基礎知識がグングン身につく!
ステップアップ企画

システム管理 入門講座

—Stage 7—

初心者をその気にさせる

バックアップ&リストアの実践ノウハウ

「バックアップ」という作業は、地味で泥臭い作業に見えるかもしれない。バックアップについてあれこれ検討するよりも、新しいソフトウェアをインストールしたり、セキュリティの設定を行ったりするほうが、カッコイイ仕事をしているように思えることもあるだろう。しかし、システムに障害が発生したとき、そのシステムを救えるのはバックアップ・データだけなのだ。今回は、計画立案からツールの使い方まで、バックアップとリストアに関する知識・ノウハウをまとめて解説する。

Text by 佐藤竜一　Illustration by 内藤しなこ　Design by 坂内正景

Part 1 必要性を確認し、検討を行う

バックアップの5W1H

ユーザーの誤操作、ディスクのクラッシュ、クラッカーによる侵入、オフィスの火災……どんなに高機能／高性能なサーバ・マシンでも、データが喪失する危険と無縁ではない。まずは、バックアップの必要性と導入時に検討すべきポイントを押さえよう。

バックアップの目的

サーバに限らず、コンピュータはいつか必ず壊れるものだ。特に、データを保存しているハードディスクという媒体は、現在のコンピュータ・システムの中で最も脆弱なコンポーネントと言ってよいだろう。

ハードディスクの信頼性を少しでも高めるべく、現在はRAID(*1)という技術が考案され広く利用されている。しかし、RAIDであっても壊れることはある。また、ハードディスクの前にRAIDコントローラ(*2)が壊れるという笑えない話もある。データのバックアップを行う目的は、このようなハードディスクの障害から貴重なデータを守ることにある。

それでは、仮に「絶対に壊れないハードディスク」があれば、バックアップは不要になるのだろうか。もちろん答えは「ノー」だ。ハードディスクが壊れなくても、一般ユーザーやシステム管理者の操作ミスによってデータが消えてしまうことがある。バックアップには、操作ミスによるデータ消失への対応という意図も含まれる。

今すぐバックアップの検討を

さて、皆さんのコンピュータのデータは、バックアップされているだろうか。また、取得したバックアップ・データの内容は、確実にリストアする（元に戻す）ことができるだろうか。

今日、サーバ・マシンの内部には、企業や組織の運営に欠かせない重要なデータが格納されている。よって、ひとたびデータが喪失すれば、その損害は計り知れない。もし、皆さんが管理しているサーバの中にバックアップを行っていないマシンがあるなら、明日からではなく、今すぐにバックアップの検討を始めるべきだ。

5W1Hを考える

バックアップを行う際には、そのタイミングや頻度など、運用上考慮すべき点がいくつかある。もちろん、その内容はバックアップの対象となるマシンによって異なる。そこでまずは、5W1Hについて検討してみるとよい。つまり、いつ（when）、どこへ（where）、誰が（who）、何を（what）、なぜ（why）、どのようにして（how）バックアップするかを考えてみるのだ。

なお、以下ではバックアップにおける5W1Hがどのようなものであるかを確認するが、具体的な計画立案の方法についてはパート2で説明する。

●when

「いつ（when）」は、すなわち「バックアップを行うタイミング」のことだ。バックアップを行うタイミングは、データの特性によって「毎日」「毎週」「毎月」など、さまざまなパターンが考えられる。場合によっては「数時間おき」に行う必要もあるだろう。

●where

「どこへ（where）」は、「バックアップ・メディア」を指す。バックアップ・メディアには、ハードディスク、CD-R、MO、磁気テープなど、さまざまな種類がある。データの特性やリカバリーの容易性などを考慮して適切な製品を選択する必要があるだろう。

なお、バックアップ・メディアをサーバ・ルームやマシンの横などに置きっぱなしにすると、紛失や損傷のおそれが生じる。バックアップ・メディアを選択する際には、その保管場所も検討しておくとよいだろう。

●who

「誰が（who）」は、「誰がバックアップを行うか」ではなく「バックアップを行うユーザー権限」を指す。rootならばどのようなデータでもバックアップを行える。しかし、一般ユーザーの権限でバックアップが行えるのであれば、そのほうが安全であることは言うまでもない（rootでの

操作ミスはシステム全体に致命的な影響を及ぼすことがある)。そこで、各データのアクセス権限を基に、どのデータをどのユーザー権限でバックアップするかを考えておくとよいだろう。

なお、人間は必ずミスを犯す。そこで、臨時や緊急時を除いて、バックアップは人間の手によってではなく自動化して行うことをお勧めする。自動化といっても難しく考える必要はなく、例えば、バックアップ用のコマンドをcronに登録すれば、安全かつ確実にバックアップを行うことが可能になる。

●what

「何を(what)」は、文字どおり「何をバックアップするか」を指す。コンピュータ内の各データを、バックアップが必要なものと不要なものとに切り分けていく。

●why

「なぜ(why)」は、「どうしてそのデータのバックアップが必要なのか」を考えることだ。個々のデータについて「どのような理由からバックアップが必要なのか」を考えていけば、バックアップの頻度、方法、バックアップ・メディアの選択などに関するヒントが得られるはずだ。

●how

「どのようにして(how)」は、「バックアップの方法」を指す。どのようなツールを利用してバックアップを取るかに加え、どのようにしてリストアを行えばよいかについても検討することが必要だ。リストアできないバックアップ・データには何の意味もない。

保存期間にも注意が必要

5W1Hを検討したら(具体的な検討方法はパート2以降を参照)、最後にもう1つの「H」について考えてみよう。それは、個々のバックアップ・データを「どの程度の期間(how long)」保存するかだ。

例えば、毎晩バックアップするデータがあるとする。そのバックアップ・データは、最新の1日分だけがあればよいのか、それとも数日分、数週間分の保存が必要なのか。

データの保存期間は、サーバの運用目的やサーバに保持されているデータの性質によって決まる。そのため、このような疑問に対する一般的な解答はない。それに答えられるのは、そのサーバにかかわっているシステム管理者、あるいはユーザーだけだ。ユーザーから「何月何日のこのデータを復旧してもらいたいんだけど」と言われたときに焦らないよう、データの保存期間についても十分に検討しておくようにしよう。

*1:RAIDは、Redundant Arrays of Inexpensive Disksの略。複数のハードディスクを接続し、信頼性の高い記憶装置を構築する技術。
*2:RAIDコントローラは、RAIDによって接続した複数台のディスクを1つのディスクに見せかけるための制御を行う装置。

Column

バックアップに関する3つの注意点

①フル・バックアップはシステムの未使用時に実行

フル・バックアップは、リストアの基礎となる大切なデータであるため、シングルユーザー・モード(rootのみがシステムにログインできるメンテナンス・モード)で行うのがよいとされている。というのも、バックアップ中に誰かがファイルの内容を変更したり、移動や削除を行ったりすると、そのファイルが正常にバックアップされない可能性があるからだ。

しかし、実際の運用時には誰かがシステムを利用しているため、シングルユーザー・モードに移行するのは難しいかもしれない。そのような場合は、夜間などなるべくユーザーがログインしていない時間帯を狙ってバックアップを行うとよい。

②圧縮はできるだけ行わない

tarアーカイブを作成するとき、習慣的にアーカイブをgzipで圧縮する方は少なくないだろう。しかし、バックアップを取るときはできるだけアーカイブを圧縮しないようにするべきだ。というのも、圧縮されたアーカイブが1バイトでも損傷すると、そのアーカイブは二度と展開できなくなり、すべてのファイルが取り戻せなくなるからだ。

どうしても圧縮を行う必要があるときは、アーカイブをなるべく小さな単位に分けて作成し、個別に圧縮するとよい。これは少々面倒な作業ではあるが、問題発生時にバックアップ・データが全滅することを思えば、さほど苦にはならないだろう。

③アーカイブを絶対パスで指定しない

cpioやtarといったコマンドでバックアップを行うときは、「アーカイブの対象とするファイルやディレクトリを絶対パス(フル・パス)で指定しない」ように注意する。コマンドの実行時に絶対パスを指定してしまうと、そのパスにしかリストアできなくなってしまうからだ。

例えば、/homeをバックアップするとき、/homeという絶対パスを指定してアーカイブを作成すると、そのアーカイブ内には/homeというパスが記録される。そのため、そのアーカイブは/homeというディレクトリにしかリストアできない。

一方、バックアップを行う前に/ディレクトリに移動して、アーカイブを./homeといった具合に相対パスで指定して作成すると、アーカイブ中には./homeというパスが記録される。こうしておくと、例えば/tmpにバックアップ・データをリストアしたければ/tmp/homeと、/varにリストアしたければ/var/homeと指定できるようになる。つまり、リストア時にリストア場所を選べるようになるわけだ。

Part 2 対象データとスケジュールを決める

バックアップの計画立案

バックアップとリストアにはそれ相応の手間と時間がかかる。そのため、むやみに何もかもバックアップすればよいというわけではない。バックアップを実行する前には入念に計画を立て、無駄なく実行するように心がけよう。

バックアップ・データの選択

パート1で紹介した5W1Hのうち、最初に検討すべきは、what（何をバックアップすべきか）だろう。もちろん、すべてのデータをバックアップできればそれにこしたことはない。しかし、利用できるバックアップ・メディアの容量が足りなくなる場合や、バックアップにかかる時間を1分1秒でも短縮しなければならない場合などは、バックアップ対象の取捨選択が必要になる。

以下、具体的な取捨選択方法を説明しよう。

対象外としてもよいデータ

まず、真っ先にバックアップの対象から除外してかまわないのは、「一時データ」（一時的に利用するファイル）のたぐいだ。/tmp(*1)やプロキシ・サーバ(*2)のキャッシュのように、一時的にしか利用しないデータを格納するディレクトリやファイルがこれに該当する。また、スワップ領域(*3)や/procファイルシステム(*4)も、バックアップする必要はない。

次に、「別のマシンに同じデータが存在するデータ」を除外する。このようなデータは、喪失しても別のマシンからコピーすることで復旧できるからだ。例えば、負荷分散のために複数のWebサーバを立ち上げているとする。その場合、全サーバのコンテンツの内容が同一であれば、そのうちの1台のデータだけをバックアップの対象とすればよいだろう。

さらに、OSもバックアップ対象から除外しても問題はない。OSは、再インストールすることによって復旧させられるからだ。また、自分でコンパイルしてインストールしたソフトウェアも、少々手間はかかるが再インストールによって復旧させることができる（Column）。

対象とすべきデータ

次に、必ずバックアップすべきデータを確認しよう。それは「同じものを作成できないデータ」と考えればよい。例えば、ユーザーが自分で作成したデータや、顧客が入力したデータなどだ。より具体的

*1：/tmpは、データを一時的に保存するためのディレクトリ。通常、/tmpのデータはLinuxの再起動時に削除される。
*2：プロキシ・サーバは、内部ネットワークとインターネットとの間に設置し、内部ネットワークにあるマシンの代わりに外部とのやり取りを行うサーバ。内部ネットワークからのアクセス頻度が高いWebデータをキャッシュ（格納）して、Webデータへのアクセス速度を高速化する機能などを備える。
*3：スワップ領域は、ディスク内に作成される仮想的なメモリ領域。メモリの利用率が高くなるとOSはスワップを行い、ディスクの仮想メモリ領域を利用する。
*4：/procファイルシステムとは、カーネルが保持している情報などを仮想的なファイルシステム内のファイルに格納し、ユーザーが参照できるようにするための仕組み。例えば、「/proc/memstat」というファイルには、現在のメモリ情報が格納されており、ユーザーはそれを参照できるようになっている。

Column

OSとソフトウェアのバックアップ

ここでは、OSが格納されているディスクと、データが格納されているディスクが分かれているケースを前提に、OSとソフトウェアのバックアップについて考えてみる。

OSが格納されているディスクが壊れたときは、そのディスクを交換してOSを再インストールすれば問題は解決する。Linuxの場合、雑誌の付録などからディストリビューションを入手できる。そのため、OSのデータを復旧するコストは非常に安い。そこで、OSが入っているディスクはバックアップ対象外とし、データが入っているディスクだけをバックアップ対象としてもよいだろう。

しかし、実際にOSを再インストールして再稼働させるにはそれなりの時間がかかる。しかも、再インストール後には各種のパッチを当てる必要がある。そこで、OSが入っているディスクもバックアップしておけば、OSの復旧にかかる時間とコストは圧倒的に短縮されることになる。

また、自分でコンパイルしてインストールしたソフトウェアも、再コンパイルするにはそれなりの時間がかかる（システム管理者の人件費も発生する）。したがって、余裕があればバックアップしておくにこしたことはない。

このように、バックアップするデータを決める際は、「バックアップ元のメディアを容易に入手できるためバックアップ対象から除外する」あるいは「バックアップにかかる手間を軽減するためバックアップ対象に含める」といった具合に、システムの運用方針に応じて決めていくことになる。

には、ホーム・ディレクトリのデータ、ファイル・サーバの公開ディレクトリ、メール・サーバのメール・スプール、Webサーバの公開コンテンツ、データベース・サーバのデータベースなどだ。

また、各種プログラムの設定ファイルもバックアップ対象とすべきだろう。設定はやり直すこともできるが、そのためにかかる手間と時間は無視できない。

さらに、意外と忘れられがちなのだが、サーバ・マシンの場合はログのバックアップも取っておく必要がある。ログは、不正アクセスを受けたときの痕跡調査や、何らかの障害が発生したときの原因究明に欠かせない。また、クラッキング行為などで法的な問題が発生したときに、その証拠物件として提出する必要が生じることもある。

ログは、あとから再作成することはできない。したがって、バックアップの取得はもちろん、どの程度の期間保存するかについても十分な検討が必要になる。保存期間は一律には決められないが、通常、2カ月から3カ月程度保存しておけばよいだろう。

バックアップの頻度

さて、バックアップ対象とするデータが決まったら、次にバックアップの頻度を決める。

バックアップの運用で最もよくないのは、「何か大きな変更が発生したとき」や「気が向いたとき」などのみに行う不定期バックアップである。このような運用方法では、バックアップの実行を忘れてしまう可能性があるし、せっかく取っておいたバックアップ・データも、「いつのデータだったか忘れてしまった」などといった事態に陥りかねない。したがって、バックアップは必ず定期的かつ自動的に行うようにする。

バックアップの頻度は、データの性質にもよるため一概には言えないが、例えばファイル・サーバのデータであれば「1日1回」がよいだろう。もちろん、それ以上の頻度で行うことも可能だが、バックアップを実行するとシステムに負荷がかかる。そのため、あえて1日に何度も行う必要性がないのであれば、「1日1回」とすることをお勧めする。

一方、バックアップ・データがOSや設定ファイルなどの場合はどうだろうか。読者の中には、「1週間に1回」あるいはそれ以上の間隔でもいいと思われる方がいるかもしれない。だが、パッチの適用などでOSには意外と変更が加わるものだ。可能であれば、こちらも「1日1回」の実行をお勧めする。

このほか、電子メールのデータには、外部(顧客など)と交わした重要な情報が含まれる。したがって、メール・サーバのメール・スプールに格納されている電子メールなどは「2時間に1回」など、より高い頻度でバックアップを行うべきだろう。

バックアップは時間との戦い

おそらく、バックアップと聞いて読者の皆さんが連想されるのは、バックアップ対象となるすべてのデータを完全にコピーする方法(フル・バックアップと呼ぶ)であろう。しかし、企業や組織で運用しているサーバ・マシンの場合、常にフル・バックアップを行えるわけではない。バックアップ・データのサイズが数百MB程度ならまだしも、数十GBともなれば、バックアップにはかなりの時間がかかる。

一般的な部門用のファイル・サーバであれば、夜間や休日はほとんど利用されないはずだ。したがって、夜間(深夜0時から4時までの間など)にフル・バックアップを行える。

一方、Webサーバやメール・サーバのように24時間連続稼働が前提のサーバの場合、バックアップのためにサービス・プログラムを止めるわけにはいかない。また、バックアップによってシステムにかかる負荷も最小限に抑えたいところだ。こうしたケースでは、以下で説明する「差分バックアップ」や「増分バックアップ」の実行を検討することになる。

差分バックアップ

「差分バックアップ」とは、フル・バックアップの実行後に変更または追加されたデータをバックアップしていく手法だ。つまり、フル・バックアップを実行した後、変更されていないファイルはバックアップしない。したがって、フル・バックアップ

に比べればバックアップにかかる時間が短くなる。

差分バックアップを実行するときは、例えば、毎週日曜日にフル・バックアップを行い、月曜日から土曜日までの毎日は差分バックアップを行うといったスケジューリングを行う（**図1**）。

バックアップ・データをリストア（元に戻す）するときは、まず、最後に取得したフル・バックアップのデータをシステムに戻す。続いて、最後に取得した差分バックアップ・データを戻せばよい。

具体的に説明しよう。仮に、上記のスケジューリングで木曜日にディスクがクラッシュしたとする。その場合は、まず日曜日のフル・バックアップ・データをシステムに戻す。続いて、水曜日の差分バックアップ・データ（クラッシュ前の最後のバックアップ・データ）をシステムに戻す。これにより、すべてのバックアップ・データがリストアされる。

差分バックアップの注意点

差分バックアップを行うときに注意が必要なのは、フル・バックアップの実行間隔だ。

フル・バックアップの実行後、時間がたてばそれだけ差分バックアップでバックアップしなければならないデータの量が増えることになる。つまり、フル・バックアップの間隔が長いと、場合によっては差分バックアップによるデータの量がフル・バックアップのそれと同程度になってしまう。差分バックアップを行う際はその点に注意して運用されたい。

増分バックアップ

「増分バックアップ」とは、最後のバックアップの実行後に更新されたデータだけをバックアップする手法だ。差分バックアップと似ているが、差分バックアップは「最後のフル・バックアップ」以降のデータを対象とする。それに対し、増分バックアップは「最後のバックアップ」以降のデータを対象とする。

例えば、日曜日にフル・バックアップ、それ以外の日に増分バックアップを行うとする。この場合、月曜日の増分バックアップには日曜日以降に変更／追加されたデータが収められる。続く火曜日の増分バックアップのデータは、月曜日以降の変更／追加データとなる（**図2**）。

一方、差分バックアップには、月曜／火曜とも、日曜日以降に変更／追加されたすべてのデータが収められる。

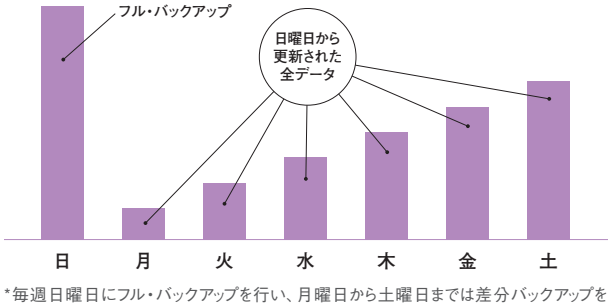
増分バックアップの長所は、1回のバックアップ量を少なくすることができる点だ。また、毎回バックアップするデータの量がほぼ一定となる。差分バックアップでは日々バックアップするデータ量が増えていくが、増分バックアップでは特定の日にデータの変更／追加が集中しないかぎり、1回のバックアップ量はほぼ一定になるのだ。そのため、バックアップ・メディアのサイズが小さい場合や、毎日のバックアップを必ず一定時間内に収めなくてはならないときには、増分バックアップが有効となる。

増分バックアップの欠点

増分バックアップには「リストアが面倒」という大きな欠点がある。

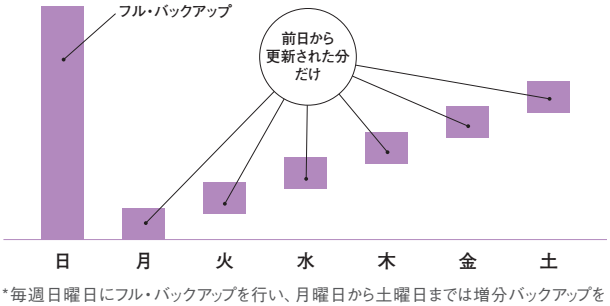
例えば、日曜日にフル・バックアップ、それ以外の日に増分バックアップという運用で、土曜日にディスクがクラッシュし

図1：差分バックアップ



*毎週日曜日にフル・バックアップを行い、月曜日から土曜日までは差分バックアップを行う場合。差分バックアップには、日曜日以降に更新されたすべてのデータが収められる。そのため、毎回データ量が増えていく。しかしその分、リストアが容易になるのが特徴。

図2：増分バックアップ



*毎週日曜日にフル・バックアップを行い、月曜日から土曜日までは増分バックアップを行う場合。増分バックアップは、一度にバックアップするデータ量が少ないため、容量の小さなバックアップ・メディアを利用でき、バックアップ時間も短縮できる。しかし、リストアが面倒になるという欠点がある。

た場合を考えてみよう。このとき、データを元に戻すには、まず日曜日のフル・バックアップ・データをリストアする。そして次に、月曜日、火曜日、水曜日……と、各曜日ごとの増分バックアップのデータを金曜日の分までそれぞれリストアしていく必要があるのだ。

増分バックアップの長所は、一度にバックアップするデータ量が少なく、バックアップ時間を短縮できる点にあるが、その反面、リストアには手間と時間がかかることになる。

なお、増分／差分バックアップとも、最初に取得したフル・バックアップのデータが失われると、データの完全な復旧は不可能となる。増分／差分のいずれかでバックアップを行うときは、フル・バックアップのデータを命に代えても（!）大切に保管しておかなければならない。

バックアップ・メディア ―MO、CD-R/RW―

最後に、バックアップ・データを保存するメディアの種類とその選択方法を簡単に説明しておこう。

個人用途で一般的に利用されるメディアは、MOやCD-R/RWだろう。しかし、それらは容量が少ないため、サーバ・マシンのバックアップには向かない。もっとも、「今すぐにバックアップを取らなければならないが、適切なメディアがない」といった場合は、CD-R/RWも1つの選択肢となろう。最近のマシンには、大抵、CD-R/RWドライブが標準で装備されているからだ。

また、データが数GBの範囲に収まるのであれば、容量がGBクラスのMOを候補に入れてもよいだろう。MOは入手が容易であるし、CD-Rとは違ってメディアを再利用することが可能だ。また、MOはいったんシステムにマウントしてしまえば、通常のディスクと同様に扱える。CD-Rのように特別なツールを利用する必要がないのもメリットだ。

バックアップ・メディア ―ハードディスク、テープ―

バックアップ・データが数十GBに上る場合、バックアップ・メディアはハードディスクまたは磁気テープに限られる。このうち、ハードディスクはメディアの取り外しが行えない。つまり、ハードディスクをバックアップ・メディアとするときは、複数台のハードディスクを準備しないかぎり、「1日分のフル・バックアップ・データのみを保持する」という運用しか行えなくなる。

一方、磁気テープは、古くからサーバ・マシンのバックアップに利用されてきたメディアだ。数十GBを超えるデータを扱えるリムーバブル・メディアは、磁気テープ以外にはない。そのため、サーバ・マシンでは必然的に磁気テープを導入することになるだろう。

なお、磁気テープにはさまざまな規格があるため、なじみがない初心者はどれを選択すればよいかで悩むことだろう。これまで、小規模なサーバ・マシンでは「QIC」（Quarter-Inch Cartridge）や「DDS」（Digital Data Storage）と呼ばれる規格のテープが比較的多く採用されてきた。しかし、最近では「DLT」（Digital Linear Tape）と呼ばれる規格に人気が集まっている。また、DLTをさらに大容量化した「SDLT」（Super DLT）と呼ばれる規格も登場している。新規にテープ・ドライブを導入するのであれば、DLTかSDLTを採用するとよいだろう。

ちなみに、今回の原稿執筆にあたっては、テープ装置メーカーのタンベルグデータの好意により、同社の「SDLT320」（SDLT対応）というテープ・ドライブを借りることができた（**写真1**）。本稿ではテープ・ドライブへのバックアップ方法も解説しているが、それらの記事の検証は「SDLT320」を用いて行った。

写真1：タンベルグデータのSuper DLTテープ・ドライブ「SDLT320」（左）と磁気テープ。非圧縮時で160GBものデータをバックアップ可能。お勧め度100%

対応OS	Linux、Solaris、NetWare、Windows NT/2000/XPなど
価格	オープン・プライス
問い合わせ	03-5566-2898
URL	http://www.tandberg.co.jp/

Part 3 dump/cpio/tarによるバックアップとリストア

基本ツールの活用法

本パートでは、UNIX系OSで古くから利用されている3つのバックアップ・ツール（dump、cpio、tar）を紹介する。各ツールによってバックアップの対象が異なるので、それぞれの機能を把握しておこう。

パーティションのバックアップ —dump—

dumpコマンドは、UNIX系OSで古くから利用されているバックアップ・ツールだ。他の多くのバックアップ・ツールがファイルを対象とするのに対して、dumpはファイルシステム（例えば、「/dev/hda1」といったパーティションに作成されたファイルシステム）を対象とする。

dumpは、その実行時に指定するパーティションなどのすべての内容をバックアップする。そのため、機能や用途別にパーティションが分割されていないシステムでは扱いづらい。例えば、ホーム・ディレクトリの/homeが独立したパーティションにはなっておらず、ルート（/）パーティションに含まれていたとする。そのようなシステムでは、/home単体のバックアップを取ることはできない。また、dumpはパーティションを対象とするため、rootでの操作が前提となる(*1)。

ダンプ・レベル

dumpには「ダンプ・レベル」という値（0～9までの値）がある。差分バックアップや増分バックアップを実行する際は、ダンプ・レベルを利用する。

「0」は「すべてのデータのバックアップ」を意味するダンプ・レベルだ。また「1～9」の値は、それぞれ「より小さな値で取得されたバックアップ・データ(なおかつ、その中で最新のもの）を調べて、それ以降に更新されたデータをバックアップする(つまり、差分バックアップや増分バックアップを行う)」ために利用するダンプ・レベルである(実際の活用法は後述)。

それでは、dumpはどのようにしてダンプ・レベルや過去に実行されたバックアップの日時を知るのであろうか。実は、dumpはそれらの情報を、バックアップ取得時に「/etc/dumpdates」というファイルに記録する。そのファイルに記録された情報によって、バックアップの履歴を調べているのだ。

例1：dumpの基本的な利用方法

①パーティション「/dev/hdb1」のフル・バックアップを行う場合

●実行形式
```
dump 0uf バックアップ先のアーカイブ名 ファイルシステム（パーティション）
```

●オプション
0：ダンプ・レベル（0はフル・バックアップ）
u：「/etc/dumpdates」（dumpコマンドの使用記録を保存するファイル）の更新を意味する
f：アーカイブ名を指定する

●実行例1
```
dump 0uf /backup/home.bak /dev/hdb1
```
「/dev/hdb1」の内容を/backup/home.bakディレクトリにフル・バックアップ

●実行例2
```
dump 0uf /dev/st0 /dev/hdb1
```
「/dev/hdb1」の内容を「/dev/st0」にフル・バックアップ（「/dev/st0」はSCSIテープ・ドライブのデバイス・ファイル名）

②パーティション「/dev/hdb1」の差分バックアップを行う場合

●実行形式
```
dump [ダンプ・レベル]uf バックアップ先のアーカイブ名 ファイルシステム（パーティション）
```

●実行例
```
dump 1uf /dev/st0 /dev/hdb1
```
「/dev/hdb1」の内容をダンプ・レベル「1」でテープ「/dev/st0」にバックアップ（「/dev/st0」はSCSIテープ・ドライブのデバイス・ファイル名）。この場合、ダンプ・レベル「0」（フル・バックアップ）からの差分バックアップが取得される

dumpの実行例

dumpでフル・バックアップを取得するには、**例1**-①のようにdumpコマンドを実

*1：ディスクを表すデバイス・ファイル（/dev/hda1など）が、root以外のグループにも読めるようになっている場合は、root以外のユーザーでもdumpを実行できる(例えば、Red Hat Linuxでは各デバイス・ファイルをdiskというグループのユーザーでも読み書きできるようになっている)。ただしその場合、「/etc/dumpdates」というファイルに対して、当該グループからの読み書きの権限を付与しておく必要がある。「/etc/dumpdates」にはdumpコマンドの使用記録が書き込まれる。

行する。また、差分や増分バックアップを行うには、例1-②の実行形式でdumpを実行すればよい。

ダンプ・レベル「1」を指定してdumpを実行すれば、フル・バックアップからの差分バックアップを行える。

また、ダンプ・レベルを「1、2、3……」と増加させながらバックアップを行えば、前回のバックアップ・データからの増分バックアップを行える。例えば、日曜日にフル・バックアップ、月曜日から土曜日までは増分バックアップを行うとする。その場合、日曜日はダンプ・レベル「0」、月曜日は「1」、火曜日は「2」……というように、ダンプ・レベルを上げていけばよい。

dumpデータのリストア ―restore―

dumpでバックアップしたデータをリストアする（元に戻す）には、restoreというコマンドを利用する。例2に、その実行方法を示すので参照されたい。

なお、restoreは、カレント・ディレクトリ（コマンドを実行したディレクトリ）にデータを書き出す。そのため、あらかじめデータをリストアするディレクトリに移動してからコマンドを実行されたい。

増分バックアップを行っているケースでリストアを行うときは、ダンプ・レベルが若いほうから順番にrestoreを実行する。例えば、ダンプ・レベル「0、1、2」という3本のバックアップ・データがあるとする。その場合、「0、1、2」の順でリストアを実行する。

このとき、カレント・ディレクトリに「restoresymtable」というファイルが作成される。このファイルにはリストアに必要な情報が入っているので、リストアが完了するまで消してはならない。

例2：restoreの基本的な利用方法（バックアップ・メディアとしてテープ・ドライブを利用しているケースを想定）

①アーカイブの内容を閲覧する場合

●実行形式
```
restore tf バックアップ先のアーカイブ名
```

●オプション
```
t：ファイル名の一覧を出力する
f：アーカイブ名を指定する
```

●実行例
```
restore tf /dev/st0
```
テープ「/dev/st0」に含まれるファイルの一覧を表示（「/dev/st0」はSCSIテープ・ドライブのデバイス・ファイル名）

②アーカイブのリストアを行う場合

●実行形式
```
restore rf バックアップ先のアーカイブ名
```

●オプション
```
r：アーカイブの内容をリストアする
f：アーカイブ名を指定する
```

●実行例
```
restore rf /dev/st0
```
テープ「/dev/st0」に含まれる全ファイルをカレント・ディレクトリにリストア（「/dev/st0」はSCSIテープ・ドライブのデバイス・ファイル名）

③アーカイブから特定のファイルだけをリストアする場合

●実行形式
```
restore xf バックアップ先のアーカイブ名 リストアするファイルの名前
```

●オプション
```
x：指定されたファイルをアーカイブからリストアする
f：アーカイブ名を指定する
```

●実行例
```
$ restore -xf /dev/st0 ./home/someone/.bashrc
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 # (none if no more volumes): 1
set owner/mode for '.'? [yn] y
```
テープ「/dev/st0」中から、「./home/someone/.bashrc」というファイルだけをカレント・ディレクトリにリストアする

ボリューム番号を指定（アーカイブが1本なら「1」を指定）

もともとの所有者／パーミッションでリストアするなら「y」

対話モード

restoreコマンドには、データの復旧をシステムと対話的に行うための「i」というオプションがある。このオプションを指定してrestoreを実行すると、「restore>」というコマンド・プロンプトが表示される。

対話モードでリストアを実行すると、このコマンド・プロンプトに続けてcdやlsなどのコマンドを実行でき、バックアップ・データの内容を通常のファイルシステムと同様の感覚で参照できる。

したがって、大きなバックアップ・データの中からある特定のファイルのみを探し出すときは便利だ。基本的な使い方は、「cdでアーカイブ内を移動する」「lsでファイルを探し出す」「addでリストアするファイルを選択する」「deleteで選択の取り消しを行う」といった具合だ。また、addで選択したファイルを最終的にリストアするには、extractというコマンドを実行する。

ファイルのバックアップ ―cpio―

cpioコマンドは、dumpと同様、UNIX系OSで広く使用されてきたバックアッ

例3：cpioの基本的な利用方法（バックアップ・メディアとしてテープ・ドライブを利用しているケースを想定）

①指定したディレクトリの内容をすべてcpioアーカイブに格納する場合（フル・バックアップに相当）

●実行形式
```
find ディレクトリ名 | cpio -oa > バックアップ先のアーカイブ名
```

●オプション
o：出力モード（バックアップを行うときに指定）を意味する
a：ファイルへの最終アクセス時刻の保存を意味する

●実行例
```
find ./home | cpio -oa > /dev/st0
```
findで/home内の全ファイルを検索／検索結果を次のコマンドに渡す／cpioは受け取ったファイルをテープ「/dev/st0」（SCSIテープ・ドライブのデバイス・ファイル名）にバックアップ

②アーカイブの内容を閲覧する場合

●実行形式
```
cpio -it < バックアップ先のアーカイブ名
```

●オプション
i：入力モード（バックアップ・データの読み込み時に指定）を意味する
t：ファイル名の一覧表示を意味する

●実行例
```
cpio -it < /dev/st0
```
テープ「/dev/st0」（SCSIテープ・ドライブのデバイス・ファイル名）の内容を閲覧

③アーカイブのリストアを行う場合

●実行形式
```
cpio -idm < バックアップ先のアーカイブ名
```

●オプション
i：入力モード（バックアップ・データの読み込み時に指定）を意味する
d：必要に応じたディレクトリの作成を意味する
m：ファイルの最終更新時刻をアーカイブ時の時刻とする

●実行例
```
cpio -idm < /dev/st0
```
テープ「/dev/st0」（SCSIテープ・ドライブのデバイス・ファイル名）の内容をリストア

④アーカイブ中から特定のファイルを指定してリストアする場合

●実行例
```
cpio -idm < /dev/st0 ./home/someone/.bashrc
```
③と同じ／リストアしたいファイルを指定（テープ「/dev/st0」内のファイル）

⑤/homeを対象に差分バックアップ／増分バックアップを行う場合

●実行例
```
find ./home -newer /var/cpio.full | cpio -oa > /dev/st0
```
/homeの中から／cpio.fullの作成後に更新されたファイルを検索／検索結果を次のコマンドに渡す／cpioは受け取ったファイルをテープ「/dev/st0」（SCSIテープ・ドライブのデバイス・ファイル名）にバックアップ

プ・ツールだ。主にファイル単位でのバックアップを行う際に利用される。

cpioの特徴は、標準入力からバックアップを行うファイルのリストを受け取り、それらのファイルの内容を標準出力に書き出すという点である（バックアップ時にはバックアップ・ファイルに「>」でリダイレクトする）。ファイル・リストの作成には一般的にfindコマンドを利用する。

つまり、cpioの強みは、findによってバックアップ対象とするファイルをリストアップできる点にある。例えば、「ユーザーfooが所有しているすべてのファイル」「1カ月前以降に変更されたすべてのファイル」といった検索条件でfindを実行すれば、その条件に合致したファイルだけをバックアップできるわけだ。

cpioの基本的な利用法については**例3**を参照していただきたい。例3-①では、findによって/home内の全ファイルをリストアップし、そのすべてをcpioによってバックアップするよう指示している。

cpioによる差分／増分バックアップ

ファイル検索コマンドのfindには「-newer」というオプションがある。「-newer」オプションに続けてファイル名を指定すると、そのファイルが修正された日以降に更新された他のファイル群がリストアップされる。この機能を利用すれば、cpioを使って差分バックアップや増分バックアップを行える。以下、その具体的な方法を紹介する。

●差分バックアップ

まず、適当なファイル（例えば、cpio.full）をtouchコマンドなどで/varディレクトリなどに作成する。このcpio.fullは、後ほど差分バックアップを行うときに、前回のフル・バックアップを実行した日時を特定するために利用する。cpio.fullを作成したら、例3-①の要領でまずはフル・バックアップを実行する（/homeのデータをバックアップする場合）。

フル・バックアップを実行した後、/homeの差分バックアップを取るには、例3-⑤のように「find ./home -newer /var/cpio.full」と指定して、cpio.fullの作成よりもあとで更新されたファイルをfindに検索させる。これにより、フル・バックアップの実行後に追加／更新されたファイルだけがfindによってリストアップされる。あとは、例3-⑤の後半部分「| cpio -oa > /dev/st0」によってテープ「/dev/st0」への差分バックアップが行

われることになる（「/dev/st0」はSCSIテープ・ドライブのデバイス名）。

●増分バックアップ

増分バックアップも差分バックアップと同じ要領で行える。差分バックアップと異なるのは、バックアップを取得するつどtouchコマンドでcpio.fullファイルを作成（更新）する点だ。つまり、cpio.fullの作成時刻をそのつど現在時刻へと変更するわけだ。あとは、例3-⑤を実行することで、前回のバックアップ実行時以降に更新されたファイルだけがバックアップされることになる（つまり、増分バックアップを行ったことになる）。

ディレクトリのバックアップ —tar—

tarは、多くのLinuxユーザーが使用しているアーカイブ・ツールだろう。tarは主に、指定したディレクトリ内のすべてのファイルをバックアップする際に利用される。tarアーカイブは、Windowsなどの他のOSでも扱える。そのため、複数のOSにまたがってアーカイブを利用するような場合には重宝するだろう。tarの利用方法については、**例4**を参照されたい。

tarの欠点は、差分／増分バックアップが面倒になるという点だ。tarは、cpioとは異なり、標準入力からファイルの一覧を読み込んで、特定のファイルだけをアーカイブするといったことが行えない。tarの「-T」というオプションを利用すれば、「あるファイルからファイル名の一覧を読み込んで、それらのファイルをアーカイブする」ことができるが、その方法はcpioほどスマートではない（例4-⑤）。

例4：tarの基本的な利用方法

①指定したディレクトリの内容をすべてtarアーカイブに格納する場合（フル・バックアップに相当）

●実行形式
```
tar cf バックアップ先のアーカイブ名 ディレクトリ名
```

●オプション
c：バックアップ・データのアーカイブの作成を指示する
f：アーカイブ名を指定する

●実行例
```
tar cf /backup/home_bak.tar ./home
```
./homeディレクトリの内容をhome_bak.tarファイルへバックアップ

②アーカイブの内容を閲覧する場合

●実行形式
```
tar tf バックアップ先のアーカイブ名
```

●オプション
t：ファイル名の一覧を出力する
f：アーカイブ名を指定する

●実行例
```
tar tf /backup/home_bak.tar
```
home_bak.tarファイルの内容を閲覧

③アーカイブのリストアを行う場合

●実行形式
```
tar xf バックアップ先のアーカイブ名
```

●オプション
x：アーカイブをリストアする
f：アーカイブ名を指定する

●実行例
```
tar xf /home/home_bak.tar
```
home_bak.tarの内容をリストア

④アーカイブ中から特定のファイルだけをリストアする場合

●実行例
```
tar xf /backup/home_bak.tar ./home/someone/.bashrc
```
/backup/home_bak.tarファイルから「./home/someone/.bashrc」というファイルだけをリストア

⑤/homeを対象に差分バックアップ／増分バックアップを行う場合（以下の2つのコマンドを実行）

●実行例
```
find ./home -newer /var/tar.full > /tmp/tar.list
```
/homeの中からtar.fullの作成後に更新されたファイルを検索　検索結果を「/tmp/tar.list」ファイルに出力
```
tar cf /backup/home_bak.tar -T /tmp/tar.list
```
tarの「-T」オプションで「/tmp/tar.list」にリストされたファイルを読み込み、それらのファイルを/backup/home_bak.tarファイルにバックアップ

操作に慣れておく

以上、簡単ではあるがLinux上で使用される基本的なバックアップ・ツールを紹介した。実際にバックアップやリストアを行う際は、これらのツールのマニュアルに目を通し、使用方法を十分確認しておいていただきたい。また、実際の運用に入る前に、できるだけ各ツールで簡単なアーカイブを作成してみて、そのリストアも実行しておくようにしたい。

運用しているシステムのディスク障害が実際に起こったときには、どうしても慌ててしまうものだ。そのため、特にリストアの方法には慣れておく必要がある。

Part 4 MO、ハードディスク、リモート・ディスクを利用する方法

パーティションの作成とマウント

MOやハードディスクなどにデータをバックアップする際は、それらのディスクにパーティションを作成し、Linuxにマウントする必要がある。本パートでは、その設定方法を解説する（実際のバックアップ方法については他のパートを参照されたい）。

ハードディスクやMOへのバックアップ

バックアップ・メディアとしてハードディスクやMOなどを利用するときは、まずメディア内にパーティションを作成し、さらにそのパーティションをフォーマットする（使用可能な状態にする）必要がある。そのためには、fdiskおよびmkfsというコマンドを利用する。

例1にfdiskによるパーティションの作成方法を示す。また**例2**に、mkfsによるパーティションのフォーマット方法を示す。ここでは、ハードディスク（デバイス名「hdb」。IDE接続のハードディスクのデバイス名は「hd」で始まる）に対する操作方法を示しているが、他のメディアでもやり方は同様だ。例えば、メディアがSCSI対応のMOであれば、デバイス名が「sda」などになるだけだ（SCSI接続のMOのデバイス名は「sd」で始まる）。

fdiskおよびmkfsでメディアをフォーマットしたら、当該パーティションをマウントすることが可能になるので、マウントしたメディアにバックアップ・データを書き込む（その方法についてはパート3を参照）。

運用上の注意点

fdiskを利用して未フォーマットのメディアにパーティションを作成すると、fdiskが大量のエラーや警告を出してくることがある。しかし、落ち着いてメッセージを読めば、fdiskが未フォーマットのディスクの情報を誤認して、存在しないパーティションの情報を報告しているだけだということがわかるはずだ。そうしたエラーが出たときは、「誤認」されたパーティションを削除して、改めてパーティションを作成しなおせばよい(*1)。

なお、ハードディスクやMOをバックアップ・メディアとして利用するときは、バックアップが完了した後に必ずアンマウントを行う習慣をつけておこう（MOなどのリムーバブル・メディアについては、メディアをドライブから取り出す習慣もつけておく）。それにより、誤ってバックアップ・データを消去してしまったり、大切なバックアップ・データがクラッカーによって書き換えられてしまったりといったリスクを回避することができる。

NFSを利用したバックアップ

バックアップを行う際、手ごろなバックアップ・メディアが見つからないときは、他のマシンで公開されている共有ディスクにネットワーク経由でバックアップするという方法もある。このとき、バックアップ先のマシンがUNIX系OSであれば、NFSを利用するとよい（紙幅の都合上、NFSの設定方法については説明を割愛する）。

NFSを利用してバックアップを行うときは、リモート・マシンのディスクをNFSによって手もとのマシンのディレクトリにマウントする（**例3**）。いったんマウントしてしまえば、あとはローカルのディスクと同様にリモートのディスクを利用できるようになる。

ただし、NFSを利用してバックアップを行うときは、セキュリティ上の注意が必要だ。NFSは、セキュリティに難のあるプロトコルであるため、例えばインターネットに接続しているようなマシンでは使用すべきでない(*2)。また、他のサーバに保存することがセキュリティ上好ましくない機密データなども、NFSでバックアップすべきではない。

Windowsディスクのマウント

Linux上のデータをリモートにあるWindowsマシンのディスクにバックアップするのであれば、Windowsマシンのディ

*1：パーティションを削除する場合は、fdiskのコマンド「d」を利用する。「d」を実行すると削除したいパーティション番号を尋ねられるので、誤認されたパーティションの番号を指定する。誤認されたパーティションがなくなるまでその操作を繰り返せばよい。
*2：NFSは、公開ディレクトリへのマウントを許可するクライアント（通信相手）をIPアドレスで決定する。そのため、IPアドレスの詐称（自分のIPアドレスを偽って接続するクラッキング行為）に弱い。

スクをLinuxのディレクトリにマウントすればよい。そのためには、ファイル・サーバ・プログラムのSambaに付属しているsmbmountというツールを利用するとよい(smbmountを利用するにはSambaをインストールしておく必要がある。Sambaのセットアップ方法の説明は紙幅の都合上割愛する)。

smbmountを利用する際は、まずWindowsマシン上に共有フォルダを作成し、「Windows上の」適当なユーザーにそのフォルダへの書き込みを許可しておく。あとは、Linuxマシンからsmbmountを利用して、Windowsマシンの共有フォルダをマウントすればよい。

例4に、smbmountコマンドの実行方法を示す。例4-②は、Windowsマシン「somepc」上の共有フォルダ「lindata」を、Linuxマシンの/winfolderディレクトリへ、ユーザー「backup」でマウントする例だ。例4-②の最後の「username」は「(Windows上の)どのユーザーでLinuxのディレクトリにマウントするか」を指定するためのオプションである。

例4-②を実行するとユーザー「backup」のパスワードを尋ねられる。「backup」のパスワードを入力するとマウントが行われる。これで、通常のディスクと同様にバックアップを行えるようになる。マウントを解除するときは、以下のようにumountコマンドを実行すればよい。

```
# umount /winfolder
```

なお、cronなどを利用した夜間の自動バックアップを行うときは、管理者がその場にいない。そのため、smbmountが実行されるときにパスワードの入力を行えない。パスワードの入力を自動化するには、適当なファイルに**例5**-①のような書式でユーザー名とパスワードを記述しておく。そして、例5-②のように、smbmountコマンドの引数にそのファイルを指定しておけば(「credentials」というオプションで指定)、管理者がパスワードを手入力する必要はなくなる。

ただし、この場合はパスワードがファイル中に直接記述されるため、当該ファイルを他人に見られないよう注意する必要がある。

例1:fdiskを利用したパーティションの作成

```
# fdisk /dev/hdb                                  ← fdiskでハードディスク「/dev/hdb」にパーティションを作成

コマンド (m でヘルプ): n                          ← コマンド「n」で新しいパーティションの作成を指示
コマンドアクション
   e   拡張
   p   基本領域 (1-4)
p                                                  ← 作成する領域を基本パーティションとする
領域番号 (1-4): 1                                  ← パーティション番号を指定(1つ目のパーティションなら「1」)
最初 シリンダ (1-1245, 初期値 1):                  ← パーティションの開始位置を指定(何も指定しなければ「1」)
初期値 1 を使います
終点 シリンダ または +サイズ または +サイズM または +サイズK (1-1245, 初期値 1245):
初期値 1245 を使います                             ← パーティションの終了位置を指定。何も指定しなければ最終シリンダの「1245」となる(注1)

コマンド (m でヘルプ): p                          ← コマンド「p」で現在の状態を表示

ディスク /dev/hdb: ヘッド 255, セクタ 63, シリンダ 1245
ユニット = シリンダ数 of 16065 * 512 バイト

 デバイス ブート  始点      終点  ブロック   ID  システム
/dev/hdb1               1      1245  10000431   83  Linux   ← 「/dev/hdb1」パーティションが作成されている

コマンド (m でヘルプ): w                          ← コマンド「w」で現在の設定を書き込む
……以下略……
```

注1:シリンダはハードディスクの記録単位の1つ。fdiskでは、パーティションの領域をシリンダ番号か、サイズ(KB/MB)で指定する。

例2:mkfsによるパーティションのフォーマット例

```
# mkfs -t ext3 /dev/hdb1    ← 「/dev/hdb1」パーティションをext3ファイルシステムでフォーマット
# mkfs -t ext3 /dev/sda1    ← 「/dev/sda1」パーティションをext3ファイルシステムでフォーマット
```

例3:NFSマウントの実行例

```
●マウント
# mount -o noexec bserver.someorg.com:/backup /bsdir
        マウント先の実行ファイルを実行できなくするオプション
                  サーバ名
                          サーバ側の公開ディレクトリ
                                  クライアント側のマウント・ポイント

●アンマウント
# umount /bsdir
        クライアント側のマウント・ポイント
```

例4:smbmountコマンドの実行例

```
①実行形式
$ smbmount //Windowsマシン名/共有名 マウント場所 -o オプション

②実行例
$ smbmount //somepc/lindata /winfolder -o username=backup
             Windowsマシンの名前
                    共有フォルダの名前
                            Linuxのマウント・ポイント(ディレクトリ)
                                    Windows上のどのユーザーでLinuxにマウントするかを指定。ここではユーザー「backup」
```

例5:smbmountコマンドのパスワード入力を自動化する方法

```
# cat /root/smbpassword                          ← 適当なファイル(smbpassword)を作成
username = backup                                ← smbmountでマウントする際のユーザーの名前
password = ,bu@ps-98/                            ← 当該ユーザーのパスワード
# smbmount //somepc/lindata /winfolder -o credentials=/root/smbpassword
                                                 ← smbmountの実行例
```

①
②

Part 5 イメージ・ファイルを作成し、cdrecordでコピーする

CD-R/RWへのバックアップ

CD-R/RWは容量こそ小さいものの、Linux以外のコンピュータでも簡単に利用できるという特徴がある。小容量だが大切なデータや、さまざまなマシンで使用されるデータをバックアップするときには有力な選択肢となるだろう。

CD-R/RWへの書き込みには専用ツールが必要

CD-RやCD-RWにデータを書き込むには、専用のソフトウェアが必要だ。

Linuxで広く利用されているソフトウェアには、GUIで操作可能な「X-CD-Roast」とコマンドラインから利用する「cdrecord」の2つがある。ここでは、サーバで利用することを想定して、コマンドライン・ツールのcdrecordの利用方法を説明しよう。

ドライブ情報の確認とイメージ・ファイルの作成

CD-R/RWへデータを書き込むには、まず、cdrecordに「-scanbus」オプションを付けて実行する。すると、**例1**のようにCD-R/RWドライブの情報が表示される。この例からは、CD-R/RWドライブのBUS(*1)番号が「0」、ID(*2)番号が「0」、LUN(*3)番号が「0」であることがわかる。これらの情報は、CD-R/RWにデータを書き込む際に必要となるので、あらかじめ確認しておく。

CD-R/RWドライブの情報を取得したら、cdrecordに付属しているmkisofsというコマンドを利用して、イメージ・ファイル(*4)をISO9660形式(*5)で作成する。**例2**に示したのは、/homeの内容をCD-ROM用のイメージ・ファイルとする例だ。途中、一部のファイル名をCD-ROMで利用可能なファイル名へと変換したことを告げるメッセージが表示されるが、それはあまり気にしなくてよい。

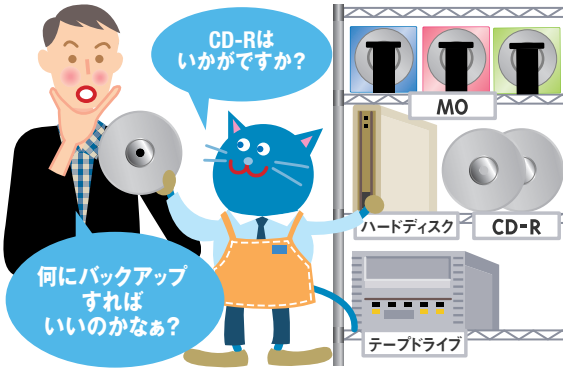
例1:ドライブの情報を取得する方法

●「cdrecord -scanbus」の使用例

```
# cdrecord -scanbus
Cdrecord 1.10 (i686-pc-linux-gnu) Copyright (C) 1995-2001 J?rg Schilling
Linux sg driver version: 3.1.24
Using libscg version 'schily-0.5'
scsibus0:                                                    CD-RWドライブの情報
        0,0,0     0) 'SONY    ' 'CD-RW  CRX145E  ' '1.0b' Removable CD-ROM
        0,1,0     1) 'MATSHITA' 'PD-1 LF-1196    ' 'A104' Removable CD-ROM
        0,2,0     2) *
左から順にBUS、ID、LUNの番号
```

データの書き込み

CD-ROM用のイメージ・ファイルを作成したら、それをCD-R/RWに書き込む。そのためには、**例3**のようにしてcdrecordを実行すればよい。イメージ・ファイルの書き込みには時間がかかるが、cdrecordはその進行状況を表示しない。その



*1:BUSはSCSIバスの番号。
*2:IDは同一バスに接続されている各SCSI機器に振られる番号
*3:LUN(Logical Unit Number)は、あるSCSI機器を論理的に複数台の装置として取り扱う場合に、個々の論理機器に振られる番号。例えば、CD-ROMチェンジャの場合、チェンジャ自体は1つの機器であるが、チェンジャ内の各ドライブがそれぞれ論理装置となり、個別のLUNが振られることになる。
*4:イメージ・ファイルとは、ディスクやメモリの内容をそのままファイルに格納したもの。
*5:ISO9660は、CD-ROM用のファイルシステム。

例2：mkisofsコマンドによるイメージ・ファイルの作成例

●mkisofsの実行例

```
$ mkisofs -r -J -o /var/tmp/backup.img /home
```

/var/tmp/backup.img：イメージ・ファイルの出力先　/home：イメージ・ファイルの作成元

＜オプション＞

-r：RockRidge拡張を行う（UNIX系OSで9文字以上のファイル名を使用できるようにする）。さらに、UIDとGIDを0（0はrootのUID/GID）にしてsuidビット(注1)を無効にする。
-J：Joliet拡張を行う（Windowsで9文字以上のファイル名を使用できるようにする）。
-o：イメージ・ファイルの出力先を指定。このオプションを指定しないと、イメージ・ファイルは標準出力に出力される。

●上のコマンドを実行すると、途中でファイル名の変換を知らせるメッセージが表示される（無視してよい）

```
Using VISIBOOO.;1 for  /home/riue/.gimp-1.2/palettes/Visibone2 (Visibone)
  3.46% done, estimate finish Sun Feb  2 18:16:41 2003
```

●イメージ・ファイルの作成が完了すると、このようなメッセージが表示される（283MBはサイズ）

```
144960 extents written (283 Mb)
```

注1：suidビットは、ファイル所有者の権限でプログラムを実行できるようにする特殊なパーミッション。例えば、パスワードを変更するためのpasswdコマンドの場合、所有者がrootでsuidビットが設定されている。それによって、一般ユーザーでもファイル所有者（root）の権限でpasswdコマンドを実行できる。

ため、本当に書き込んでいるのかどうか途中で心配になるかもしれないが、気長に待つしかない。

例3を見ればわかるように、cdrecordコマンドには複数のオプションが指定される。そのため、頻繁にCD-R/RWを作成する場合は、「/etc/cdrecord.conf」ファイルにふだん利用するオプションやデバイスを定義しておくとよい。そうすれば、コマンド入力の手間が省ける。

例4に、「/etc/cdrecord.conf」ファイルの記述例を示す。このように設定すると、「speed」「dev」「fs」の3つのオプションの入力を省けるようになる。つまり、例3と同等の処理を以下のコマンドラインで実現することができる。

```
# cdrecord -eject -dao /var/tmp/backup.img
```

差分／増分バックアップの取り方

CD-R/RWへのバックアップでは、「指定したディレクトリの内容をすべてCD-R/RWに書き込む」という方法を取る。そ

例3：cdrecordの実行例

```
# cdrecord -eject dev=0,0,0 speed=4 -dao /var/tmp/backup.img
```

-eject：書き込み後、CD-Rを自動的に排出する
dev=0,0,0：先に調べたBUS/ID/LUNの番号
speed=4：書き込み速度（この例では4倍速）
-dao：Disk At Onceモード（1枚のメディアに全データを1回で書き込む）
/var/tmp/backup.img：CD-Rに書き込むイメージ・ファイル

*cdrecordに「-dummy」オプションを付ければ「書き込みテスト」が実施される。
*cdrecordに「blank=値」という引数を指定すればCD-RWの内容を消去できる。通常は「blank=fast」と指定すればよい。ただし、データが壊れているCD-RWでは「blank=all」と指定しないと消去できないことがある（「blank=all」を実行すると処理時間が長くなる）。

例4：「/etc/cdrecord.conf」ファイルの内容例

```
$ cat /etc/cdrecord.conf
CDR_DEVICE=cdrw
CDR_SPEED=4
CDR_FIFOSIZE=4m

# 以下はデバイスの定義
cdrw=          0,0,0   -1      -1      ""
cdr=           0,1,0    8      -1      "burnproof"
```

CDR_DEVICE=cdrw：CD-R/RWドライブの設定。cdrwがどのドライブかは下記で定義（これにより、例3の「dev」オプションを指定する必要がなくなる）
CDR_SPEED=4：書き込み速度の設定（これにより、例3の「speed」オプションを指定する必要がなくなる）
CDR_FIFOSIZE=4m：FIFOサイズ(注1)の設定（これにより「fs」オプションを指定する必要がなくなる）
"burnproof"：オプション「deviceopts」に指定する値(注2)
cdr=：デバイス名
0,1,0：BUS/ID/LUNの番号
8：書き込み速度（「-1」は上で設定した「CDR_SPEED」の値を利用するという意味）
-1：FIFOサイズ（「-1」は上で設定した「CDR_FIFOSIZE」の値を利用するという意味）

注1：FIFOサイズの設定によって、CD-R/RWへの書き込み中に使用するバッファ（メモリ領域）を増やすことができる。初期値は4MB（4m）。この値は利用するドライブの性能に応じて設定する必要があるため、ドライブのマニュアルをよく読んでから設定されたい。
注2：現在のcdrecordのバージョンではオプション「deviceopts」に指定できる値は「burnproof」のみ。「burnproof」は、ドライブのBurn Proofという機能を有効にする値。

Column 1

イメージ・ファイルの内容を確認する

イメージ・ファイルをCD-R/RWにコピーする前に内容を確認したいときは、ループバック・デバイスという機能を利用すればよい。ループバック・デバイスとは、あるファイルをファイルシステムとしてマウントできるようにする機能だ（ファイルシステムとしてマウントできれば、その内容を参照できる）。

例えば、イメージ・ファイルbackup.imgを、/tmp/cdimageにマウントするには例Aのようにmountコマンドを実行すればよい。

例A：イメージ・ファイルの内容を確認

```
# mount -t iso9660 -r -o loop backup.img /tmp/cdimage
```

-t iso9660：「-t」オプションでCD-ROMのファイルシステム（ISO9660）を指定
-r：「-r」オプションで「読み込み専用」を指示
-o loop：「-o」オプションで利用するオプションを指定。ここでは、「loop」オプション（ループバック・デバイスの使用）を指示
backup.img：イメージ・ファイルの名前
/tmp/cdimage：イメージ・ファイルのマウント先

の際、書き込まれたCD-R/RWのディレクトリ構成は、バックアップ時に指定したディレクトリの構成そのものとなる。つまり、ディレクトリの内容は無条件にすべてCD-R/RWへと書き込まれるため、パート3で紹介したcpioのように「ある日時以降に変更／追加されたファイルだけを選択して差分／増分バックアップを取る」といったことは行えない。

そのため、CD-R/RWへの差分／増分バックアップは、**例5**のような方法で行う。例5では、事前にcpioやdumpなどで増分バックアップのアーカイブ・ファイルを作成し、それをイメージ・ファイルにしてからCD-R/RWに書き込んでいる。

例5：CD-R/RWで増分バックアップを取る方法

```
$ cd /                                                          ← /ディレクトリへ移動
$ find ./home -newer /var/cpio.full | cpio -oa > /var/tmp/backup/backup.cpio   ← cpioを利用してbackup.cpioという増分バックアップのアーカイブ・ファイルを作成（コマンドの詳細はパート3の例3-⑤を参照）
$ mkisofs -r -J -o /var/tmp/backup.img /var/tmp/backup          ← イメージ・ファイル（backup.img）の作成（コマンドの詳細は例2を参照）
# cdrecord -eject dev=0,0,0 speed=4 -dao /var/tmp/backup.img    ← イメージ・ファイル（backup.img）をCD-R/RWにコピー（コマンドの詳細は例3を参照）
```

Column 2

ATA接続のMOやCD-Rを利用する

Linuxは、WindowsとはATA[*1]接続のMOやCD-Rを直接利用することができない。そのため、LinuxにはATA接続の機器をSCSI接続に見せかける「SCSIエミュレーション」という仕組みが備わっている。SCSIエミュレーションでは、「ide-scsi」という特別なプログラム（モジュール）を仮想的なSCSIホスト・アダプタ[*2]として利用する。

ここでは、デバイス「hdc」にCD-RW、「hdd」にMOドライブが接続されているシステムを例に、ATA接続の機器をSCSI接続に見せかける方法を説明しよう。

一般的なディストリビューションのカーネルでは、最初からSCSIエミュレーションの機能がモジュールとして提供されている[*3]。それを有効にするには、オプション「ide-scsi」を指定してカーネルを起動すればよい。カーネルにオプションを渡すには、ブート・ローダの設定ファイルを**例A**のように変更する。

また、実際にCD-RやMOをSCSI機器として利用するときは、それらの機器への接続があった時点でモジュール（ide-scsi）が読み込まれる必要がある。そのため、「/etc/modules.conf」ファイルに、**例B**のような設定を追加しておく。ここで「scd0」とはCD-RWのSCSI機器としてのデバイス名であり、「sda」はMOのSCSI機器としてのデバイス名である。

この設定が完了したら、システムを再起動してみよう。そうすれば、デバイス名「/dev/scd0」でCD-RWに、「/dev/sda」でMOにアクセスできるはずである。

なお、接続されているATA機器のデバイス名がわからない場合は、dmesgというコマンドを実行してみるとよい。**例C**は筆者のシステムでdmesgを実行した例だ。これを見れば、デバイス名「hdc」と「hdd」がCD-ROMに割り当てられていることがわかるだろう（実際には、hdcがCD-RW、hddがPD）。

例A：ブート・ローダの設定

●liloの場合（lilo.confファイル）

```
image=/boot/vmlinuz-2.4.18-14
      label=linux
      initrd=/boot/initrd-2.4.18-14.img
      read-only
      append="root=LABEL=/ hdc=ide-scsi hdd=ide-scsi"
```

← hdcとhddに「ide-scsi」というオプションを指定してSCSIエミュレーションを有効にする

●grubの場合（grub.confファイル）

```
title Red Hat Linux (2.4.18-18.8.0)
      root (hd0,0)
      kernel /vmlinuz-2.4.18-18.8.0 ro root=LABEL=/ hdc=ide-scsi hdd=ide-scsi
      initrd /initrd-2.4.18-18.8.0.img
```

← hdcとhddに「ide-scsi」というオプションを指定してSCSIエミュレーションを有効にする

*デバイス・ファイル「hdc」にCD-RW、「hdd」にMOドライブが接続されているシステムを想定。どちらの例も、hdcおよびhddに対してSCSIエミュレーションを適用するようカーネルに指示している。

例B：「/etc/modules.conf」の設定例

```
alias scsi_hostadapter ide-scsi   ← SCSIホスト・アダプタとして「ide-scsi」を利用する指示
alias scd0 sr_mod                 ← CD-RW利用時にSCSIモジュール「sr_mod」をメモリに読み込ませるという指示
alias sda  sr_mod                 ← MO利用時にSCSIモジュール「sr_mod」をメモリに読み込ませるという指示
```

*SCSI接続のCD-ROM/CD-R/RWのデバイス名は「scdX」（Xは0、1、2……）となり、ハードディスクは「sdY」（Yはa、b、c……）となる。MOはマウント時にハードディスクとして扱われる点に注意。

例C：dmesgの出力例（一部抜粋）

```
hda: IC35L060AVVA07-0, ATA DISK drive          ← ハードディスクのデバイス名はhda
hdb: ST310230A, ATA DISK drive                 ← ハードディスクのデバイス名はhdb
hdc: SONY CD-RW CRX145E, ATAPI CD/DVD-ROM drive   ← CD-ROMドライブのデバイス名はhdc
hdd: MATSHITAPD-1 LF-1196, ATAPI CD/DVD-ROM drive ← CD-ROM（実際はPD）ドライブのデバイス名はhdd
```

*1：ATAは、AT Attachmentの略。PCのハードディスク／CD-ROMへ接続するための規格。一般的なPCのハードディスクやCD-ROMは、ATAに従って接続される。なお、ATAは一般的には「IDE（Integrated Drive Electronics）」とも呼ばれる。

*2：マシンに接続されたSCSI機器を制御するためのハードウェア。「SCSIカード」の同義語と考えてよい。

*3：自分でカーネルをコンパイルしていてSCSIエミュレーション機能が組み込まれていない場合は、SCSIエミュレーションを組み込んでからカーネルを再コンパイルする必要がある。

Part 6 テープ・ヘッドの移動操作やファイルの参照方法を知る

磁気テープの基本操作

磁気テープには、他の記憶装置とは大きく異なる点がある。1つは、マウント操作を行わずにデバイス・ファイルを直接扱える点。もう1つは、送りや巻き戻しといったテープ独特の操作を覚える必要がある点だ。

テープの構造

図1に、磁気テープの構造を示す。テープの先頭には先頭を示すBOT（Beginning of Tape）というマークが記される。また、テープに格納される各ファイルの最後にはEOF（End of File）というマークが記される。テープに格納されているファイルを参照するときは、これらのマークを手がかりに「データが格納されている位置までテープを送り」「実際にデータを読み出す」という2段階の処理を行うことになる（操作方法は後述）。

テープの操作コマンド —mt—

Linuxで磁気テープを操作するには、mtというコマンドを利用する。mtの基本的な実行形式は以下のとおりだ。

```
mt -f テープ・デバイス名 操作
```

「-f」オプションによって操作対象テープのデバイス・ファイル(*1)を指定する。「-f」オプションを指定しないと、mtは「/dev/tape」というデバイスを操作対象とする。したがって、ふだん利用するテープのデバイスへのシンボリック・リンクを「/dev/tape」という名前で作成しておけば、いちいち「-f」オプションでデバイス名を指定する必要がなくなるので便利である。

仮に、ふだん利用するテープが「/dev/nst0」であれば、「ln -s /dev/nst0 /dev/tape」というコマンドを実行して、/dev/nst0へのシンボリック・リンク（/dev/tape）を作成しておけばよい(*2)。

これにより、mtは以下の形式で実行できるようになる。

```
mt 操作
```

なお、上の「操作」の部分には、テープに対して行う実際の処理を指定する。**表1**に、mtの操作として利用できる主な引数を示すので参照されたい。

テープ内のtarアーカイブの内容を参照する

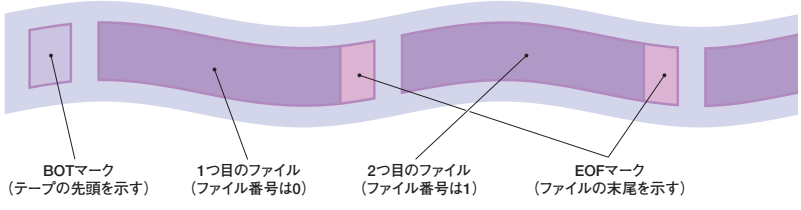
それでは、磁気テープの操作方法を説明しよう。

まず、引数に操作「status」を指定してmtコマンドを実行すれば、現在のテープ・ヘッドの位置を調べられる（次ページの**例1**-①）。例1-②を見ると、テープ・ヘッ

*1：SCSI接続のテープのデバイス・ファイルは、操作モードに応じて2種類用意されている。1つは、操作完了時に自動的に巻き戻しを行いたいときに利用する「/dev/st0、1、2……」、もう1つは、操作完了時に自動巻き戻しを行わないときに利用する「/dev/nst0、1、2……」というもの（IDE接続のテープのデバイス・ファイル名は「/dev/ht0、1、2……」となる）。

*2：mtは、実行時に環境変数TAPEの設定を参照する。そこで、シンボリック・リンクを作成する代わりに「export TAPE=/dev/nst0」と実行して、環境変数TAPEに「/dev/nst0」をセットしてもよい。

図1：磁気テープに書き込まれるファイル・データ



＊各ファイル・データの最後には、EOFマークが自動的に付与される。

表1：mtコマンドで利用可能な主な操作（nは数値を示す）

コマンドラインに指定する文字	操作
status	テープの状態を表示する
rewind	テープを先頭まで巻き戻す
retension	テープのたるみを取る
offline	テープ・ドライブからテープを排出する
compression n	ハードウェア圧縮（テープ・ドライブの圧縮機能）の有効／無効を切り替える（0：無効、1：有効）
fsf n	n個先のEOFまで、テープを早送りする
bsf n	n個前のEOFまで、テープを巻き戻す
bsfm n	n個前のファイルの先頭まで、テープを巻き戻す

ドはテープの先頭（BOT）にあることがわかる。ここで「tar tf デバイス・ファイル名（/dev/nst0）」と実行すると、テープの先頭位置のデータ（ファイル番号0のデータ）の内容が表示される。

仮に、このテープにtarアーカイブが3つ収められているとしよう。2番目のデータを参照したいときは、テープ・ヘッドを2つ目のデータの先頭に送らなければならない。そのためには、引数に操作「fsf 1」を指定してmtコマンドを実行する（例1-③）。

例1-③の実行後に再び操作「status」でテープの状態を見てみると、ヘッドの位置は「EOF」、つまりファイル番号0の最後にあることが確認できる（例1-④）。

この時点でtarを実行すると、ファイル番号0の次に書かれたデータ（ファイル番号1のデータ）が表示される（例1-⑤）。

例1：mtの使用例（テープの操作）

```
$ mt -f /dev/nst0 status                    ← 現在のテープ・ヘッドの位置を表示する ①
SCSI 2 tape drive:
File number=0, block number=0, partition=0. ← 次に読み出されるデータはファイル番号0のデータ
Tape block size 0 bytes. Density code 0x49 (no translation).
Soft error count since last status=0
General status bits on (41010000):
 BOT ONLINE IM_REP_EN                       ← 「BOT」と表示されているということは、現在位置はテープの先頭であるということ ②
$ mt -f /dev/nst0 fsf 1                     ← テープの位置を1データ分先に送る（fsf 1を指定） ③
$ mt -f /dev/nst0 status                    ← 再度、テープの状態を表示
SCSI 2 tape drive:
File number=1, block number=0, partition=0. ← 次に読み出されるデータはファイル番号1のデータ
Tape block size 0 bytes. Density code 0x49 (no translation).
Soft error count since last status=0
General status bits on (81010000):
 EOF ONLINE IM_REP_EN                       ← 「EOF」と表示されているということは、直前のファイル（ファイル番号0）の終端にいるということ ④
$ tar tf /dev/nst0                          ← tarでファイル番号1のデータを表示する ⑤
……中略……
$ mt -f /dev/nst0 status                    ← 表示後（読み出し完了後）の状態を確認する
SCSI 2 tape drive:
File number=1, block number=46, partition=0. ← ファイル番号1のブロック番号（テープ・ヘッドの位置を示す番号）は46（ファイル番号1のデータの読み出しが完了し、ブロック番号が0から46まで進んだ） ⑥
Tape block size 0 bytes. Density code 0x49 (no translation).
Soft error count since last status=0
General status bits on (1010000):
 ONLINE IM_REP_EN                           ← データの読み出しが完了したのに「EOF」が表示されていない ⑦
$ mt -f /dev/nst0 fsf 1                     ← テープの位置を1つ送る ⑧
$ mt -f /dev/nst0 status
SCSI 2 tape drive:
File number=2, block number=0, partition=0. ← 次に読み出されるデータはファイル番号2のデータ
Tape block size 0 bytes. Density code 0x49 (no translation).
Soft error count since last status=0
General status bits on (81010000):
 EOF ONLINE IM_REP_EN                       ← 「EOF」と表示されているということは、直前のファイル（ファイル番号1）の終端にいるということ
```

2つ目のtarアーカイブを参照するには

tarでファイル番号1のデータを読み出した後、再びテープの状態を確認すると、例1-⑥のようにヘッドの位置を示すブロック番号が0から46まで進んでいる。さらに、ここで注目していただきたいのは例1-⑦だ。ご覧のように、データの末尾であることを示す「EOF」が表示されていない。

これは、tarがテープ上の「EOF」ではなく、アーカイブ内に書き込まれている「EOF」を検知してデータの読み出しを終了したためだ。テープ上の「EOF」とアーカイブ内の「EOF」はまったく別のものであり、この段階でテープ・ヘッドはテープ上の「EOF」までは移動していない。したがって、再度tarを実行しても何も表示されない。ファイル番号1のデータはすでに読み出しが終了しており、なおかつ、テープ・ヘッドがテープ上の「EOF」まで移動していないからだ。

次のデータ（ファイル番号2のデータ）を参照するには、ファイル番号1のデータのテープ上の「EOF」まで移動しなければならない。そのためには、例1-⑧のように再度「fsf 1」を指定してmtコマンドを実行すればよい。

運用上の注意点

以上、磁気テープの操作方法について簡単に説明した。表1に示したmtの操作を利用すれば、テープの送りや巻き戻しを行える。実際のバックアップとリストアの方法については、パート3やパート7を参照していただきたい。

なお、テープの巻き戻しや送りといった操作は、「/dev/st0」のように自動巻き戻しを行うデバイスに対しては行えない。したがって、1本のテープに複数のデータを書き込むときは、「/dev/nst0」といった自動巻き戻しを行わないデバイス・ファイルを利用する。

また、磁気テープはハードディスクとは異なり、実際にテープ・ヘッドがテープの表面に接触することでデータを読み出す。そのため、テープ・ヘッドは定期的にクリーニングする必要がある。各テープ・ドライブ・メーカーから、そのドライブに適合するクリーニング・キットが提供されているので、そうしたキットを入手して利用されたい。

このほか、テープは高温や湿度に非常に弱い。大切なバックアップ・データを守るためにも、テープの保管場所には十分に気を配る必要がある。

Part 7 フリーの高機能バックアップ・ツール

Amandaの活用法

**Amandaは、フリーでありながら非常に高機能なバックアップ・ツールだ（対象メディアは磁気テープ）。
高機能なだけに使い方は多少複雑だが、大規模サイトから1台のサーバまで、どのような局面にも対応できる数少ないツールだと言えよう。**

テープを対象としたバックアップ・ツール

Amanda（The Advanced Maryland Automatic Network Disk Archiver）は、米国メリーランド大学で開発された高機能なバックアップ・ツールだ。その特徴は、クライアント／サーバ形式で動作し、テープ・ドライブへのバックアップを行えること。また、テープ・チェンジャ(*1)のサポート、データ圧縮機能、フル・ダンプ（フル・バックアップ）の自動スケジューリング機能、テープ管理機能など、フリーのバックアップ・ツールとしては非常に高度な機能を備えている。

本パートでは、「1日1回のフル・バックアップをテープ・ドライブに取得する」ケースを想定して、Amandaの活用法を説明する。

Amandaのインストール

Red Hat Linuxなどのディストリビューションには、Amandaのrpmパッケージが用意されている。そのためAmandaは、ディストリビューターのWebサイトや各ディストリビューションのインストールCDからrpmコマンドでインストールできる。

レッドハットは、**表1**に示す4つのrpmパッケージを用意しているが、Amandaを利用する際は、amanda-devel以外のパッケージをインストールすればよい（**例1**）。

インストール後には、「/etc/xinetd.d/amanda」および「/etc/xinetd.d/amandaidx」ファイルを編集して、両サーバがスーパー・デーモンのxinetd経由で起動するように設定する。具体的には、両ファイル中の「disable=yes」という行を「disable=no」に変更すればよい。

なお、Amandaのソース・コードをコンパイルしてインストールする場合は、Amandaの公式Webサイト（http://www.amanda.org/）から、最新のソースを入手すればよい。原稿執筆時点の最

*1：複数のテープを格納してテープを交換しながらバックアップを行えるようになっている機器。

表1：Red Hat Linux 8.0に含まれるAmandaのパッケージ

パッケージ名	内容
amanda-2.4.2p2	Amandaの基本パッケージ
amanda-client-2.4.2p2	Amandaのクライアント・コンポーネント
amanda-server-2.4.2p2	Amandaのサーバ・コンポーネント
amanda-devel-2.4.2p2	Amandaの開発用パッケージ

例1：Amandaのrpmパッケージからのインストール方法（3つのパッケージをインストール）

```
$ wget ftp://ftp.redhat.com/pub/redhat/linux/8.0/ja/os/i386/RedHat/RPMS/amanda-2.4.2p2-9.i386.rpm          ← amandaをダウンロード
$ wget ftp://ftp.redhat.com/pub/redhat/linux/8.0/ja/os/i386/RedHat/RPMS/amanda-client-2.4.2p2-9.i386.rpm  ← amanda-clientをダウンロード
$ wget ftp://ftp.redhat.com/pub/redhat/linux/8.0/ja/os/i386/RedHat/RPMS/amanda-server-2.4.2p2-9.i386.rpm  ← amanda-serverをダウンロード
$ su                                                                                                       ← rootになる
Password:                                                                                                  ← rootのパスワードを入力
# rpm -ivh amanda-2.4.2p2-9.i386.rpm                                                                       ← amandaをインストール
# rpm -ivh amanda-client-2.4.2p2-9.i386.rpm                                                                ← amanda-clientをインストール
# rpm -ivh amanda-server-2.4.2p2-9.i386.rpm                                                                ← amanda-serverをインストール
```

*amanda-serverをインストールするには、あらかじめgnuplot（グラフ作成ツール）というパッケージがインストールされている必要がある。gnuplotがインストールされていないときは、まず、wgetなどでgnuplot-3.7.2-1.i386.rpmをダウンロードし、インストール（「rpm -ivh gnuplot-3.7.2-1.i386.rpm」を実行）しておく。

新版（安定版）は2.4.3であった。ソースのインストール方法については、**例2**を参考にしていただきたい。

設定ファイルの準備

Amandaの設定ファイルは、バックアップ時の動作などを設定するamanda.confと、バックアップの領域を指定するdisklistの2つに分かれている。また、Amandaは両方のファイルを格納したディレクトリを1つの設定とみなす構造になっている。このディレクトリの名称が「設定名」となり、Amandaの各コマンドは「設定名」を基に動作する。

rpmパッケージからAmandaをインストールした場合、/etc/amandaディレクトリの下にDailySet1というディレクトリが作成される。その中にamanda.confとdisklistが存在する。つまり、DailySet1ディレクトリが「設定名」となるわけだ。ここでは、このディレクトリを設定の雛型として利用する方法を説明する。

また、Amandaをソースからインストールした場合は、ソース中のexamplesというディレクトリの下にあるサンプルを利用して、設定の雛型（DailySet1ディレクトリ）を作成する（**例3**）。

amanda.confの設定

バックアップに関する一般的な設定は、前述したようにamanda.confで行う。amanda.confにはさまざまな設定項目が用意されているが、通常の用途であれば**例4**の項目を設定すれば十分だろう。

以下、amanda.confの設定でポイントとなる点を説明しておこう。

dumpcycleとrunspercycle

例4-①の「dumpcycle」と「runspercycle」項目には、それぞれ「フル・ダンプ（フル・バックアップ）を実行するサイクル」

例2：Amandaのソースからのインストール方法

```
①ユーザー「amanda」を作成
# useradd -g disk -c "Amanda user" amanda

②コンパイルとインストール
Amandaのサイト（http://www.amanda.org/）から最新のソース（今回の場合、amanda-2.4.3.tar.gz）をダウンロード
$ tar zxf amanda-2.4.3.tar.gz          ← ソース・アーカイブを展開
$ cd amanda-2.4.3/                     ← amanda-2.4.3ディレクトリに移動
$ ./configure --with-user=amanda --with-group=disk   ← configureスクリプトでMakefileを作成（Amandaを起動するユーザーをamanda、そのグループをdiskに設定）
$ make                                 ← 作成したMakefileに従ってコンパイル
# make install                         ← インストール
# install -d -o amanda -g disk /usr/local/var/amanda/gnutar-lists   ← 所有者をamanda、所有グループをdiskとして、ディレクトリ/usr/local/var/amanda/gnutar-listsを作成

③「/etc/services」ファイルに以下の登録を追加
amanda          10080/tcp            # amanda backup services
amanda          10080/udp            # amanda backup services

④「/etc/xinetd.d/amanda」ファイルを以下の内容で作成
service amanda
{
    socket_type     = dgram          ← ソケットの種類を指定。種類にはストリーム型（stream）とデータグラム型（dgram）がある。Amandaの場合はdgramを指定
    protocol        = udp            ← プロトコルをudpに指定
    wait            = yes            ← 複数の接続要求を同時に受け付けるかどうかを設定。yesは、複数の接続を受け付けないという意味
    user            = amanda         ← Amandaを立ち上げるユーザーの名前（①のconfigureの実行時にamandaとした）
    group           = disk           ← ユーザー「amanda」が属するグループ（①のconfigureの実行時にdiskとした）
    server          = /usr/local/libexec/amandad   ← amandad（Amandaのデーモン）へのパス
    disable         = no             ← xinetd経由でサービス（Amanda）を起動するかしないかを設定。起動させたい場合は「no」と記述する
}

⑤xinetdを再起動
# service xinetd restart

⑥次の内容で「/home/amanda/.amandahosts」というファイルを作成
localhost amanda

⑦「/etc/amandates」ファイルを作成
# touch /etc/amandates
# chown amanda.disk /etc/amandates   ← 「/etc/amandates」ファイルの所有者をamandaに、所有グループをdiskに設定
```

例3：設定の雛型を作成（Amandaをソースからインストールした場合の作業）

```
# mkdir /usr/local/etc/amanda/DailySet1
# cd amanda-2.4.3/example
# cp cp amanda.conf disklist /usr/local/etc/amanda/DailySet1
# chown -R amanda.disk /usr/local/etc/amanda
```

- mkdir … DailySet1ディレクトリの作成
- cd … amanda-2.4.3/exampleディレクトリへ移動
- cp … 「amanda.conf」と「disklist」の両ファイルをDailySet1ディレクトリにコピー
- chown … /usr/local/etc/amanda以下のすべてのファイル／ディレクトリの所有者をamandaに、所有グループをdiskに設定

例4：「amanda.conf」ファイルの設定例（一部抜粋）

```
org "DailySet1"
mailto "amanda root"
dumpuser "amanda"
tapedev "/dev/nst0"
dumpcycle 1 days
runspercycle 1 days
tapecycle 2 tapes
tapetype SDLT
labelstr "^DailySet1[0-9][0-9]*$"
infofile "/var/lib/amanda/DailySet1/curinfo"
logdir   "/var/lib/amanda/DailySet1"
indexdir "/var/lib/amanda/DailySet1/index"

define tapetype SDLT {
    comment "SDLT tape drives"
    length 160000 mbytes
    filemark 2000 kbytes
    speed 1536 kbytes
}

holdingdisk hd1 {
    comment "main holding disk"
    directory "/var/tmp/amhold"
    use 500 Mb
    chunksize 500 Mb
}

define dumptype global {
……中略……
}

define dumptype basic-tar {
    global
    program "GNUTAR"
    comment "etc dumped with tar"
    compress none
}
define dumptype basic-dump {
    global
    program "DUMP"
    comment "home partitions dumped with dump"
    compress none
    index
}
```

- org … Amandaからの報告メールの表題
- mailto … 報告メールの宛先（複数指定する場合はスペースで区切る）
- dumpuser … バックアップを行うユーザー
- tapedev … 利用するテープ・デバイス（巻き戻しを行わないデバイス「/dev/nst0」を指定）
- dumpcycle … ダンプ（バックアップ）のサイクル（1日1回）
- runspercycle … 1サイクル当たりのバックアップの実行回数（1回）
- ①
- tapecycle … 1回のバックアップで必要なテープの本数（直前のフル・バックアップ・データを上書きしないようにするため、2とする）
- ②
- tapetype … テープ・メディアの種類（下の「define」項目で定義）
- ③
- labelstr … どのようなラベルのテープの利用を許可するかを設定。「^DailySet1[0-9][0-9]*$」は、「DailySet1」で始まり、次に「0〜9」の数値が2つ連続するというパターン（例えば、DailySet101）のラベルを意味する
- infofile … 履歴データベースの配置場所
- logdir … ログの出力場所
- ④
- indexdir … バックアップ内容がインデックス化されたカタログの格納場所。⑧のように「index」を指定した場合、このディレクトリにカタログが格納される
- define tapetype SDLT { … テープ・メディアの種類を定義
- comment … コメント文
- length … 非圧縮時のテープ容量
- filemark … ファイル・マークの長さ(注1)
- speed … 1秒当たりの転送速度
- ⑤
- holdingdisk hd1 { … Amanda用ディスク領域の設定
- comment … コメント文
- directory … バックアップ用ディスク領域（保持領域）の場所
- use … 保持領域のサイズ
- chunksize … 保持領域を超えるデータは500MBに分割するという意味
- ⑥
- define dumptype global { …… } … 下に示す各dumptype項目で共通利用される定義を記述(注2)
- define dumptype basic-tar { … バックアップ方法の定義（tarコマンド）
- global … このdumptypeで共通利用される設定（global）を取り込む
- program … バックアップに利用するコマンド（ここではtarを指定）
- comment … コメント文（/etcの内容はtarでバックアップするという意味。その設定は例5のdisklistファイルで行う）
- compress none … バックアップ時に圧縮は行わない
- ⑦
- define dumptype basic-dump { … バックアップ方法の定義（dumpコマンド）
- global … このdumptypeで共通利用される設定（global）を取り込む
- program … バックアップに利用するコマンド（ここではdumpを指定）
- comment … コメント文（パーティションはdumpでバックアップするという意味。その設定は例5のdisklistファイルで行う）
- compress none … バックアップ時に圧縮は行わない
- index … バックアップ内容のカタログを作成する
- ⑧

注1：ファイル・マークとは各データ・ブロックの最後に配置される特殊な領域。データ・ブロックの区切りを明確にする目的で配置される。
注2：仮に、ここに「compress none」と指定すると、⑦と⑧の両方に「圧縮を行わない」という設定を行ったことになる。
*なお、例4に記した各項目のほか、サンプルのamanda.confに含まれている「tpchanger」「rawtapedev」「changerdev」という3項目は、コメントアウトしておく（行頭に「#」記号を付けて設定を無効にする）必要がある。

と「1サイクル当たりの実行回数」を定義する。今回は「1日1回、必ずフル・ダンプを行う」というスケジュールとするため、両項目の値を「1」としている。

tapecycle

例4-②の「tapecycle」項目には、1回のサイクルで必要なテープの本数を設定する。この項目には、「1サイクル当たりの実行回数×1回のバックアップに必要なテープ本数＋1」という式の値を設定すればよい。最後の「＋1」は、最新のフル・バックアップによって直前のフル・バックアップ・データを上書きしないようにするためだ。今回は、1回のバックアップに必要なテープ本数が1本であることを想定している。そのため、テープ本数の部分は「2」となる。

tapetype

例4-③の「tapetype」項目には、バッ

クアップで利用するテープ・メディアの種類を指定する。

例4では、テープ・メディアの仕様から判断した種類（SDLT）を設定している（今回は、テープ・ドライブとしてタンベルグデータのSuper DLTテープ・ドライブ「SDLT320」を使用している。そこで、メディアの種類をSDLTとした）。ここに指定する種類は、別途「define」という項目に「tapetype」という項目を立てて、設定ファイル中に指定しておかなければならない（例4-⑤）。

labelstr

例4-④の「labelstr」項目には、テープのラベルというものを指定する。Amandaでは、テープの管理を行う関係上、Amandaのサブコマンドであるamlabelによってラベルが設定されたテープしか利用できない。そのため、amanda.confには「どのようなラベルのテープの利用を許可するか」を指定する。

ここで指定している「^DailySet1[0-9][0-9]*$」は、「DailySet1」で始まり、次に「0〜9」の数値が2つ連続するという文字パターンだ。これによりAmandaでは、そのパターンに当てはまるラベルが設定されたテープにのみバックアップを行えるようになる（後ほどラベルを作成する際に、このパターンに当てはまる名前とする）。

holdingdisk

Amandaは、データをテープにコピーする前に、いったんディスクにコピーする。例4-⑥の「holdingdisk」項目にはそのディスク領域に関する設定を行う。具体的には、「use」項目にAmandaが利用する領域（保持領域）の最大利用可能サイズを指定する。また、「chunksize」項目にはその保持領域を超えるサイズのデータが書き出されたときの処理を指定する。

「chunksize」が負（マイナス）の値の場合、保持領域を超えるサイズのデータは保持領域を利用せず、直接テープに書き出すという設定となる。一方、正（プラス）の値の場合は、保持領域に収まるようデータを分割するという設定となる（分割した場合でも、テープにコピーされる際は1つのデータとなるので心配はない）。なお、保持領域にはテープにコピーする前のバックアップ・データが格納されるため、セキュリティ上、ユーザーamandaだけがアクセス可能なように設定しておいたほうがよいだろう。

dumptype

例4-⑦/⑧の「dumptype」項目には、バックアップの実行方法を定義する。初期設定のamanda.confには、最初からいくつかの「dumptype」が定義されている。その中に気に入った実行方法がなければ、自分で新しく定義することもできる。例4では⑦と⑧の2つの「dumptype」を定義している（詳細は例4の説明を参照）。

disklistの設定

次に、もう1つの設定ファイルであるdisklistについて説明しよう。disklistには、バックアップ対象とするホスト、バックアップ領域、バックアップ方法（例4-⑦/⑧に連動）を指定する。

例5にdisklistの設定例を示す。ここでは、/etcディレクトリと「/dev/hda1」パーティションのバックアップを指示している。「/dev/hda1」のバックアップはパーティション単位となるため、dumpコマンドの利用を指示している。一方、/etcはパーティションとして独立していないため、tarコマンドの利用を指示している。

ラベルの設定

amanda.confとdisklistの設定が完了したら、バックアップに利用するテープにラベルを付ける。前述したように、Amandaは、例4-④で設定したラベルにマッチするテープにしかバックアップを実行できない。テープにラベルを付けるには、テープをドライブにセットした後、例6のようにしてamlabelコマンドを実行すればよい。

ラベルを設定したら、そのテープをテープ・ドライブに入れたまま設定ファイルに誤りがないかどうかをチェックしてみよう。設定の確認は、amcheckというコマンドで行える（例7）。例7を実行して「ERROR」という文字が表示された場合は、設定に何らかの間違いがあるので修正する必要がある。また、ラベルを付けたテープがドライブに入っていないとエラーとなるので注意されたい。

バックアップの方法

以上の設定が完了したら、実際にAmandaでバックアップを実行してみよう。そのためには、テープをドライブにセ

ットした後、「amdump 設定名」というコマンドをamandaユーザーで実行すればよい（**例8**）。

バックアップの結果は、amanda.confで指定したユーザーに電子メールで報告される。また、例4-⑧のように「dumptype」項目で「index」と指定した場合、バックアップの内容がインデックス化されたファイル（カタログ）も作成される。このカタログは、例4の「indexdir」項目で指定したディレクトリに格納される。

例8を実行し、手動でバックアップを取れることが確認できたら、そのコマンドをcronに登録してみよう。そのためには、まず**例9**のようなファイルを作成し（ファイル名は「amanda」などとする）、そのファイルを/etc/cron.dディレクトリに格納すればよい。例9では、毎日午前3時1分から、DailySet1の設定に従って自動バックアップが実行されるようになる。

リストアの方法

Amandaで取得したバックアップは、amrestoreというコマンドでリストアすることができる。

例10にその実行例を示す。例10-①では、amrestoreコマンドの引数にホスト名「localhost」のみを指定している。この場合、自ホストの全データがリストアされる。また例10-②と③のamrestoreコマンドには、ホスト名に加えてバックアップ領域（パーティション）や日付が指定されている。このように、バックアップ・データをリストアする際には、システムに戻すデータを絞り込むことが可能である。

amrestoreでリストアを行うと、amrestoreを実行したディレクトリに「ホスト名.領域名.日付.ダンプ・レベル」という形式の名前のファイルが作成される。そのファイルは、amanda.confの設定に応じてtarやdump形式となる。例4-⑦のようにバックアップ方法に「GNUTAR」が指定されているバックアップ領域はtar形式、例4-⑧のように「DUMP」が指定されている領域はdump形式となるわけだ。

「ホスト名.領域名.日付.ダンプ・レベル」という形式のファイルが作成されたら、あとは、restoreやtarなどのコマンドで個別にリストアを行えばよい（restoreやtarによるリストアの方法についてはパート3を参照）。

例5：disklistの設定例

```
localhost /etc      basic-tar     ← /etcディレクトリのバックアップを指示
localhost /dev/hda1 basic-dump    ← 「/dev/hda1」パーティションのバックアップを指示
```

- localhost：バックアップ対象のホスト名（ここでは自ホストであるlocalhostを指定）
- /etc, /dev/hda1：バックアップ領域。tarを利用する場合はディレクトリ名を指定可能。dumpの場合は必ずデバイス名を指定する
- basic-tar, basic-dump：例4-⑦/⑧で定義したバックアップ方法

例6：amlabelによるラベルの設定

```
# su amanda -c "amlabel -f DailySet1 DailySet101"
```

実行ユーザー／ラベル設定用のコマンド／Amandaの設定名／ラベル名（DailySet1にDailySet101というラベルが付与される）

＊ラベルの設定は、例4-②の「tapecycle」で指定したテープの本数分行う。1本目のテープには「DailySet101」、2本目のテープには「DailySet102」といった具合に、用意した個々のテープに異なったラベルを付与する。

例7：amcheckの実行例

```
# su amanda -c "amcheck DailySet1"
```

実行ユーザー／設定確認用のコマンド／DailySet1の設定を確認

＊Amandaをソースからインストールした場合、amcheckは/usr/local/sbinに置かれている。

例8：Amandaによるバックアップの実行例

```
# su amanda -c "amdump DailySet1"
```

実行ユーザー／バックアップ用のコマンド／設定名（DailySet1の設定内容に沿ってバックアップが実行される）

＊Amandaをソースからインストールした場合、amdumpは/usr/local/sbinに置かれている。

例9：cronによるバックアップの自動実行の設定

```
$ cat amanda                                  ← cronに登録するファイルをamandaという名前で作成
01 3 * * * amanda /usr/sbin/amdump DailySet1
```

毎日3時1分に／ユーザー「amanda」が／DailySet1の設定に沿ってバックアップを実行する

例10：バックアップ・データのリストアの実行例

①localhostのデータをすべてリストアする場合

```
# su amanda -c "amrestore /dev/nst0 localhost"
```

実行ユーザー／リストア用のコマンド／テープのデバイス名／ホスト名（このホストからデータをリストアする）

②localhostの「/dev/hdb1」パーティションのデータをリストアする場合

```
# su amanda -c "amrestore /dev/nst0 localhost /dev/hdb1"
```

実行ユーザー／リストア用のコマンド／テープのデバイス名／ホスト名（このホストからデータをリストアする）／パーティション

③localhostの「/dev/hdb1」パーティションの中から2003年2月25日のデータをリストアする場合

```
# su amanda -c "amrestore /dev/nst0 localhost /dev/hdb1 20030225"
```

実行ユーザー／リストア用のコマンド／テープのデバイス名／ホスト名（このホストのデータをリストアする）／パーティション／日付

＊Amandaをソースからインストールした場合、amrestoreコマンドは/usr/local/sbinに置かれている。

Practice バックアップ&リストアの実践ノウハウ

演習問題

問題 1 サーバのバックアップを行う際、バックアップを取る必然性が最も低いディレクトリは以下のうちどれか。

①/bin ②/boot ③/var/log ④/tmp ⑤/home

問題 2 restoreコマンドのオプションで、データ全体を復旧させる際に利用するものは以下のうちどれか。

①r ②x ③b ④u ⑤i

問題 3 日曜日にフル・バックアップ、月曜日から土曜日に差分バックアップというバックアップの運用で、木曜日の日中にディスクがクラッシュした。このとき、復旧に必要となるバックアップ・データは以下のうちどれか。なお、バックアップは夜間に行っているものとする。

①水曜日のバックアップ・データのみ
②日曜日と水曜日のバックアップ・データ
③日曜日、月曜日、火曜日、水曜日のバックアップ・データ
④日曜日のバックアップ・データのみ
⑤完全な復旧は不可能

問題 4 mtコマンドによるテープ操作で、テープを取り出すための「操作」は以下のうちどれか。

①restore ②fsf ③offline ④deposit ⑤retension

問題 5 Amandaのコマンドの中で、リストアに利用するものは以下のうちどれか。

①amrescue ②amcheck ③amdump ④amextract ⑤amrestore

解答

[問題1] 正解は④。/tmpに格納されているデータはすべて一時的に利用するデータであり、バックアップを取る必然性は極めて低い。
[問題2] 正解は①。「r」は、アーカイブに含まれるすべてのデータをリストアするためのオプション。個別のファイルをリストアする際には「x」、対話的にデータを復旧させる際は「i」を利用すればよい。
[問題3] 正解は②。差分バックアップでは、最後のフル・バックアップからの差分をすべて取得する。そのため、最後のフル・バックアップと最後に取得した差分バックアップがあればデータを復旧できる。
[問題4] 正解は③。mtを利用してテープを取り出すには、「mt -f /dev/nst0 offline」のように操作「offline」を指定してmtを実行する。
[問題5] 正解は⑤。Amandaでデータをリストアする際はamrestoreコマンドを実行する（バックアップはamdump）。