlepszym przykładem zastosowania programu *dd* w roli narzędzia archiwizującego jest skrypt *oraback.sh* (więcej na jego temat zamieszczono w rozdziale 16.) bazy danych Oracle, wykonujący gorącą kopię zapasową. Ponieważ baza danych Oracle do przechowywania danych może używać zarówno niesformatowanych partycji, jak i plików, skrypt nie jest w stanie przewidzieć, które polecenie zastosować. Jednak program *dd* obsługuje niesformatowane partycje i pliki!

Użycie programu dd do konwertowania danych

Program *dd* może też posłużyć do konwersji danych z jednego formatu na drugi w ramach jednej operacji.

Konwertowanie danych przekazywanych innemu poleceniu

Operacja taka jest wykonywana przy użyciu bloków wejściowych i wyjściowych o różnych rozmiarach (za pomocą opcji `ibs=` i `obs=`). Jeśli polecenie takie jak `restore` jest w stanie odczytać bloki tylko o określonych rozmiarach i dysponuje się woluminem zapisanym z użyciem bloku o innym rozmiarze, wolumin można odczytać za pomocą programu *dd*, a następnie wyniki za pośrednictwem potoku przekazać narzędziu *restore*.

Konwertowanie danych mających nieprawidłowy format

Choć można pomyśleć, że program *dd* jest kopiarką bitów, potrafi też przetwarzać format danych. Program umożliwia przeprowadzenie konwersji między różnymi zestawami znaków, małymi i dużymi znakami, a także rekordami o stałej i zmiennej długości.

`conv=ascii`
: Konwertuje z formatu EBCDIC na format ASCII.

`conv=ebcdic`
: Konwertuje z formatu ASCII na format EBCDIC.

`conv=ibm`
: Konwertuje z formatu ASCII na format EBCDIC, używając tabeli konwersji IBM.

`conv=lcase`
: Mapuje znaki alfabetu US ASCII na ich odpowiedniki stosujące małe znaki.

`conv=ucase`
: Mapuje znaki alfabetu US ASCII na ich odpowiedniki stosujące duże znaki.

`conv=swab`
: Zamienia miejscami każdą parę bajtów. Opcja może być użyta do odczytu woluminu zapisanego z zastosowaniem innego uporządkowania bajtów.

`conv=noerror`
: Po wystąpieniu błędu nie przerywa przetwarzania.

`conv=sync`
: Uzupełnia każdy blok wejściowy o jego rozmiar.

`conv=notrunc`
: Nie obcina na wyjściu istniejącego pliku.

`conv=block`
: Zamienia rekord wejściowy na rekord o stałej długości określonej przez opcję `ibs`.

`conv=unblock`
: Konwertuje rekordy o stałej długości na rekordy o zmiennej długości.

`conv=..., ...`
: Używa wielu metod konwersji oddzielonych od siebie przecinkami.

Zastosowanie programu dd do określenia rozmiaru bloku na taśmie

Operacja ta jest pewnego rodzaju sztuczką. Jeśli nakaże się programowi *dd* odczytanie jednego bloku danych, a następnie zapisanie go na dysku, można będzie przyjrzeć się rozmiarowi tego bloku, aby stwierdzić, jaki jest rozmiar bloku taśmy. Ponieważ nie jest znany rozmiar bloku, należy najpierw dla posiadanego urządzenia zastosować największy rozmiar bloku obsługiwany przez system operacyjny, który zwykle wynosi 128 lub 256 kB (może być większy).

```
# dd if=urządzenie bs=128k of=/tmp/junk count=1
```

Polecenie nakazuje programowi *dd* odczytywanie danych z zastosowaniem bloku o rozmiarze 128 kB do momentu osiągnięcia pierwszej przerwy międzyrekordowej. Jeżeli rozmiar bloku jest mniejszy niż 128 kB, odczyt danych zakończy się przy przerwie. Gdy rozmiar bloku przekracza 128 kB, program *dd* interpretuje to jako błąd wejścia-wyjścia i generuje komunikat. Wystarczy zwiększyć rozmiar bloku i spróbować ponownie (warto ustawić tym razem rozmiar 256 kB). Proces tworzy plik */tmp/junk*, którego wielkość jest równa rozmiarowi bloku na taśmie!

Użycie programu dd do identyfikacji formatu kopii zapasowej

To jeszcze jedna sztuczka. W celu utworzenia pliku */tmp/junk* należy zastosować to samo polecenie co w poprzednim punkcie, a następnie wykonać następujące polecenie:

```
# file /tmp/junk
```

Polecenie używa pliku */etc/magic*, aby określić typ pliku. Jeśli plik jest archiwum programu *tar* lub *cpio*, zwykle uzyska się odpowiednią informację. Jeżeli polecenie nie jest w stanie odgadnąć typu pliku, po prostu zwraca łańcuch `data` (dane), który nie jest zbytnio przydatny.

Innym interesującym zastosowaniem programu *dd* jest użycie go z narzędziem *ssh* lub *rsh*. Trzeba zapoznać się z podrozdziałem „Zastosowanie programu ssh lub rsh w roli kanału między systemami" zamieszczonym na końcu rozdziału.

Zastosowanie narzędzia rsync

O narzędziu *rsync* należy myśleć jak o prostym poleceniu kopiującym dane między systemami. Choć pod względem składni narzędzie to najbardziej przypomina program *rcp*, w pewnym stopniu można je też przyrównać do polecenia `copy` systemu Windows. Jednakże narzędzie *rsync* to coś znacznie więcej niż tylko prosty program kopiujący, ponieważ dodano do niego następujące funkcje:

Kopiowanie dowiązań, urządzeń, właścicieli, grup i uprawnień
: Oznacza to, że narzędzie *rsync* może poprawnie kopiować wszystko z miejsca źródłowego do docelowego, z uwzględnieniem specjalnych plików i wszystkich odpowiednich uprawnień. Narzędzie potrafi kopiować zarówno dowiązania twarde, jak i symboliczne.

Możliwość użycia dowolnej transparentnej zdalnej powłoki, w tym ssh lub rsh
: Choć obecnie mechanizm uwierzytelniania narzędziu *rsync* domyślnie zapewnia program *ssh*, z łatwością można to zmienić przez ustawienie dla zmiennej `RSYNC_RSH` wartości `rsh`.

Możliwość uruchamiania narzędzia jako uwierzytelnionego lub anonimowego demona (procesu działającego w tle)
: Oprócz tego, że jest uwierzytelniane za pośrednictwem programów *rsh* i *ssh*, narzędzie *rsync* może działać jako demon w trybie uwierzytelnionym lub anonimowym. Pierwszy tryb zapewnia bezpieczniejszy mechanizm uwierzytelniania, a drugi naprawdę dobrze sprawdza się w przypadku mirroringu.

Możliwość użycia zaawansowanych opcji wykluczających
: Narzędzie *rsync* wyklucza pliki w taki sam sposób jak wersja GNU programu *tar*, stosując w tym celu łańcuchy umieszczane w wierszu polecenia lub tworząc plik wykluczeń i podając jego nazwę obok opcji `exclude-from`. Dodatkowo narzędzie *rsync* może zostać skonfigurowane tak, aby pomijać te pliki, które zostałyby zignorowane przez system kontroli wersji CVS (*Concurrent Versions System*).

Wysyłanie tylko zmienionych bloków zmodyfikowanych plików
: Jest to największa różnica między narzędziami *rsync* i *rcp*, a także najwspanialsza funkcja pierwszego z nich. Wiele osób nie wie o jej istnieniu. Podczas aktualizowania danych docelowych lokalizacje źródłowe i docelowe dzielą każdy zmieniony plik na bloki i dla każdego z nich przeprowadzają dwa sprawdzenia sum kontrolnych CRC. Przesyłane są tylko te bloki danych, w przypadku których nie są zgodne sumy kontrolne CRC. W ten sposób program *rsync* może synchronizować często modyfikowane duże pliki za pośrednictwem znacznie mniejszych potoków.

Wysyłanie kilku zmienionych plików w postaci jednego dużego pliku
: Ponieważ narzędzie *rsync* przeprowadza mnóstwo operacji na pojedynczych plikach w celu skrócenia czasu oczekiwania, może je skumulować w ramach jednego dużego transferu.

Możliwość usuwania plików
: Jest to kolejna znacząca różnica między narzędziami *rcp* i *rsync*. Program *rsync* potrafi usuwać pliki z lokalizacji docelowej, których nie ma już w miejscu źródłowym.

Wiele osób, włącznie ze mną, tak naprawdę nie pomyślało o programie *rsync* jak o narzędziu archiwizującym. Jeden z powodów jest taki, że w rzeczywistości jest to narzędzie synchronizujące, a nie archiwizujące. Oznacza to, że bez określonego rodzaju interwencji użytkownika każde kolejne uruchomienie narzędzia *rsync* spowoduje nadpisanie kopii zapasowej niepoprawną kopią oryginalnych danych lub usunięcie z kopii zapasowej pliku, który usunięto z oryginału. Czyż nie wydaje się, że nie jest to zbyt dobre narzędzie archiwizujące?

Jednak nie trzeba włożyć zbyt wiele pracy, żeby narzędzie *rsync* zyskało na wartości. Jeśli zapisze się poprzednie wersje danych przed nadpisaniem ich przez nowsze wersje lub usunięciem, narzędzie *rsync* może stać się znakomitym narzędziem archiwizującym. W książce zamieszczono dwa przykłady zastosowania tego programu. W rozdziale 5. omówiono narzędzie *BackupPC*, a w rozdziale 7. opisano niemal ciągłą ochronę danych przy wykorzystaniu programu *rsync* i powiązanych narzędzi.

Podstawowa składnia narzędzia rsync

Oto podstawowa składnia programu *rsync*:

```
% rsync miejsce_źródłowe [miejsce_źródłowe ...] miejsce_docelowe
```

Polecenie kopiuje jeden lub więcej plików lub katalogów źródłowych do docelowego katalogu znajdującego się na tym samym komputerze.

```
% rsync miejsce_źródłowe [miejsce_źródłowe ...]
    nazwa_użytkownika@nazwa_hosta:miejsce_docelowe
```

Polecenie kopiuje jeden lub więcej plików lub katalogów źródłowych do docelowego katalogu innego komputera. Uwierzytelnianie odbywa się za pomocą programu *rsh* lub *ssh*, gdy dla zmiennej `RSYNC_RSH` ustawiono wartość `ssh`.

```
% rsync miejsce_źródłowe [miejsce_źródłowe ...]
    nazwa_użytkownika@nazwa_hosta::miejsce_docelowe
```

Ponieważ najczęstszym zastosowaniem narzędzia *rsync* związanym z archiwizowaniem danych jest transferowanie całego drzewa katalogów między dwoma komputerami, warto przedstawić odpowiedni przykład. Chcemy przenieść katalog */home* do katalogu */kopie_zapasowe* komputera *serwer_kopii_zapasowych*. Ponadto zależy nam na zarchiwizowaniu wszystkiego, co znajduje się w katalogu */home* (należy użyć opcji rekursywności `-r`), sporządzeniu kopii zapasowej dowiązań symbolicznych (opcja `-l`), zachowaniu ich czasów (opcja `-t`), uprawnień (opcja `-p`), a także prawa własności (opcja `-o`) i uprawnień grupowych (opcja `-g`). Dodatkowo chcemy przenieść wszystkie pliki specjalne (opcja `-D`). Polecenie będzie wyglądać podobnie do poniższego.

```
% rsync -rlptgoD /home serwer_kopii_zapasowych:/kopie_zapasowe
```

Na szczęście twórcy narzędzia *rsync* zdali sobie sprawę z tego, że wymienione opcje bardzo często stosowano do celów archiwizacyjnych. W związku z tym utworzyli opcję `-a`, która odpowiada zestawowi opcji `-rlptgoD`. W efekcie poniższe proste polecenie ma takie samo działanie jak poprzednie.

```
% rsync -a /home serwer_kopii_zapasowych:/kopie_zapasowe
```

Po dodaniu do polecenia opcji szczegółowych informacji (`-v`) i kompresji (`-z`) ma ono następującą postać:

```
% rsync -avz /home serwer_kopii_zapasowych:/kopie_zapasowe
```

Aby synchronizacja rzeczywiście miała miejsce, do polecenia trzeba dodać znacznik usuwania.

```
% rsync -avz --delete /home serwer_kopii_zapasowych:/kopie_zapasowe
```

Przy każdym uruchomieniu programu *rsync* zostanie skopiowana cała zawartość katalogu */home* do katalogu */kopie_zapasowe/home*, z którego zostaną usunięte wszystkie pliki nieobecne w katalogu */home*. Trzeba jedynie po stronie docelowej zastosować jakiś mechanizm gromadzenia danych historycznych i uzyska się system archiwizowania!

Czytelnik musi przeczytać rozdział 7., poświęcony systemom open source niemal ciągłej ochrony danych, a także rozdział 5., w którym omówiono narzędzie *BackupPC* i dokładniej wyjaśniono, jak zastosować program *rsync* do archiwizowania danych.

Kilka modyfikacji

Wszystkie powyższe polecenia kopiują zawartość katalogu */home* do katalogu */kopie_zapasowe* komputera *serwer_kopii_zapasowych*. Oznacza to, że tworzą katalog */kopie_zapasowe/home*. Jeśli zamierza się skopiować zawartość katalogu */home* do katalogu */kopie_zapasowe* bez tworzenia podkatalogu */home*, wystarczy na końcu nazwy źródłowego katalogu wstawić znak `/`.

```
% rsync -avz /home/ serwer_kopii_zapasowych:/kopie_zapasowe
```

Polecenie to wykonuje to samo zadanie co poniższe, tylko wymaga mniejszej liczby wciśnięć klawiszy.

```
% rsync -avz /home serwer_kopii_zapasowych:/kopie_zapasowe/home
```

Domyślnie program *rsync* przeprowadza proces uwierzytelniania za pomocą narzędzia *ssh*. Proces może być realizowany przy użyciu program *rsh*. W tym celu dla zmiennej `RSYNC_RSH` trzeba ustawić wartość `rsh`. Ponadto można również nakazać narzędziu *rsync*, aby nawiązało połączenie z demonem *rsync* uruchomionym na innym komputerze. Robi się to przez wstawienie za nazwą hosta dwóch dwukropków zamiast jednego.

```
% rsync -avz /home/ serwer_kopii_zapasowych::/kopie_zapasowe
```

Jeśli demon *rsync*, z którym nawiązuje się połączenie, wymaga hasła, można je określić za pomocą zmiennej `RSYNC_PASSWORD`.

Narzędzie rsync w systemie Windows

Choć narzędzie *rsync* jest programem uniksowym, można je uruchomić pod systemem Windows, jeśli zastosuje się emulator systemu Unix, taki jak *cygwin*. Część zespołu rozwijającego narzędzie *rsync* wykonała ciężką pracę i stworzyła prekompilowane binaria uwzględniające pliki *cygwin1.dll* i *rsync.exe*. Instrukcje dotyczące uruchamiania narzędzia *rsync* pod systemem Windows, z uwzględnieniem działania w trybie usługi (demona), można znaleźć na głównej stronie internetowej programu, znajdującej się pod adresem *http://samba.org/rsync/nt.html*.

Narzędzie rsync w systemie Mac OS

Użycie narzędzia *rsync* w systemie Mac OS jest dość proste. Trzeba jedynie dodać opcję `-E` lub `-extended-attributes`, które nakazują systemowi przeniesienie dodatkowych atrybutów plików. Opcja `-E` po prostu instruuje system, aby dokonał transferu rozwidleń zasobów (dziwić może jedynie to, że opcja ta została dodana do narzędzia *rsync*, które przewidziano do przenoszenia bitu wykonywania transferowanego pliku).

Odtwarzanie danych za pomocą narzędzia rsync

Odtwarzanie danych przy użyciu programu *rsync* wygląda tak samo jak archiwizowanie, z tym że w przypadku pierwszego procesu w wierszu polecenia zmienia się kolejność lokalizacji źródłowej i docelowej. Jako źródłową należy podać lokalizację, która normalnie jest docelową. Z kolei jako docelowe miejsce dla danych należy określić lokalizację, która standardowo jest źródłową. To wystarczy do uzyskania polecenia `rsync` przeprowadzającego proces odtwarzania danych. Użyjmy komputera z wcześniejszego przykładu i odwróćmy kolejność katalogów źródłowych i docelowych.

```
% rsync -avz serwer_kopii_zapasowych:/kopie_zapasowe/home/ /home
```

Polecenie nakazuje programowi *rsync* odtworzenie całej zawartości katalogu */kopie_zapasowe/home* komputera *serwer_kopii_zapasowych* i zapisanie jej w katalogu */home* lokalnego serwera. Oczywiście można też odtworzyć pojedynczy plik.

```
% rsync -avz serwer_kopii_zapasowych:/kopie_zapasowe/home/jnowak/podsumowanie.doc
    /home/jnowak
```

Prawdziwym wyzwaniem związanym z odtwarzaniem za pomocą programu *rsync* nie jest jego składnia, lecz monitorowanie tego, które pliki powinny zostać przywrócone i które pliki w rzeczywistości są uszkodzonymi kopiami, do pominięcia w procesie przywracania. Za coś takiego odpowiada używany program archiwizujący. Jeśli zastosowano by narzędzie wykonujące obrazy danych, takie jak omówione w książce, w celu uzyskania wczorajszej wersji pliku po prostu do łańcucha dodałoby się coś w rodzaju *dzienna.1*.

```
% rsync -avz
    serwer_kopii_zapasowych:/kopie_zapasowe/dzienna.1/home/jnowak/podsumowanie.doc
    /home/jnowak
```

Więcej na temat tworzenia obrazów danych za pomocą narzędzia *rsync* można znaleźć w rozdziale 7.

Archiwizowanie i odtwarzanie danych przy użyciu narzędzia ditto

ditto jest narzędziem systemu Mac OS X służącym do rekursywnego kopiowania. Podobnie do narzędzia *tar* i *cpio* program ten może być użyty do tworzenia plików archiwum. Interesujące w przypadku programu *ditto* jest to, że jest jedynym wbudowanym narzędziem z możliwością tworzenia pełnych kopii zapasowych dla wszystkich wersji systemu Mac OS X. Jest tak od momentu przeniesienia narzędzia z systemu NEXT-STEP, obsługującego funkcje systemu plików HFS+, takie jak rozwidlenia zasobów (więcej na temat systemu plików HFS+ można znaleźć w punkcie „Zróżnicowanie systemów plików systemu Mac OS", zamieszczonym wcześniej w rozdziale).

Narzędzie *ditto* może kopiować pliki i katalogi do jednego z trzech typów lokalizacji docelowych: katalogu, pliku archiwum ZIP lub pliku archiwum programu *cpio*. Narzędzie *ditto* nie pozwala skopiować danych bezpośrednio na taśmę. Narzędzie to nie używa jeszcze jednego formatu archiwizowania. W związku z tym nie otrzyma się kopii zapasowych w formacie, którego po kilku latach nie da się odczytać.

Składnia narzędzia ditto w przypadku archiwizowania

Narzędzie *ditto* najczęściej jest używane do rekursywnego kopiowania plików i katalogów.

```
$ ditto -V --rsrc lokalizacja_źródłowa... katalog_docelowy
```

Opcja `-V` powoduje wyświetlanie wszystkiego, co jest kopiowane przez narzędzie *ditto*. Opcja `--rsrc` zapewnia, że są kopiowane atrybuty i rozwidlenia zasobów systemu plików HFS+ (jest to domyślne działanie począwszy od wersji 10.4 systemu Mac OS X). Dodatkowe informacje dotyczące systemu plików HFS+ są przechowywane w formacie AppleDouble, w przypadku którego dane dla pliku *nazwa_pliku* znajdują się w pliku *._nazwa_pliku*.

Użycie narzędzia *ditto* w taki sposób bardzo przypomina zastosowanie polecenia `cp -R`, z jedną znaczącą różnicą. Załóżmy, że ktoś zamierza sporządzić kopię katalogu. Gdyby wykonał polecenie `cp -R źródłowy_katalog docelowy_katalog`, zawartość katalogu *źródłowy_katalog* znalazłaby się w katalogu *docelowy_katalog/źródłowy_katalog/*. W przypadku polecenia `ditto źródłowy_katalog docelowy_katalog` zawartość katalogu *źródłowy_katalog* trafiłaby bezpośrednio do katalogu *docelowy_katalog/* (może to być trochę zaskakujące, jeśli się tego nie oczekuje). Ponadto narzędzie *ditto* tworzy katalog *docelowy_katalog/*, jeżeli jeszcze nie istnieje.

W większości przypadków program *ditto* sporządza dokładną kopię danych źródłowych. Jednak jest kilka rzeczy, których narzędzie *ditto* nie skopiuje. Wiąże się to z brakiem następujących informacji:

- Nazwane gniazda. Należy zapoznać się z plikami dokumentacji *socket(2)* i *bind(2)* (okazuje się, że z jakiegoś powodu nie istnieją w wersji 10.4 systemu Mac OS X, choć występowały w starszych wersjach). Jednakże gniazda powinny być tworzone dynamicznie przez używające ich programy.
- Nazwane potoki (lub kolejki FIFO). Należy zapoznać się z plikiem dokumentacji *mkfifo*. Na szczęście sam system Mac OS X nie stosuje żadnych nazwanych potoków.
- Znaczniki BSD. Należy przeczytać plik dokumentacji *chflags*. System Mac OS X nie ustawia tych znaczników dla żadnego pliku.
- Rozszerzone listy kontroli dostępu ACL. Należy zapoznać się z plikami dokumentacji *chmod* i *fsaclctl*. Domyślnie systemy plików nie mają włączonej funkcji obsługi rozszerzonych list kontroli dostępu ACL.

Są to oczywiste ograniczenia mechanizmu binarnych plików BOM (*Bill-of-Materials*) używanego przez narzędzie *ditto* (należy przeczytać pliki dokumentacji: *bom*, *mkbom* i *lsbom*). Program *mkbom* również nie stosuje nazwanych gniazd lub potoków, a format plików BOM nie uwzględnia pól dla znaczników BSD lub rozszerzonych list ACL.

Oprócz wykonywania zwykłych kopii plików i katalogów program *ditto* może skopiować je do pliku archiwum. W celu utworzenia pliku archiwum narzędzia *cpio* (z opcjonalną kompresją obsługiwaną przez program *gzip*) należy zastosować następujące polecenie:

```
$ ditto -V --rsrc -c -z katalog_źródłowy docelowy.cpgz
```

Aby utworzyć plik ZIP, należy wykonać poniższe polecenie.

```
$ ditto -V --rsrc -c -k katalog_źródłowy docelowy.zip
```

Gdy podczas tworzenia pliku ZIP użyje się znacznika `--sequesterRsrc`, w katalogu o nazwie *__MACOSX* zostaną zapisane specjalne dane dotyczące systemu plików HFS+. Narzędzie zgodne z programem *PKZIP* (inne niż *ditto*) mogą radzić sobie z tym lepiej niż format AppleDouble.

Podczas tworzenia rekursywnych kopii łańcuch *katalog_źródłowy* jest usuwany ze ścieżek zapisywanych w pliku archiwum. Aby w ścieżkach zachować ten łańcuch, należy użyć znacznika `--keepParent`.

Narzędzie *ditto* nie pozwala na jedno: selektywne archiwizowanie tylko części zawartości katalogu. Przykładowo nie można zastosować wzorca nazwy pliku lub sporządzać przyrostowych kopii zapasowych. *ditto* nadaje się wyłącznie do archiwizowania całych drzew katalogowych.

Za pomocą narzędzi *ssh* i *dd* można wykonywać kopie zapasowe zdalnych systemów w taki sam sposób jak w przypadku programów *tar* i *cpio*. Oto przykład:

```
$ ditto -V --rsrc -c -k źródłowy_katalog - | (sh zdalny_host dd of=docelowy.zip)
```

Warto zauważyć, że w przykładzie narzędzie *ditto* może przesłać plik archiwum do standardowego wyjścia, a także jako źródło zaakceptować standardowe wejście. Prawdopodobnie funkcjonalność taka może być wykorzystana na potrzeby archiwizowania i odtwarzania danych z taśmy (jeśli są dostępne odpowiednie sterowniki napędu taśmowego). Jednak nie było to testowane.

Opcje polecenia ditto

ditto jest bardzo prostym poleceniem o stosunkowo niewielkiej liczbie opcji i zrozumiałej składni argumentów. Oto część opcji, z których można skorzystać (więcej informacji można znaleźć w dokumentacji):

`-v`
: Wyświetla nazwę każdego kopiowanego źródłowego katalogu.

`-V`
: Wyświetla wiersz dla każdego pliku i katalogu kopiowanego przez narzędzie *ditto*.

`-c`
: Zamiast do innego katalogu zawartość źródłowego katalogu jest kopiowana do pliku archiwum. Jeśli nie zastosuje się opcji `-k`, domyślnie jest to plik archiwum narzędzia *cpio*.

`-z`
: Za pomocą programu *gzip* jest kompresowany plik archiwum narzędzia *cpio*.

`-k`
: Zamiast pliku archiwum programu *cpio* jest tworzone skompresowane archiwum ZIP.

`-X`
: Zapobiega w czasie kopiowania przekroczeniu przez narzędzie *ditto* granicy partycji.

`--keepParent`
: Uwzględnia źródłowy katalog w ścieżkach zapisywanych w pliku archiwum.

`--rsrc`
: Oprócz standardowych uniksowych atrybutów i rozwidleń danych kopiuje atrybuty i rozwidlenia zasobów systemu plików HFS+. Jest to domyślne działanie w przypadku wersji 10.4 systemu Mac OS X i nowszych. Zamiennie można też zastosować opcję `--rsrcFork`.

`--norsrc`
: Zapobiega kopiowaniu przez narzędzie *ditto* atrybutów i rozwidleń zasobów systemu plików HFS+. Jest to domyślne działanie w przypadku wersji 10.3 systemu Mac OS X i starszych bądź wersji 10.4 i nowszych, gdy zostanie ustawiona zmienna `DITTONORSRC`.

`--sequesterRsrc`
: Zapisuje rozwidlenia danych i zasobów systemu plików HFS+ w katalogu o nazwie *__MACOSX* zamiast przy użyciu formatu AppleDouble.

`--arch`
: W przypadku wykonywania kopii aplikacji obsługującej architektury wieloprocesorowe (dawniej aplikacje takie nazywano grubymi binariami; obecnie firma Apple określa je mianem uniwersalnych) kopiuje tylko elementy określonej architektury. Architekturą może być `ppc` (procesory PowerPC) lub `i386` (procesory Intel; odniesienie do pierwszego układu Intela obsługiwanego przez system NEXTSTEP).

`--bom`
: Kopiuje jedynie pozycje wyszczególnione w konkretnym pliku BOM (można go utworzyć za pomocą polecenia `mkbom` *`katalog`*; więcej informacji można znaleźć w dokumentacji polecenia). Pliki BOM są używane w przypadku pakietów instalatora firmy Apple, a także uprawnień, praw własności i sumy kontrolnej rekordów dla każdej pozycji instalowanej przez pakiet.

Składnia narzędzia ditto w przypadku odtwarzania

Zawartość pliku archiwum utworzonego przy użyciu narzędzia *ditto* jest odtwarzana za pomocą opcji `-x` (od słowa *extract* — wyodrębnianie). W celu odtworzenia danych ze skompresowanego archiwum programu *cpio* należy wykonać następujące polecenie:

```
$ ditto -V --rsrc -x źródłowy.cpgz docelowy_katalog
```

Katalog docelowy jest tworzony, jeśli jeszcze nie istnieje. Warto zauważyć, że opcja `-z` nie jest wymagana. Narzędzie *ditto* automatycznie obsługuje skompresowane pliki programu *cpio*.

Aby odtworzyć dane z archiwum ZIP, należy zastosować poniższe polecenie.

```
$ ditto -V --rsrc -x -k źródłowy.zip docelowy_katalog
```

Nie ma nic specjalnego w plikach archiwum tworzonych przez narzędzie *ditto*. Za jego pomocą można wyodrębnić dane z każdego pliku archiwum ZIP lub programu *cpio*.

Jeśli zamierza się odtworzyć tylko wybrane pozycje pliku archiwum, należy użyć bezpośrednio narzędzia *cpio* lub *unzip*, ponieważ nie jest możliwe wykonanie tej operacji za pomocą programu *ditto*.

Wyświetlanie listy plików archiwum

W celu wygenerowania listy plików znajdujących się w skompresowanym archiwum *cpio* należy wykonać następujące polecenie:

```
$ cpio -itvz < zrodlowe.cpgz
```

Aby wyświetlić pliki zawarte w archiwum ZIP, należy zastosować polecenie:

```
$ unzip -lv zrodlo.zip
```

Porównanie narzędzi: tar, cpio i dump

Kilka lat temu John Pezzano z firmy Hewlett-Packard sporządził dokument porównujący wbudowane produkty archiwizujące. Ponieważ to najlepsze porównanie, z jakim do tej pory się spotkałem, poprosiłem go o zgodę na nieznaczną aktualizację dokumentu w celu uwzględnienia zmian, które zaszły w narzędziach, a także na zamieszczenie go w tej książce. W tabeli 3.3 porównano narzędzia: *tar*, *cpio* i *dump*.

Tabela 3.3. Porównanie wbudowanych narzędzi

Cecha	tar	cpio	dump
Łatwość uruchamiania	Bardzo duża (polecenie `tar c pliki`).	Wymaga użycia programu *find* do określenia nazw plików.	Duża — niewiele opcji.
Radzenie sobie z błędami wejścia-wyjścia	Brak własnego rozwiązania. Trzeba samemu napisać skrypt.	W przypadku systemu HP-UX opcja `resync` powoduje utratę części danych.	Automatycznie są pomijane niepoprawne sekcje.
Archiwizowanie specjalnych plików	Późniejsze wersje.	Tak.	Tak.
Archiwizowanie na wielu woluminach	Późniejsze wersje.	Tak.	Tak.
Archiwizowanie za pośrednictwem sieci	Zastosowanie wyłącznie narzędzia *rsh* lub *ssh*.	Zastosowanie wyłącznie narzędzia *rsh* lub *ssh*.	Tak.
Dołączanie plików do kopii zapasowej	Tak (polecenie `tar -r`).	Nie.	Nie.
Wiele niezależnych kopii zapasowych na jednej taśmie	Tak.	Tak.	Tak.
Łatwość generowania listy plików woluminu	Niewielka. Trzeba przeszukiwać całą kopię zapasową (polecenie `tar -t`).	Niewielka. Trzeba przeszukiwać całą kopię zapasową (polecenie `cpio -it`).	Duża — na początku woluminu znajduje się indeks (polecenie `restore -t`).
Łatwość i szybkość szukania określonego pliku	Niewielka. Brak znaków wieloznacznych. Trzeba przeszukiwać cały wolumin.	Średnia. Dostępne znaki wieloznaczne. Trzeba przeszukiwać cały wolumin.	Interaktywność. Bardzo duża łatwość użycia w przypadku takich poleceń, jak `cd` i `ls`.
Przyrostowe kopie zapasowe	W przypadku stosowania wersji GNU narzędzia *tar* można użyć opcji `-newer` programu *find*.	Trzeba użyć programu *find*, aby zlokalizować nowe i zmodyfikowane pliki.	Możliwa przyrostowa kopia zapasowa wyłącznie całego systemu plików. Wiele poziomów.
Tworzenie listy plików podczas archiwizowania	`tar cvf 2>plik_dziennika`	`cpio -v 2>plik_dziennika`	Tylko po zarchiwizowaniu danych za pomocą polecenia `restore -t >plik_dziennika` (jednak program *dump* może pokazać w procentach postęp procesu).
Archiwizowanie oparte na innych kryteriach	Tak (za pomocą wersji GNU narzędzia *tar*).	Program *find* może użyć wielu kryteriów.	Nie.

Tabela 3.3. Porównanie wbudowanych narzędzi — ciąg dalszy

Cecha	tar	cpio	dump
Przywracanie bezwzględnych ścieżek do postaci względnych	Tak (za pomocą wersji GNU narzędzia *tar*).	Za pomocą polecenia `cpio -I` lub wersji GNU narzędzia *cpio*.	Ścieżki zawsze są względne w stosunku do bieżącego katalogu roboczego.
Możliwość interaktywności w czasie odtwarzania danych	Tak. Brak możliwości w przypadku polecenia `tar -w`.	Dla każdego pliku można określić nową ścieżkę lub nazwę.	W trybie interaktywnym można wybrać poszczególne pliki.
Zgodność	Wiele platform.	Wiele platform z nagłówkiem ASCII, lecz nie zawsze możliwość przeniesienia.	Możliwość odczytu między kilkoma platformami. Jednak nie zawsze można na to liczyć.
Podstawowe zastosowanie	Archiwizowanie systemu, gdy stosuje się wersję GNU narzędzia *tar*. W przeciwnym razie archiwizowanie danych poszczególnych użytkowników i przenoszenie plików między systemami plików.	Archiwizowanie systemu i przenoszenie plików między systemami plików.	Archiwizowanie systemu.
Efektywność zapisu woluminu	Średnia. Zwykle występuje ograniczenie do bloków o rozmiarze 10 kB.	Średnia. Choć zazwyczaj występuje ograniczenie tylko do bloków o rozmiarze 5 kB, w przypadku niektórych systemów operacyjnych można określić większy rozmiar.	Wysoka. Zwykle można podać maksymalny rozmiar bloku obsługiwany przez urządzenie.
Stosowanie znaków wieloznacznych podczas procesu odtwarzania	Nie.	Tak.	Tylko w trybie interaktywnym.
Łatwość zaznaczania archiwizowanych plików znajdujących się w kilku katalogach	Niewielka. Trzeba określić niezależnie każdy katalog, włącznie z podkatalogami.	Średnia. Zastosowanie opcji programu *find*.	Brak możliwości. Program archiwizuje tylko i wyłącznie jeden system plików.
Możliwość odtworzenia plików zawartych w katalogu przez określenie tego katalogu	Tak.	Nie. Trzeba użyć *ścieżka/**.	Tak.
Zatrzymanie odczytu taśmy po znalezieniu odtwarzanego pliku	Nie.	Nie.	Przestaje czytać taśmę od razu po znalezieniu ostatniego pliku.
Śledzenie usuwanych plików	Nie.	Nie.	Jeśli odtwarza się dane z użyciem opcji `-r`, usuwane są pliki, które usunięto przed wykonaniem ostatniej przyrostowej kopii zapasowej.
Wydajność systemów plików	Lepsza.	Najgorsza (pliki uzyskują statystyki zarówno z programu *find*, jak i *cpio*).	Najlepsza.
Prawdopodobieństwo tego, że plik istnieje w tabeli zawartości, lecz nie w archiwum	Niskie.	Niskie.	Średnie (ponieważ jako pierwsza jest tworzona tabela zawartości).

Choć standardowe narzędzia archiwizujące mogą być niezbyt atrakcyjne, a nawet pozbawione wielu funkcji, po zapoznaniu się z nimi zawsze będzie można z nich skorzystać. Niektóre częściowo wbudowane polecenia (na przykład wersja GNU programów *tar* i *cpio*) również są bardzo przydatne, lecz nie zawsze dostępne. A zatem dobra praktyczna znajomość naprawdę wbudowanych narzędzi może okazać się bardzo przydatna w razie kłopotów lub gdy ktoś wręczy nieznany wolumin z pytaniem: „Czy można to odczytać?".

Zastosowanie programu ssh lub rsh w roli kanału między systemami

W niniejszym podrozdziale wyjaśniono, jak użyć narzędzia *rsh* lub *ssh* w roli kanału między systemami, zwłaszcza wtedy, gdy jednocześnie zastosuje się program *dd* i jakieś inne polecenia czytające lub zapisujące dane w standardowym wejściu `stdin`. Jeśli nawet narzędzie archiwizujące obsługuje zdalne urządzenia (na przykład *rdump*), zwykle korzysta z mechanizmu uwierzytelniania programu *rsh*. Jeżeli Czytelnik zrozumie zawartość podrozdziału, będzie mógł użyć narzędzia *ssh*, które zapewnia większe bezpieczeństwo kopii zapasowych.

Większość innych narzędzi archiwizujących potrafi jedynie odczytywać lub zapisywać dane w standardowym wejściu `stdin`, natomiast program *dd* może jednocześnie wykonywać obie operacje. To sprawia, że program *dd* jest bardzo uniwersalnym narzędziem archiwizującym (i jedynym wbudowanym), które może być użyte do przekazywania strumienia danych między dwoma poleceniami lub między komputerem i urządzeniem zdalnego hosta (przy wykorzystaniu narzędzia *rsh* lub *ssh*).

Odczytanie kopii zapasowej ze zdalnego urządzenia umożliwią narzędzia: *restore*, *tar* (wersja GNU) i *cpio* (wersja GNU), gdy w ich wierszu polecenia jako nazwę urządzenia po prostu wstawi się łańcuch *`zdalny_host`*`:`*`zdalne_urządzenie`*. Jednak wbudowane wersje narzędzi *tar* i *cpio* nie pozwalają na coś takiego. W ich przypadku wystarczy za pośrednictwem narzędzia *rsh* lub *ssh* wykonać polecenie `dd` na zdalnym komputerze i odczytać zwrócony strumień danych na lokalnym systemie.

```
# rsh zdalny_host "dd if=urządzenie ibs=rozmiar_bloku" | tar xvBf -
```

Trzeba pamiętać o tym, że podczas odczytu woluminu z taśmy za pomocą programu *dd* standardowo trzeba określić rozmiar bloku. Jeśli się tego nie zrobi, program zastosuje rozmiar bloku 512, który spowoduje wygenerowanie błędu wejścia-wyjścia, gdy woluminu na taśmie nie zapisano z wykorzystaniem bloku o takiej wielkości. Warto też zwrócić uwagę na znaki cudzysłowu obejmujące zdalne polecenie *dd*. W przypadku tego polecenia znaki te właściwie są zbędne, ponieważ potok jest przetwarzany na lokalnym systemie. W innych, bardziej złożonych poleceniach, w których potok będzie wykonywany na zdalnym systemie, umieszczenie zdalnego polecenia w znakach cudzysłowu zagwarantuje poprawne działanie (w powyższym poleceniu znaki cudzysłowu jedynie zwiększają czytelność).

Zapisywanie kopii zapasowej na zdalnym urządzeniu jest trochę bardziej złożone. Może być konieczne utworzenie powłoki podrzędnej[9], w której będą wykonywane polecenia `rsh` i `dd`,

[9] Nakład pracy będzie różny, ponieważ nie wszystkie wersje systemu Unix wymagają tworzenia powłoki podrzędnej.

a także przekazanie za pomocą potoku wyniku lokalnie uruchomionego programu archiwizującego.

```
# tar cvf - . \ | (rsh zdalny_host dd of=urządzenie obs=rozmiar_bloku)
```

Umieszczenie zdalnego polecenia w nawiasach okrągłych powoduje utworzenie powłoki podrzędnej. Warto zauważyć, że trzeba określić rozmiar zdalnego bloku i zachować przy tym ostrożność. Jeśli zamierza się utworzyć wolumin, który będzie mógł być odczytywany przez program *tar*, trzeba użyć rozmiaru bloku obsługiwanego przez ten program, takiego jak 10 240 (zwykle jest to największy rozmiar bloku, który może być odczytywany lub zapisywany przez narzędzie *tar*; aby go ustawić, należy określić współczynnik bloków o wartości 20).

Jeżeli nie jest możliwe skorzystanie z programu *rsh*, zamiast niego można użyć narzędzia *ssh*. Program ten stosuje znacznie bardziej bezpieczny mechanizm uwierzytelniania i umożliwia użycie tego samego typu poleceń co narzędzie *rsh* przy braku tworzonych przez nie luk w zabezpieczeniach. Jednak w przypadku zastosowania funkcji obsługi zdalnych urządzeń oferowanej przez wersję GNU narzędzi *tar* i *cpio*, a także przez program *dump*, domyślnie jest używany program *rsh*. Jeśli nie można skorzystać z narzędzia *rsh* i dostępne jest tylko *ssh*, w celu zintegrowania narzędzi: *dump*, *tar* i *cpio* z programem *ssh* można wykonać polecenia podobne do poniższych.

Aby odczytać taśmy zdalnych hostów, należy zastosować polecenia:

```
# ssh zdalny_host "dd if=urządzenie bs=rozmiar_bloku" | tar xvBf -
# ssh zdalny_host "dd if=urządzenie bs=rozmiar_bloku" \ | restore rvf -
# ssh zdalny_host "dd if=urządzenie bs=rozmiar_bloku" | cpio -itv
```

W celu utworzenia kopii zapasowych na taśmach zdalnych hostów należy wykonać polecenia:

```
# dump 0bdsf 64 100000 100000 - \ | ssh zdalny_host "dd if=urządzenie bs=64k"
# tar cvf - | ssh zdalny_host "dd if=urządzenie bs=10k"
# cpio -oacvB | ssh zdalny_host "dd if=urządzenie bs=5k"
```

Niektóre polecenia zadziałają z programem *ssh*, jeśli po prostu dla zmiennej środowiskowej `rsh` ustawi się ścieżkę `/usr/bin/ssh`.

Witryna BackupCentral.com oferuje dla każdego rozdziału książki stronę umożliwiającą internautom zamieszczanie własnych uwag i opinii. Pod adresem *http://www.backupcentral.com* można przeczytać aktualizowane informacje lub dodać do nich własne.