LinuC

Linuxシステム管理標準教科書

（Ver.1.0.1）

LPI-JAPAN

目次

まえがき

このたび、特定非営利活動法人エルピーアイジャパンは、Linux 技術者教育に利用していただくことを目的とした教材、「Linux システム管理標準教科書」を開発し、インターネット上にて公開し、提供することとなりました。

この「Linux システム管理標準教科書」は、既に公開し、多くの教育機関や学習者に活用されている「Linux 標準教科書」「Linux サーバー構築標準教科書」の続編として作成されました。本教科書では、Linux を扱うシステム管理者が知っておきたい運用管理の基本を、実習を交えて理解することを目的にしています。

公開にあたっては、「Linux システム管理標準教科書」に添付されたライセンス（クリエイティブ・コモンズ・ライセンス）の下に公開されています。

本教科書は、最新の技術動向に対応するため、随時アップデートを行っていきます。また、テキスト作成やアップデートについては、意見交換のメーリングリストで、誰でもオープンに参加できます。詳しくは Linux システム管理標準教科書のホームページをご覧ください。

Linux システム管理標準教科書 https://linuc.org/textbooks/admin/

執筆者・制作者紹介

平愛美（1 章〜3 章執筆担当）

Linux は進化を続けており、システムを管理する仕組みも日々変化しています。最近で言えば、systemd の登場が大きな変化と言えるでしょう。この教科書も、最新の Linux システムに追従できるように systemd の説明を行っています。新しい仕組みに触れるきっかけ作りになれば幸いです。

面和毅（4 章〜6 章執筆担当）

Linux も一般的なシステム用の OS として広く使われるようになったため、導入する側も管理する側も同じレベルで「運用に対する基礎知識」が必要とされています。この教科書で、まずは基礎レベルを抑えていただき、運用を考えていく際のベースとしていただければ幸いです。

宮原徹（7 章執筆および全体調整担当）

「Linux 標準教科書」「Linux サーバー構築標準教科書」と合わせて、「標準教科書三部作」が完成しました。しかし、この教科書でも説明しきれていない事柄がまだまだ残っています。今後も教科書の改訂や新しい教科書の企画などで皆さんと情報を共有できればと思います。

松田神一（レビュー担当）

Linux 関連の技術の進歩は著しく、現在でも日々、新しい機能が追加され続けています。この教科書をレビューしている時も、技術の変化を実感しました。実際に運用する対象の OS は、最新版のこともあれば、旧バージョンのこともあるでしょうし、Red Hat 系以外のディストリビューションのこともあると思います。この教科書で運用の基礎や考え方を学んだ上で、OS のディストリビューションやバージョンによる違いを必要に応じてマスターする、最新の技術動向についてフォローする、といった努力を続けていただければと思います。

高橋征義（PDF ／ EPUB 版制作担当）

本教科書の PDF・EPUB 作成のお手伝いをいたしました。サーバーエンジニアの方々の技術力向上に貢献できれば幸いです。

著作権

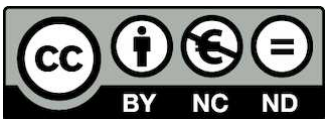


図 1　CC BY-NC-ND

本教科書の目的

本教科書の目的は、実習を通して LPIC 試験に含まれる Linux システムの運用管理に関わる知識を学習することにあります。実際にユーザーやファイル、ディレクトリを作成し、Web サーバーなどを動作させながら、運用管理に関する基本的な技術を修得できます。

想定している実習環境

本教科書での実習環境として、以下の環境を構築しています。

学習者
学習者は 1 名で実習を進めることができます。

マシンの準備

多くの実習は 1 台で実習を進めることができます。ただし、一部の実習は複数のマシンが必要となります。マシンを複数台用意するか、仮想マシンを利用して下さい。講義形式の場合には、受講者同士でネットワーク接続して実習を行うことができます。一部の実習では Windows クライアントから Linux サーバーへのアクセスを行っていますが、Windows クライアントが用意できない場合には実習を行わなくても構いません。

教室とホスト名、IP アドレスなどの割り当て

講義形式で実習を行う場合、ホスト名や IP アドレスが重複しないように環境構築を行ってください。その場合、事前に「Linux サーバー構築標準教科書」の内容を実習していることが望ましいでしょう。

OS

本教科書では、Linux ディストリビューションとして CentOS のバージョン 6.6（64 ビット版）を利用します。ただし、現在は CentOS 7 もリリースされているため、第 7 章で CentOS 7 での変更点について解説しています。

ネットワーク

実習で利用するマシンはインターネットへ接続可能な状態を想定しています。インターネットに接続できない場合には、各種ソフトウェアのインストールを OS のインストール DVD メディアから行う、あるいは必要なファイルを別のインターネット接続可能なマシンでダウンロードして、実習環境にコピーする必要があります。

実習環境の構築

実習環境の構築は「Linux サーバー構築標準教科書」などを参考に、以下の設定で行っていることを想定しています。教室などで各学習者のマシンが 1 つのネットワークに接続している場合には、講師の指示に従ってホスト名、IP アドレスを変更してください。

OS のインストール

CentOS 6.6 64 ビット版を、インストール種別は「Desktop」でインストールします。実習で必要となるパッケージは、随時 yum コマンドを使って追加インストールします。

ネットワークの設定

ネットワークの設定は、以下の通り固定の IP アドレスで設定しています。インターネットへの接続を行う場合には、環境に合わせて変更を行ってください。

設定項目	設定値
ホスト名	server.example.com
IP アドレス	192.168.0.10
ネットマスク	24 ビット（255.255.255.0）
ゲートウェイ	192.168.0.1
DNS サーバー	192.168.0.1

時刻の設定

「システムクロックで UTC を使用」のチェックを外してください。

初期ユーザーの作成

ユーザー名「sato」で初期ユーザーを作成してください。パスワードは任意のパスワードを設定します。

全体の流れ

本教科書では、以下の通りに実習を進めます。

- 第 1 章 ユーザーとグループの管理
- 第 2 章 ネットワークの管理
- 第 3 章 サービスの管理
- 第 4 章 ファイルシステムの管理
- 第 5 章 システムのメンテナンス
- 第 6 章 トラブルシューティング
- 第 7 章 CentOS 7 への移行

後の章では、それよりも前の章での実習を前提に実習が進められている場合があります。実習を省略した場合、実習結果が異なる場合があります。また、ファイアーウォール（iptables）や SELinux などセキュリティに関する設定について理解ができていることを想定しています。正常に実習が行えない場合には、「第 6 章トラブルシューティング」の内容を参考にしてみてください。

実行例の記載

実習はほとんどの作業がコマンドで行われます。読みやすさを高めるために、以下のルールで記述されています。

- Linux マシンが 1 台の場合、コマンドプロンプトが冗長にならないよう、root ユーザーの場合には「#」のみを記しています。設定ファイルのコメントアウトと混同しないように注意が必要です。
- ユーザーを切り替えて実行する場合、Linux マシンが 2 台の場合、どちらのユーザー、マシンで実行するのかを分かるようにするためにコマンドプロンプトを長く記述しています。
- 実行結果の表示が長い場合、改行後の先頭に⇒が入ります。

以下はいくつかの例です。

```
# command ※root ユーザーとして操作
$ command ※一般ユーザーで操作
[root@server ~]# command ※サーバーで root ユーザーとして操作
[sshuser@client ~]$ command ※クライアントでユーザー sshuser として操作
$ id
uid=500(sato) gid=500(sato) 所属グループ=500(sato)
    context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

1 ユーザとグループの管理

1.1 ユーザの管理

Windows や Mac OS X などの「デスクトップ OS」では、基本的には 1 人のユーザが 1 台のマシンを占有して利用します。それに対して、UNIX や Linux などの「サーバ OS」では、複数のユーザが同時に利用できるように最適化された環境になっています。そのため、管理者は利用者一人一人にユーザアカウントを作成し、ユーザは自分専用のユーザアカウントでログインする仕組みとなっています。

1.1.1 ユーザアカウントの種類

ユーザアカウントの種類には、一般アカウントの「一般ユーザ」と、管理者権限である「root ユーザ」(スーパーユーザ)、そしてアプリケーションで利用される「システムユーザ」が存在します。

ユーザの種類	用途
一般ユーザ	ユーザ個人のアカウント
root ユーザ (スーパーユーザ)	管理者権限を持つユーザ
システムユーザ	システムやアプリケーションで使用するユーザ

1.1.2 一般ユーザ

通常のユーザがサーバにログインして利用するためのアカウントです。ユーザ情報の確認は、id コマンドを使います。以下の例では、CentOS インストール時に作成したユーザ sato でログインした後、id コマンドでユーザ情報を確認しています。

```
$ id
uid=500(sato) gid=500(sato) 所属グループ=500(sato)
    context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

uid はユーザを識別するための ID、gid は主グループ ID を示しています。所属グループ（groups）には所属している主グループ ID およびサブグループ ID が表示されます。一般ユーザの uid は、CentOS 6 では 500〜65535 までが使用できます（ディストリビューションによって範囲が異なります）。

1.1.3 root ユーザ

管理者権限を持っている特別なユーザで uid には 0 が付与されています。スーパーユーザとも呼ばれます。root ユーザになるには、主に以下の方法があります。

■Linux のローカルコンソールから root ユーザでログインする

Linux の動作しているマシンを直接操作できる時には、ローカルコンソールから root ユーザでログインできます。

■一般ユーザでログインした後、su コマンドを実行する

一般ユーザから root ユーザになるには、su コマンドを使います。su コマンドを実行するときに-（ハイフンのみ）オプションを付けると、そのユーザでログインしたのと同じことになります。

```
$ su -
Password: ※root ユーザのパスワードを入力
#
```

プロンプトが root ユーザ用の「#」に変わったのが分かります。id コマンドで、ユーザ識別子を表示します。

```
# id
uid=0(root) gid=0(root) 所属グループ=0(root)
    context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

root ユーザの uid は必ず 0 になります。

root ユーザは一般ユーザのように常用してはいけません。何故なら、入力したコマンドの間違い等のオペレーションミスで、システム障害に繋がる危険性があるからです。安全な root 権限の使い方については別章で説明します。

1.1.4 システムユーザ

システムユーザは、バックグラウンドで動くサービスが使用するユーザです。CentOS 6 では uid の 1 から 499 がシステムユーザ用に予約されています。一般ユーザのようにログインをして利用するユーザではありません。

たとえば、SSH を動作させるためには、「sshd」ユーザがシステム内部で sshd デーモンを実行しています。root ユーザは id コマンドを使って、他のユーザの情報を確認できます。sshd サービスのシステムユーザの情報を確認しましょう。

```
# id sshd
uid=74(sshd) gid=74(sshd) 所属グループ=74(sshd)
```

1.1.5 useradd コマンドによる一般ユーザの作成

新規に一般ユーザを作成するには、root ユーザで useradd コマンドを実行します。必要に応じて passwd コマンドでパスワードを設定します。useradd コマンドの引数にユーザ名を指定します。

以下の例では、-c オプションででコメントを入れています。

```
# useradd -c "Ichiro Suzuki" suzuki
# id suzuki
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki)
```

useradd コマンドの主なオプションは以下の通りです。

オプション	説明
-u	ユーザ ID を指定する
-g	主グループ名または主グループ ID を指定する
-G	サブグループを指定する。複数指定するときはカンマ (,) で区切る
-s shell	ユーザのログインシェルを指定する
-c コメント	コメントを入れる（ユーザのフルネームなど）
-d ディレクトリ名	ホームディレクトリを指定する
-e YYYY-MM-DD	ユーザアカウントが無効になる年月日を指定する

1.1.6 パスワードの設定

次に、passwd コマンドを使って、指定ユーザの初期パスワードを設定します。

```
# passwd suzuki
ユーザー suzuki のパスワードを変更。
新しいパスワード: ※ユーザsuzukiの新しいパスワードを入力
新しいパスワードを再入力してください: ※ユーザsuzukiの新しいパスワードを再入力
passwd: 全ての認証トークンが正しく更新できました。
```

管理者である root ユーザが初期パスワードを設定した場合、該当のユーザがログインをしてパスワードを変更します。この時、既存のパスワードと新しく設定するパスワードの両方を聞かれます。新しく設定するパスワードは、誕生日や名前など推測されやすいパスワードにしないように注意しましょう。

以下の例では、ユーザ suzuki でログインした後、パスワードを変更しています。

```
$ passwd
ユーザー suzuki のパスワードを変更。
suzuki 用にパスワードを変更中
現在のUNIXパスワード: ※ユーザsuzukiの現在のパスワードを入力
新しいパスワード: ※ユーザsuzukiの新しいパスワードを入力
新しいパスワードを再入力してください: ※ユーザsuzukiの新しいパスワードを再入力
passwd: 全ての認証トークンが正しく更新できました。
```

1.1.7 ユーザ情報の確認

ユーザ情報は/etc/passwd ファイルに保管されています。ファイルの内容を閲覧する cat コマンドを使って/etc/passwd ファイルを確認します。

```
# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
```

```
（略）
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
tcpdump:x:72:72::/:/sbin/nologin
sato:x:500:500::/home/sato:/bin/bash
suzuki:x:501:501:Ichiro Suzuki:/home/suzuki:/bin/bash
```

/etc/passwd は左から順に以下の情報が入っていて、コロン (:) で区切られています。

項目	意味
ユーザ名	ユーザアカウント名
パスワード	x はシャドウパスワードが設定されている
ユーザ ID	ユーザ ID
グループ ID	グループ ID
コメント	ユーザに関するコメント
ホームディレクトリ	ユーザのホームディレクトリ
シェル	ログインした時に起動するシェル

1.1.8 シャドウパスワードについて

かつての UNIX では、ユーザのパスワードを暗号化して/etc/passwd に記録していました。しかし、/etc/passwd は誰でも内容を読むことができるファイルのため、一般ユーザにパスワードを解析されてしまう危険性がありました。そのため、root ユーザのみ読み取れるシャドウファイル (/etc/shadow) を用意し、暗号化（ハッシュ化）したパスワードを別途格納しています。暗号化は元に戻すこと（復号）ができますが、ハッシュ化は元に戻すことができない（困難）な仕組みです。パスワードがシャドウファイルに格納されていると、/etc/passwd のパスワード部分には x が入るようになっています。

ちなみに、/etc/shadow のパーミッションは 000 に設定されています（ディストリビューションによっては 400)。root ユーザはパーミッションに関係なく読み取ることができますが、その他のユーザは読み書きすることができません。パーミッションの詳細については後述します。

```
# ls -l /etc/shadow
----------. 1 root root 1164  1月  6 06:48 2015 /etc/shadow
```

root ユーザで、ユーザ suzuki のシャドウパスワードを確認してみましょう。

```
# grep suzuki /etc/shadow
suzuki:$6$Tq1q9Ztw$8sh1KFpEGFAmU68P8hYLuGjIml01omSdTELmhGNFLWdielH8CzmLLrIc88G.yGqxty4vuI3xiTKWKJ6H
```

1.2 グループの管理

ユーザを管理するひとつの単位として、「グループ」機能があります。この「グループ」では、所属する部署やユーザの役割などによってユーザにグループを割当て、グループ単位での適切な管理が行えます。

たとえば、特定のディレクトリ配下へのアクセスや、特定のファイルの読み書きを特定のグループのみに制限したいとき、「パーミッション」という機能を使って管理できます。

1.2.1 主グループとサブグループ

ユーザは、1 つ以上のグループに所属している必要があります。ユーザが最初に所属するグループを「主グループ」(またはプライマリーグループ) といいます。所属できる主グループは一つのみで、複数のグループにユーザを所属させたい場合はサブグループを割り当てます。

```
# id suzuki
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki)
```

gid が主グループです。所属グループはユーザが所属しているすべてのグループが表示されます。

1.2.2 /etc/group を確認する

グループの設定は/etc/group ファイルに格納されていますので、確認してみましょう。

```
# cat /etc/group
root:x:0:
bin:x:1:bin,daemon
 （略）
sato:x:500:
suzuki:x:501:
```

useradd コマンドは、主グループの指定が無かった場合には作成するユーザと同じ名前のグループを作成し、そのグループをユーザの主グループに指定します。グループ ID も、作成するユーザの uid と同じになります。

1.2.3 グループの作成

groupadd コマンドでグループを作成します。グループ ID を指定したい時は-g オプションを使います。グループ ID が指定されなかった時には、自動的に割り当てられます。

groupadd コマンドの書式は以下の通りです。

```
groupadd [-g グループID] グループ名
```

グループ ID 5000 を指定して grouptest というグループを作成します。

```
# groupadd -g 5000 grouptest
```

/etc/group ファイルを確認します。

```
# grep grouptest /etc/group
grouptest:x:5000:
```

1.2.4 グループ名の変更

groupmod コマンドでグループ名を変更します。

groupmod コマンドの書式は以下の通りです。

```
groupmod [-n 新しいグループ名] グループ名
```

グループ名を grouptest から eigyou に変更します。

```
# groupmod -n eigyou grouptest
```

/etc/group ファイルを確認します。

```
# grep eigyou /etc/group
eigyou:x:5000:
```

1.2.5 ユーザをサブグループに所属させる

ユーザをサブグループに所属させるときは、usermod コマンドを使います。-G（大文字）オプションで所属させたいサブグループをカンマ区切りですべて指定します。所属していたサブグループの指定を忘れると、そのサブグループの所属から外れてしまうので注意してください。所属しているサブグループが多い場合には、後述する gpasswd コマンドを使用すると、サブグループを一つずつ指定できます。

usermod コマンドの書式は以下の通りです。

```
usermod [-G サブグループ名 [,...]] ユーザ名
```

ユーザ suzuki の所属するサブグループとして eigyou グループを指定します。

```
# usermod -G eigyou suzuki
```

id コマンドでサブグループが追加されているか確認します。

```
# id suzuki
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki),5000(eigyou)
```

所属グループに eigyou サブグループが追加されているのが分かります。

1.2.6 サブグループの所属ユーザを確認する

グループにサブグループとして所属しているユーザは、/etc/group に記述されています。

```
# grep eigyou /etc/group
eigyou:x:5000:suzuki
```

ユーザ suzuki が、eigyou グループにサブグループとして所属しているのが分かります。

1.2.7 gpasswd コマンドを使ってサブグループを管理する

ユーザの所属しているサブグループが多い場合、gpasswd コマンドを使うのが便利です。gpasswd コマンドは、追加や除外したいサブグループを 1 つだけ指定できます。

gpasswd コマンドの書式は以下の通りです。

```
gpasswd -a 追加するユーザ名 グループ名
gpasswd -d 除外するユーザ名 グループ名
```

ユーザ suzuki の所属するサブグループから eigyou グループを除外します。

```
# gpasswd -d suzuki eigyou
Removing user suzuki from group eigyou
# id suzuki
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki)
```

/etc/group を確認して、eigyou グループからユーザ suzuki が削除されていることを確認します。

```
# grep eigyou /etc/group
eigyou:x:5000:
```

eigyou グループからユーザ suzuki が除外されているのが分かります。

再度、ユーザ suzuki の所属するサブグループに eigyou グループを指定します。

```
# gpasswd -a suzuki eigyou
Adding user suzuki to group eigyou
# id suzuki
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki),5000(eigyou)
```

ユーザ suzuki のサブグループに eigyou グループが追加されました。

1.3 パーミッションを使ったファイルシステムのアクセス管理

ファイルシステムのアクセス管理を行うには、「パーミッション」という機能を利用します。パーミッション (permission) とは、日本語で「許可」を意味します。許可されたユーザやグループのみが特定のファイルやディレクトリにアクセスでき、読み取り・書き込み・スクリプト等の実行ができるように設定できます。

1.3.1 パーミッションの確認

ユーザのホームディレクトリに移動します。cd コマンドにオプションや引数を付けなければ、ユーザのホームディレクトリに移動できます。ホームディレクトリとは、ユーザ個人に割り当てられた領域になり、ここにユーザが作ったファイルやプログラムを保存したり、ユーザ独自の設定ファイルを格納しています。

pwd コマンドで現在のディレクトリがホームディレクトリになっていることを確認します。

```
$ cd
$ pwd
/home/suzuki
```

次に touch コマンドで、空のファイルを作ります。

```
$ touch test.txt
```

ls コマンドに、-l オプションを付けて実行し、ファイルの詳細を表示します。

```
$ ls -l
合計 0
-rw-rw-r--. 1 suzuki suzuki 0  1月  6 07:34 2015 test.txt
```

上記の「rw-rw-r–」がパーミッションです。

なお、CentOS 6 の環境では、ll コマンドでも「ls -l」と同様の結果が得られます。「ll」は「ls -l」のエイリアスとなるように設定されています。

```
$ ll
合計 0
-rw-rw-r--. 1 suzuki suzuki 0  1月  6 07:34 2015 test.txt
```

設定されているエイリアスは alias コマンドで確認できます。

```
$ alias
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias vi='vim'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
```

1.3.2 パーミッションの表記方法

パーミッションは rwx の 3 つの文字で表されます。それぞれの意味は以下の通りです。また、各パーミッションは数値で表記することもできます。数値表記は chmod コマンドなどで使用されます。

意味	文字表記	数値表記
読取許可 (Readable)	r	4
書込許可 (Writable)	w	2

意味	文字表記	数値表記
実行許可 (eXecutable)	x	1
何も許可しない	-	0

上記 test.txt のアクセス権限は以下のようになっています。

	所有ユーザ（user）	所有グループ (group)	その他 (other)
文字表記	rw-	rw-	r–
数値表記	4+2+0=6	4+2+0=6	4+0+0=4

先頭から、所有ユーザ、グループ、どちらにも該当しないその他のユーザのアクセス権限を示しています。

- 先頭の「rw-」が所有ユーザ、すなわちユーザ suzuki の権限で、読取・書込ができます。
- 次の「rw-」が所有グループ、すなわち suzuki グループの権限で、読取・書込ができます。
- 最後の「r–」がユーザにもグループにも該当しないその他のユーザの権限で、読み取りのみ可能です。

1.3.3 パーミッションの数値表記

パーミッションの数値表記は、設定したい権限に対応する数値の合計値を、所有ユーザ、所有グループ、その他の順に並べた 3 桁の数値で表されます。

たとえば、上記の例の「rw-rw-r–」というパーミッションを数字で表記すると「664」になります。

1.3.4 ディレクトリのパーミッション

ディレクトリのパーミッションも、基本的な考え方はファイルのパーミッションと同じです。異なる点として、実行権限が無いとそのディレクトリに移動してカレントディレクトリにすることができません。

mkdir コマンドで testdir という新規ディレクトリを作成し、アクセス権限を変更してみます。

```
$ mkdir testdir
$ ls -l
合計 4
-rw-rw-r--. 1 suzuki suzuki    0  1月  6 07:34 2015 test.txt
drwxrwxr-x. 2 suzuki suzuki 4096  1月  6 07:42 2015 testdir
```

testdir ディレクトリのパーミッション表記の先頭にディレクトリを識別する d が付いていることが確認できます。このディレクトリのパーミッションは、rwx(4+2+1)、rwx(4+2+1)、r-x(4+1) で 775 になっています。その他のユーザはこのディレクトリにアクセスすることはできますが、書き込みは行えません。

パーミッションの変更は chmod コマンドを使用します。chmod コマンドの書式は以下の通りです。

```
chmod モード ファイル
```

モードの指定は文字表記、数値表記の両方が行えます。数値表記は指定された値に設定しますが、文字表記は + と-でパーミッションの付与、または解除を指定します。

以下、文字表記でのモード指定の例です。

モード指定	意味
ug+x	ユーザとグループに実行権限を付与
a+x	すべてのユーザに実行権限を付与
g-w	グループの書き込み権限を解除

以下の例では、chmod コマンドでディレクトリのパーミッションからユーザ自身の実行権限を解除することで、カレントディレクトリにできなくなることを確認しています。

```
$ chmod u-x testdir
$ ls -l
合計 4
-rw-rw-r--. 1 suzuki suzuki    0  1月  6 07:34 2015 test.txt
drw-rwxr-x. 2 suzuki suzuki 4096  1月  6 07:42 2015 testdir
$ cd testdir
-bash: cd: testdir: 許可がありません
$ chmod u+x testdir
$ cd testdir
$ pwd
/home/suzuki/testdir
```

1.3.5 ユーザアカウントの有効期限を設定する

ユーザアカウントが使用できる有効期限を設定できます。たとえば、期限が決まっているプロジェクトなど、ユーザアカウントの使用が期間限定の場合に有効期限を設定します。新規アカウント追加時には useradd コマンド、すでに存在するアカウントの場合には usermod コマンドに-e オプションを付与して有効期限を指定できます。

ユーザアカウントの有効期限設定の書式は以下の通りです。

```
useradd -e YYYY-MM-DD ユーザ名
usermod -e YYYY-MM-DD ユーザ名
```

usermod コマンドで既存ユーザアカウントの有効期限を設定します。ユーザアカウントに有効期限を設定すると、設定日にアカウントがロックされて使用できなくなります。

以下の例では、動作確認のために有効期限を本日の日付で設定しています。

```
# usermod -e 2015-1-6 suzuki
```

アカウント有効期限を確認するために、chage コマンドを使います。

```
# chage -l suzuki
Last password change                    : Jan 05, 2015
Password expires                        : never
Password inactive                       : never
Account expires                         : ※Jan 06, 2015
Minimum number of days between password change      : 0
Maximum number of days between password change      : 99999
Number of days of warning before password expires   : 7
```

確認のため、アカウント有効期限を設定したユーザアカウントでマシンにログインします。「Your account has expired」と表示され、アカウントがロックされている状態になっています。

```
login: suzuki
Password: ※ユーザsuzukiのパスワードを入力
Your account has expired; please contact your system administrator
```

有効期限をリセットして無期限有効にするには、以下のように「''」（シングルクォート 2 つ）で空の有効期限を指定します。アカウントの有効期限（Account expires）が never に設定されます。

```
# usermod -e '' suzuki
# chage -l suzuki
```

```
Last password change                : Jan 05, 2015
Password expires                    : never
Password inactive                   : never
Account expires                     : ※never
Minimum number of days between password change      : 0
Maximum number of days between password change      : 99999
Number of days of warning before password expires   : 7
```

1.3.6 パスワードの有効期限を設定する

ユーザのパスワードの有効期限を設定したいときは、chage コマンドを使います。-M オプションでパスワードの有効な日数を指定します。

以下の例ではパスワードの有効日数を 30 に設定しているので、30 日毎にパスワードを再設定する必要があります。

```
# chage -M 30 suzuki
```

パスワードの有効期限を確認します。Password expires で表示された日付の翌日以降になると、ユーザログイン時、強制的にパスワードの変更要求を行います。

```
# chage -l suzuki
Last password change                : Jan 05, 2015
Password expires                    : ※Feb 04, 2015
Password inactive                   : never
Account expires                     : never
Minimum number of days between password change      : 0
Maximum number of days between password change      : 30
Number of days of warning before password expires   : 7
```

パスワードの有効期限を即座に失効させるには、-d オプションで 0 を指定します。このオプションは、パスワードが最後に変更された日付の値を 1970 年 1 月 1 日に設定し、即座にパスワードを失効させ、ユーザログイン時に強制的にパスワード変更を要求できます。

```
# chage -d 0 suzuki
```

chage コマンドでアカウントの情報を確認してみると、Last password change、Password expires、Password inactive の値が「password must be changed」になっていることが分かります。

```
# chage -l suzuki
Last password change                : ※password must be changed
Password expires                    : ※password must be changed
Password inactive                   : ※password must be changed
Account expires                     : never
Minimum number of days between password change      : 0
Maximum number of days between password change      : 30
Number of days of warning before password expires   : 7
```

確認のため、設定したユーザアカウントでログインします。即座にパスワード再設定が要求されます。

```
login: suzuki
Password: ※ユーザsuzukiの現在のパスワードを入力
You are required to change your password immediately (root enforced)
Changing password for suzuki.
(current) UNIX password: ※ユーザsuzukiの現在のパスワードを入力
New password: ※ユーザsuzukiの新しいパスワードを入力
```

```
Retype new password: ※ユーザsuzukiの新しいパスワードを再入力
```

1.3.7 ユーザの削除

ユーザを削除します。ユーザの削除後はログインできなくなります。

ユーザに登録してある cron も実行できなくなりますので、念のため確認をしてから削除しましょう。一般ユーザにシステムを動かすために必要な cron を登録しないようにしましょう。そのユーザが削除されてしまった場合、cron が実行されなくなりシステムに重大な障害を与えてしまう場合があります。cron については第 3 章で解説します。

以下の例では、ユーザ testuser を追加し、削除しています。

```
# useradd testuser
# id testuser
uid=502(testuser) gid=502(testuser) 所属グループ=502(testuser)
# userdel testuser
# id testuser
id: testuser: そのようなユーザは存在しません
```

userdel コマンドをオプション無しで実行すると、ユーザのホームディレクトリや受信したメールを格納するメールスプールは削除されません。ユーザ削除と同時にホームディレクトリなども削除したい場合には、userdel コマンドに-r オプションをつけて実行する必要があります。

```
# ls -l /home
合計 28
drwx------.  2 root   root   16384  1月  6 06:07 2015 lost+found
drwx------. 26 sato   sato    4096  1月  6 06:49 2015 sato
drwx------.  5 suzuki suzuki  4096  1月  6 09:00 2015 suzuki
drwx------.  4   ※502    502※  4096  1月  6 09:56 2015 testuser
# ls -l /var/spool/mail
合計 0
合計 0
-rw-rw----. 1 rpc    mail 0  1月  6 06:11 2015 rpc
-rw-rw----. 1 sato   mail 0  1月  6 06:23 2015 sato
-rw-rw----. 1 suzuki mail 0  1月  6 06:48 2015 suzuki
-rw-rw----. 1   ※502※ mail 0  1月  6 09:56 2015 testuser
```

このように、所有ユーザが削除されたディレクトリやファイルは、パーミッションを確認すると所有ユーザが元のユーザ ID で表示されるようになります。

再度ユーザ testuser を作成します。

```
# useradd testuser
useradd: 警告: ホームディレクトリが既に存在します。
skel ディレクトリからのコピーは行いません。
メールボックスファイルを作成します: ファイルが存在します
# ls -l /home
合計 28
drwx------.  2 root     root     16384  1月  6 06:07 2015 lost+found
drwx------. 26 sato     sato      4096  1月  6 06:49 2015 sato
drwx------.  5 suzuki   suzuki    4096  1月  6 09:00 2015 suzuki
drwx------.  4 ※testuser testuser※  4096  1月  6 09:56 2015 testuser
# ls -l /var/spool/mail
合計 0
-rw-rw----. 1 rpc      mail 0  1月  6 06:11 2015 rpc
```

```
-rw-rw----. 1 sato     mail 0  1月  6 06:23 2015 sato
-rw-rw----. 1 suzuki   mail 0  1月  6 06:48 2015 suzuki
-rw-rw----. 1 ※testuser※ mail 0  1月  6 09:56 2015 testuser
```

同じユーザ ID（上記の例では 502）でユーザが追加されたので、ホームディレクトリとメールスプールはユーザ testuser が再度所有ユーザになっています。もし、削除後に追加された別のユーザに同じユーザ ID（502）が割り当てられると、ファイルやディレクトリの所有権が別のユーザに移ってしまうので注意が必要です。

userdel -r コマンドで削除します。ユーザ testuser のホームディレクトリとメールスプールが同時に削除されます。

```
# userdel -r testuser
# ls -l /home
合計 24
drwx------.  2 root   root   16384  1月  6 06:07 2015 lost+found
drwx------. 26 sato   sato    4096  1月  6 06:49 2015 sato
drwx------.  5 suzuki suzuki  4096  1月  6 09:00 2015 suzuki
# ls -l /var/spool/mail
合計 0
-rw-rw----. 1 rpc    mail 0  1月  6 06:11 2015 rpc
-rw-rw----. 1 sato   mail 0  1月  6 06:23 2015 sato
-rw-rw----. 1 suzuki mail 0  1月  6 06:48 2015 suzuki
```

1.3.8 グループの削除

グループを削除するには、groupdel コマンドを使用します。ユーザが所属している主グループは削除できませんが、サブグループは警告無しに削除されます。グループを削除する前に/etc/group を参照して、そのグループに所属しているユーザがいないか確認しておきます。

以下の例ではユーザ testuser（主グループ testuser）を作成し、グループ testgroup にサブグループとして所属させています。主グループは削除できませんが、サブグループは削除できます。

```
# useradd testuser
# groupadd testgroup
# gpasswd -a testuser testgroup
Adding user testuser to group testgroup
# id testuser
uid=502(testuser) gid=502(testuser) 所属グループ=502(testuser),5001(testgroup)
# groupdel testuser
groupdel: ユーザ 'testuser' のプライマリグループは削除できません。
# groupdel testgroup
# id testuser
uid=502(testuser) gid=502(testuser) 所属グループ=502(testuser)
```

1.4 SSH によるリモートログイン

SSH (Secure Shell) とは、リモート (遠隔) のサーバにログインしてサーバを操作するためのプロトコルです。SSH は、外部へ通信の内容が漏れないように通信が暗号化されています。

Linux では、OpenSSH のサーバおよびクライアントが利用できます。また、Linux サーバに対して Windows から SSH でリモートログインすることもできます。

1.4.1 環境の準備

2 台の Linux マシンを使って SSH によるリモートログインを試してみましょう。SSH クライアントから、SSH サーバに対して SSH を使って接続をします。

以下の例では、2 台の Linux マシンを使って説明します。

役割	ホスト名	IP アドレス
サーバ	server.example.com	192.168.0.10
クライアント	client.example.com	192.168.0.101

また、それぞれの Linux マシンを名前解決できるように、/etc/hosts に以下の記述を追加しておきます。

```
192.168.0.10    server.example.com server
192.168.0.101   client.example.com client
```

1.4.2　SSH サービスの状態確認と開始

CentOS では OpenSSH サーバはデフォルトでインストールされて自動的に起動しています。sshd デーモンが起動していることをを確認しておきます。SSH プロトコルはポート番号 22 番を使用しています。

```
[root@server ~]# lsof -i:22
COMMAND   PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd     1718   root    3u  IPv4  13399      0t0  TCP *:ssh (LISTEN)
sshd     1718   root    4u  IPv6  13401      0t0  TCP *:ssh (LISTEN)
```

1.4.3　SSH のログイン認証方法

SSH のログイン認証方法には、以下の方法があります。

■パスワード認証による接続

サーバに登録済みのユーザ名と、ユーザのログインパスワードを使ってログイン認証を行います。簡単で分かりやすい認証方式ですが、ユーザ名とパスワードが分かれば誰でもログインできてしまうので、インターネットに接続するサーバなどでは使用しません。デフォルトで有効になっているので、SSH サーバの設定を変更して無効にしておきます。

■公開鍵認証による接続

事前に作成した公開鍵をログインしたいサーバに登録しておきます。公開鍵に対応した秘密鍵を持っているユーザだけがログインできます。パスワード認証に比べて事前の設定が必要になりますが、ログインするためには秘密鍵が必要になるので、パスワード認証より安全な認証の仕組みです。

1.4.4　パスワード認証による接続

パスワード認証を使って、SSH サーバに接続してログイン認証を行います。

サーバに事前にログイン用のユーザ sshuser を作成します。

```
[root@server ~]# useradd sshuser
[root@server ~]# passwd sshuser
ユーザ sshuser のパスワードを変更。
新しいパスワード: ※ユーザsshuserの新しいパスワードを入力
新しいパスワードを再入力してください: ※ユーザsshuserの新しいパスワードを再入力
passwd: 全ての認証トークンが正しく更新できました。
```

クライアントにも同様にユーザ sshuser を作成しておきます。SSH クライアントは接続時にログイン認証で使用するユーザ名を指定できるので、クライアントでのユーザ作成は省略しても構いません。

クライアントにユーザ sshuser でログインし、サーバに SSH で接続します。接続するには ssh コマンドを使用します。ユーザ名を省略すると、ssh コマンドを実行したユーザのユーザ名が指定されたことになります。

ssh コマンドの書式は以下の通りです。

```
$ ssh [ユーザ名@]接続先
```

接続先には IP アドレス、又は名前解決できるホスト名を指定します。

```
[sshuser@client ~]$ ssh sshuser@server
```

SSH サーバに接続すると、サーバから「SSH サーバ証明書」が送られてきます。初回の接続時には以下のように尋ねられるので、yes と入力し、サーバで作成したユーザ sshuser のパスワードを入力します。サーバにログインすると、コマンドプロンプトの表示がサーバ側のものに変わったことが分かります。

```
[sshuser@client ~]$ ssh sshuser@server
The authenticity of host 'server (192.168.0.10)' can't be established.
RSA key fingerprint is b6:95:54:92:62:cb:c8:f7:17:97:88:8e:69:f9:2a:dd.
Are you sure you want to continue connecting (yes/no)? ※yes ←yesと入力
Warning: Permanently added 'server,192.168.0.10' (RSA) to the list of known hosts.
sshuser@server's password: ※サーバに作成したユーザsshuserのパスワードを入力
[sshuser@server ~]$
```

サーバにログインできたら、ifconfig コマンドで IP アドレスを確認しましょう。IP アドレスがサーバ側のもの（192.168.0.10）であることが確認できます。

```
[sshuser@server ~]$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:1C:42:65:AF:C4
          inet addr:192.168.0.10  Bcast:10.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::21c:42ff:fe65:afc4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:19972 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11094 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:15984761 (15.2 MiB)  TX bytes:992110 (968.8 KiB)
```

サーバからログアウトするには、exit コマンドを使用します。

```
[sshuser@server ~]$ exit
logout
Connection to server closed.
[sshuser@client ~]$
```

1.4.5 ssh コマンドの冗長モードによるトラブルシューティング

もし、ログインがうまくいかない場合は ssh コマンドに-v オプション (冗長モード) を付けてデバッグ用のメッセージを表示させ、詳細を確認します。

```
[sshuser@client ~]$ ssh -v sshuser@server
OpenSSH_5.3p1, OpenSSL 1.0.1e-fips 11 Feb 2013
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: Applying options for *
debug1: Connecting to server [192.168.0.10] port 22.
debug1: Connection established.
（以下略）
```

1.4.6 SSH サーバ証明書による「なりすまし」の防止

一度接続したことのあるサーバの SSH サーバ証明書は、クライアントのホームディレクトリに作られた.ssh ディレクトリの中に作成された known_hosts ファイルに保存されます。2 回目以降の接続時には、初回に尋ねられた表示は出ず、すぐに認証のためのパスワード入力が要求されます。

```
[sshuser@client ~]$ ssh sshuser@server
sshuser@server's password:
```

cat コマンドでクライアントにある~/.ssh/known_hosts ファイルの中身を確認してみましょう。

```
[sshuser@client ~]$ cat ~/.ssh/known_hosts
server,192.168.0.10 ssh-rsa
    AAAAB3NzaC1yc2EAAAABIwAAAQEA0xULiTzWSingpALtma51pnsMrOwW8drd+9S2ocC9/LF0ThhnQCZ49xAYx2DRNqTNNS
```

SSH サーバ証明書は、SSH サーバに接続した際にサーバからクライアントに対して送られてきます。初回接続時は /.ssh/known_hosts に保存されている SSH サーバ証明書が無いので、接続してもよいか確認されます。yes と答えるとサーバ証明書は/.ssh/known_hosts に保存されます。

2 回目以降の接続では、送られてきた SSH サーバ証明書と known_hosts に保存してある SSH サーバ証明書を比較して、同一であれば同じサーバであることが分かります。もし異なる SSH サーバ証明書が送られてきた場合には、別のサーバが「なりすまし」をしている可能性があるので、ssh コマンドは警告を表示して接続を中断します。

また、仮にコピーした SSH サーバ証明書を送ってきてサーバなりすましをしようとしても、その後の接続手順の中で確認作業を行っているので、やはり接続は中断され、サーバなりすましは失敗します。SSH サーバには、SSH サーバ証明書（公開鍵）とサーバ秘密鍵の、ペアになった 2 つの鍵が必要だからです。公開鍵と秘密鍵については後述します。

サーバの再インストールなどを行うと、サーバの SSH サーバ証明書は再作成され、変更されてしまいます。その場合には、クライアントの /.ssh/known_hosts ファイルに登録されている SSH サーバ証明書を削除して下さい。/.ssh/known_hosts ファイルは単なるテキストファイルなので、vi エディタなどで該当する SSH サーバ証明書を 1 行削除します。

1.4.7 公開鍵認証による接続

パスワード認証では「ユーザ名」と「パスワード」で認証しますが、もしこのパスワードが漏れてしまうと非常に危険です。また、セキュリティ攻撃用のプログラムを使って手当たり次第にパスワードを試す「総当たり攻撃」の可能性もあります。そこで、より安全な認証方法として公開鍵認証による接続が利用できます。インターネット上に公開するサーバの場合には、パスワード認証を禁止し、この公開鍵認証で接続します。

公開鍵認証は、SSH 接続用の公開鍵と秘密鍵のキーペアを生成し、接続先のサーバに公開鍵を登録して認証します。これを「公開鍵認証」といいます。

公開鍵認証の大まかな手順としては、以下のようになります。

1. 公開鍵と秘密鍵のキーペアを生成する
2. クライアントに公開鍵と秘密鍵を設置する
3. サーバに公開鍵を設置する

1.4.8 SSH 公開鍵・秘密鍵の作成

SSH 公開鍵認証に使用する公開鍵と秘密鍵を作成します。Linux では ssh-keygen コマンドを使用します。クライアントで作成すれば、公開鍵・秘密鍵の生成と、クライアントへの鍵の設置が同時に行えるので、以下の作業はクライアント上で行います。鍵の作成後、公開鍵をサーバに設置します。

ssh-keygen コマンドを実行すると、鍵の設置場所とパスフレーズの入力が求められます。公開鍵と秘密鍵の設置場所は、デフォルトでは ssh-keygen コマンドを実行したユーザのホームディレクトリにある.ssh ディレクトリに作成されます。パスフレーズは、公開鍵認証を行う際に秘密鍵を有効にするためのパスワードのようなものです。万一秘密鍵を盗まれたとしても、パスフレーズが分からなければ鍵の所有ユーザになりすまして SSH サーバに接続することはできません。

```
[sshuser@client ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/sshuser/.ssh/id_rsa): ※Enterキーを押す
Enter passphrase (empty for no passphrase): ※秘密鍵のパスフレーズを入力
Enter same passphrase again: ※秘密鍵のパスフレーズを再入力
Your identification has been saved in /home/sshuser/.ssh/id_rsa.
Your public key has been saved in /home/sshuser/.ssh/id_rsa.pub.
The key fingerprint is:
91:47:d4:85:39:58:59:7e:d4:0b:50:7c:56:f7:28:45 sshuser@client
The key's randomart image is:
+--[ RSA 2048]----+
|          .o==OE *|
|          o. *= =+|
|         o . ..* +|
|          o   . o |
|         S        |
|                  |
|                  |
|                  |
|                  |
+------------------+
```

~/.ssh ディレクトリに作成された公開鍵（id_rsa.pub）と秘密鍵 (id_rsa) を確認します。.ssh ディレクトリは ssh コマンド実行時、または ssh-keygen コマンド実行時に自動的に作成されます。

```
[sshuser@client ~]$ ls -ld .ssh
drwx------. 2 sshuser sshuser 4096  1月  7 14:17 2015 .ssh
[sshuser@client ~]$ ls -l .ssh
合計 8
-rw-------. 1 sshuser sshuser 1743  1月  7 14:17 2015 id_rsa
-rw-r--r--. 1 sshuser sshuser  396  1月  7 14:17 2015 id_rsa.pub
```

鍵の中身はテキストファイルになっているので、cat コマンドで確認できます。

```
[sshuser@client ~]$ cat .ssh/id_rsa.pub
ssh-rsa
     AAAAB3NzaC1yc2EAAAABIwAAAQEAxaKrCiK5rrJBqtjG3NbWoRlGJMGEqkND6WYTfLhBby55+1C4kLL6GGXkGPWqIIqFk6
     sshuser@client
```

公開鍵は 1 行のテキストですが、秘密鍵は何行かに分かれたフォーマットになっています。

```
[sshuser@client .ssh]$ cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,9A38288879701873A

kSkjcd/9+VWwk2NR8CuET4CXKu7ZIAOkNmvHwUZVMpUlnDwqxeznXP4NVGEq5uFD
（略）
Jw6FruKNyjl8mqLtrj+eltCUh6N4Z+NPVzlAHMQ9IQmBjdpArj0SLQ==
-----END RSA PRIVATE KEY-----
```

1.4.9 クライアントの.ssh ディレクトリおよび公開鍵・秘密鍵のパーミッション

公開鍵認証に使用する公開鍵・秘密鍵、およびそれらを格納する.ssh ディレクトリはセキュリティを守るためパーミッションが厳密に決められています。ssh-keygen コマンドを使用して鍵を作成した際にはパーミッションは適切に設定されていますが、別のマシンで作成した鍵をコピーしてくる場合には、パーミッションを自分で設定する必要があります。また、所有ユーザは ssh コマンドを実行するユーザである必要があります。初期設定時などにユーザの作成から公開鍵・秘密鍵の設置までを root ユーザで行っていると、所有ユーザが root になってしまうので注意が必要です。

設定するパーミッションは以下の通りです。

ディレクトリおよびファイル	パーミッション
~/.ssh ディレクトリ	rwx——(700)
id_rsa.pub（公開鍵）	rw-r–r–(644)
id_rsa（秘密鍵）	rw———(600)

1.4.10 サーバへの公開鍵の設置

次に、クライアントの SSH 公開鍵 (id_rsa.pub) をサーバに設置します。以下の手順で設置を行います。

1. クライアントからサーバに公開鍵をコピー
2. ~/.ssh ディレクトリを作成
3. ~/.ssh/authorized_keys ファイルを作成
4. 公開鍵の内容を~/.ssh/authorized_keys に追加
5. 公開鍵認証でログインできることを確認

＃ 1

1. クライアントからサーバに公開鍵をコピー

クライアントからサーバに公開鍵（id_rsa.pub）をコピーします。ここでは SSH プロトコルを使ったファイルコピーを行う scp コマンドを使います。

scp コマンドの書式は以下の通りです。

```
scp コピー元ファイル ユーザ名@接続先:コピー先ファイル
```

以下のように scp コマンドを実行して、~/.ssh/id_rsa.pub を、サーバのユーザ sshuser のホームディレクトリにリモートコピーします。

```
[sshuser@client ~]$ scp ~/.ssh/id_rsa.pub sshuser@server:~
sshuser@server's password: ※サーバに作成したユーザsshuserのパスワードを入力
id_rsa.pub                                    100%  396     0.4KB/s   00:00
```

＃ 2

1. ~/.ssh ディレクトリを作成

ssh コマンドでサーバにログインし、公開鍵の設置を行います。

```
[sshuser@client ~]$ ssh sshuser@server
sshuser@server's password: ※サーバに作成したユーザsshuserのパスワードを入力
Last login: Tue Jan  6 10:58:42 2015 from client
[sshuser@server ~]$
```

公開鍵（id_rsa.pub）がコピーされていることを確認します。

```
[sshuser@server ~]$ ls -l
合計 4
-rw-r--r--. 1 sshuser sshuser 396  1月  6 10:56 2015 id_rsa.pub
[sshuser@server ~]$ cat id_rsa.pub
ssh-rsa
     AAAAB3NzaC1yc2EAAAABIwAAAQEAxaKrCiK5rrJBqtjG3NbWoRlGJMGEqkND6WYTfLhBby55+1C4kLL6GGXkGPWqIIqFk6
     sshuser@client
```

ホームディレクトリに.ssh ディレクトリを作成し、chmod コマンドでパーミッションを変更します。

```
[sshuser@server ~]$ mkdir .ssh
[sshuser@server ~]$ chmod 700 .ssh
[sshuser@server ~]$ ls -ld .ssh
drwx------. 2 sshuser sshuser 4096  1月  6 10:59 2015 .ssh
```

3

1. ~/.ssh/authorized_keys ファイルを作成

.ssh ディレクトリの中に authorized_keys ファイルを作成し、パーミッションを変更します。公開鍵はこのファイルの中に追加していきます。

```
[sshuser@server ~]$ touch .ssh/authorized_keys
[sshuser@server ~]$ chmod 600 .ssh/authorized_keys
[sshuser@server ~]$ ls -l .ssh
合計 0
-rw-------. 1 sshuser sshuser 0  1月  6 10:59 2015 authorized_keys
```

4

1. 公開鍵の内容を~/.ssh/authorized_keys に追加

公開鍵を authorized_keys に追加します。cat コマンドで出力をリダイレクトします。出力で “»” を使うと既存ファイルの authorized_keys ファイルを上書きせずに追記する事ができます。authorized_keys ファイルを作成する作業では、cp コマンドや mv コマンドは使用しないでください。authorized_keys ファイルを上書きする危険性がある他、SELinux が有効になっている場合、正常に動作しないことがあります。

```
[sshuser@server ~]$ cat id_rsa.pub >> .ssh/authorized_keys
[sshuser@server ~]$ cat .ssh/authorized_keys
ssh-rsa
     AAAAB3NzaC1yc2EAAAABIwAAAQEAxaKrCiK5rrJBqtjG3NbWoRlGJMGEqkND6WYTfLhBby55+1C4kLL6GGXkGPWqIIqFk6
     sshuser@client
```

5

1. 公開鍵認証でログインできることを確認

一旦サーバからログアウトし、再度接続します。公開鍵が正しく設置されていれば、秘密鍵のパスフレーズの入力が求められます。もしパスワード認証を求められるような場合には、ファイル名やパーミッションなど、設置の手順を再確認してみてください。

```
[sshuser@server ~]$ exit
logout
Connection to server closed.
[sshuser@client ~]$ ssh sshuser@server
Enter passphrase for key '/home/sshuser/.ssh/id_rsa': ※秘密鍵のパスフレーズを入力
```

```
Last login: Tue Jan  6 10:59:03 2015 from client
[sshuser@server ~]$
```

1.4.11 ssh-copy-id コマンドを使った公開鍵の設置

SSH 公開鍵を手動で登録する方法の他に、ssh-copy-id コマンドを使った公開鍵の登録方法があります。ssh-copy-id コマンド一つで、ホストの authorized_keys に公開鍵を自動的に登録できます。

ssh-copy-id コマンドの書式は以下の通りです。

```
$ ssh-copy-id ユーザ名@接続先
```

公開鍵と秘密鍵を作成した後、ssh-copy-id コマンドを実行してサーバに公開鍵を登録します。

```
[sshuser@client ~]$ ssh-copy-id sshuser@server
sshuser@server's password:
Now try logging into the machine, with "ssh 'sshuser@server'", and check in:

  .ssh/authorized_keys

to make sure we haven't added extra keys that you weren't expecting.
```

公開鍵認証で SSH 接続できるか確認します。

```
[sshuser@client ~]$ ssh sshuser@server
Enter passphrase for key '/home/sshuser/.ssh/id_rsa': ※秘密鍵のパスフレーズを入力
Last login: Tue Jan  6 11:01:52 2015 from client
[sshuser@server ~]$
```

公開鍵認証で接続できるようになったら、OpenSSH サーバの設定を変更してパスワード認証による接続を禁止します。設定方法は後述します。

1.4.12 scp コマンドを使ったファイル転送

scp コマンドを使うと、SSH プロトコルを使用してファイル転送ができます。クライアントで作成した公開鍵をサーバにコピーするために既に使用しましたが、ディレクトリ内の複数のファイルを再帰的に転送することもできます。

クライアントのホームディレクトリに testdir ディレクトリを作成し、その中に複数のファイルを作ります。次に、scp コマンドに-r オプションを付与して実行し、ディレクトリの中身をすべて再帰的に転送します。

```
[sshuser@client ~]$ mkdir testdir
[sshuser@client ~]$ cd testdir
[sshuser@client testdir]$ touch testfile1 testfile2
[sshuser@client testdir]$ ls
testfile1  testfile2
[sshuser@client testdir]$ cd
[sshuser@client ~]$ scp -r testdir sshuser@server:~
Enter passphrase for key '/home/sshuser/.ssh/id_rsa': ※秘密鍵のパスフレーズを入力
testfile1                                     100%    0     0.0KB/s   00:00
testfile2                                     100%    0     0.0KB/s   00:00
```

転送先のサーバでファイルが転送されたか確認します。

```
[sshuser@client ~]$ ssh sshuser@server
Enter passphrase for key '/home/sshuser/.ssh/id_rsa': ※秘密鍵のパスフレーズを入力
Last login: Tue Jan  6 11:02:46 2015 from client
```

```
[sshuser@server ~]$ ls
id_rsa.pub  testdir
[sshuser@server ~]$ ls -l testdir
合計 0
-rw-rw-r--. 1 sshuser sshuser 0  1月  6 11:04 2015 testfile1
-rw-rw-r--. 1 sshuser sshuser 0  1月  6 11:04 2015 testfile2
```

1.4.13 sftp コマンドを使ったファイル転送

SFTP (SSH File Transfer Protocol) とは、SSH でファイルを送受信できるプロトコルです。動作は FTP に似ています。

あらかじめ転送用のファイルを作成した後、sftp コマンドでクライアントからサーバにログインします。ログインできると、”sftp>” というプロンプトが表示されます。

```
[sshuser@client ~]$ touch sftptestfile
[sshuser@client ~]$ ls
sftptestfile  testdir
[sshuser@client ~]$ sftp sshuser@server
Connecting to server...
Enter passphrase for key '/home/sshuser/.ssh/id_rsa': ※秘密鍵のパスフレーズを入力
sftp>
```

“put ファイル名” でサーバにファイルを転送できます。

```
sftp> put sftptestfile
Uploading sftptestfile to /home/sshuser/sftptestfile
sftptestfile                                  100%    0     0.0KB/s   00:00
```

ls コマンドでファイルの確認ができます。

```
sftp> ls
id_rsa.pub      sftptestfile    testdir
sftp> ls -l
-rw-r--r--    1 sshuser  sshuser       396 Jan  6 10:56 id_rsa.pub
-rw-rw-r--    1 sshuser  sshuser         0 Jan  6 11:20 sftptestfile
drwxrwxr-x    2 sshuser  sshuser      4096 Jan  6 11:04 testdir
sftp> exit
[sshuser@client ~]$
```

SFTP の主なコマンドは以下の通りです。

コマンド	動作
pwd	カレントディレクトリの表示
ls	ファイルの表示
cd [パス]	カレントディレクトリの移動
put [-P] ローカルパス [リモートパス]	ファイルをリモートに転送。-P オプションは所有権やパーミッションを維持
get [-P] リモートパス [ローカルパス]	ファイルをローカルに転送。-P オプションは所有権やパーミッションを維持
rm パス	ファイルを削除
mkdir パス	ディレクトリを作成
rmdir パス	ディレクトリを削除
lpwd	ローカルのカレントディレクトリの表示
lls [ls コマンドのオプション] [パス]	ローカルのファイルの表示
lcd パス	ローカルのカレントディレクトリの移動
lmkdir パス	ローカルのディレクトリを作成

1.4.14 Tera Term を使った Windows クライアントからのパスワード認証による接続

Windows クライアントから Tera Term を使って SSH で OpenSSH サーバにログインできます。また、Tera Term の機能の一つである「SSH SCP」でファイルを送受信できます。

Tera Term はインターネット上にある Tera Term の Web サイトからダウンロードして、インストールできます。インストーラーでインストールしたい場合には.EXE 形式のファイルをダウンロードして実行します。

```
http://sourceforge.jp/projects/ttssh2/
```

クライアントにインストールした Tera Term を起動して、サーバに接続します。

1. Tera Term を起動します。「新しい接続」ダイアログが表示されます。

図 2 接続先の IP アドレス、あるいはホスト名を指定します

「ホスト」に接続先として IP アドレスか名前解決可能なホスト名を入力します。サービスは「SSH」を選択します。「OK」ボタンをクリックして接続します。

＃ 2

2. 初回のみ、「セキュリティ警告」が表示されます。

初めての接続先の場合、「セキュリティ警告」ダイアログが表示されます。接続先から送られてきたサーバ証明書が knows hosts リストに登録されていないためです。確認の上、「続行」ボタンをクリックします。

＃ 3

3. パスワード認証を行います。

「SSH 認証」ダイアログが表示されるので、ユーザ名、パスフレーズ（パスワード）を入力して、パスワード認証でサーバにログインします。

＃ 4

4. ターミナル画面が表示されます。

パスワード認証が成功すると、ターミナル画面が表示されてログインシェルが起動します。

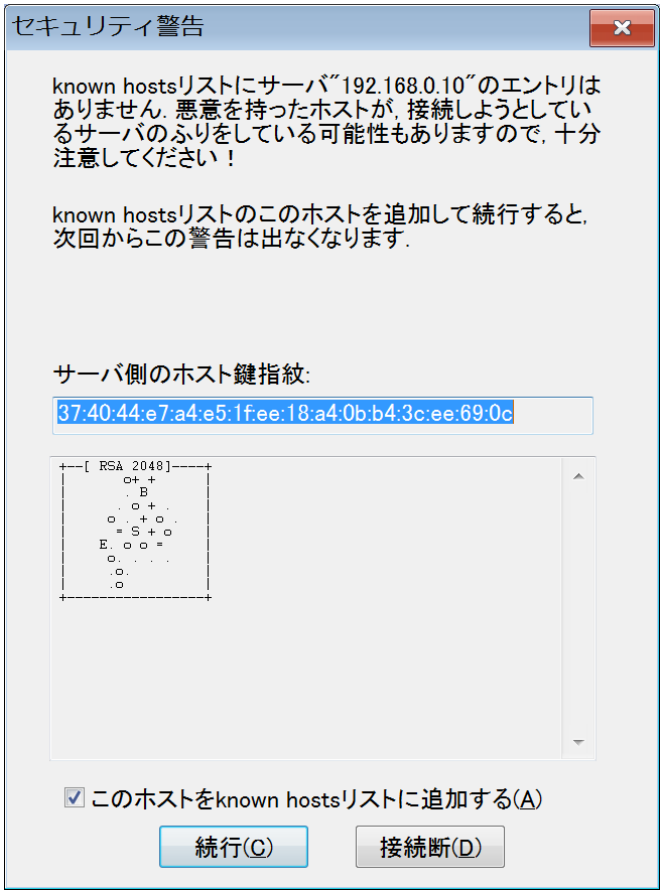


図 3　セキュリティ警告は初回のみ表示されます

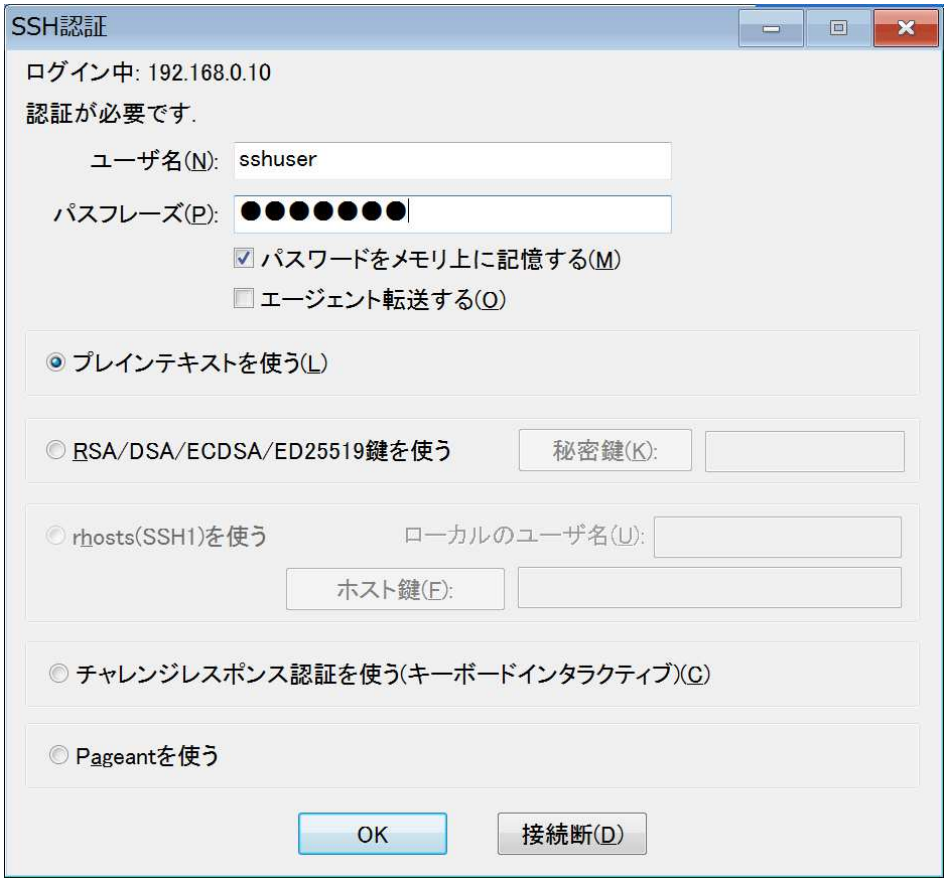


図 4　ユーザ名とパスワードを入力します

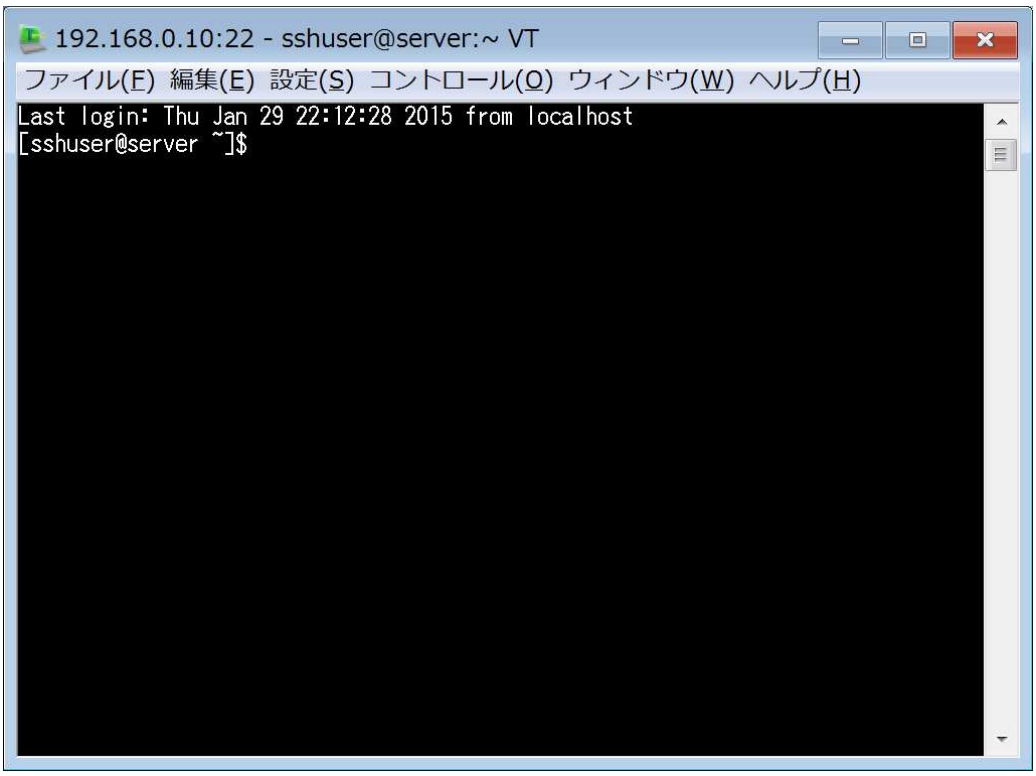


図 5　ターミナル画面が表示されます

1.4.15　Tera Term を使った公開鍵・秘密鍵の作成

Tera Term の公開鍵・秘密鍵の作成機能を使って鍵を作成し、サーバに転送して公開鍵認証を行えるようにします。

1. 鍵生成ダイアログを呼び出します。

図 6　「生成」ボタンをクリックします

Tera Term のターミナル画面の「設定」メニューから「SSH 鍵生成」を選択します。サーバに接続していない場合には、「新しい接続」ダイアログのキャンセルボタンをクリックすれば、サーバ接続を行わずにターミナル画面を表示できます。「TTSSH: 鍵生成」ダイアログが表示されるので「生成」ボタンをクリックします。

＃ 2

2. 公開鍵と秘密鍵を保存します。

図 7　パスフレーズを入力して、公開鍵と秘密鍵をそれぞれ保存します

パスフレーズを入力し、「公開鍵の保存」ボタン、「秘密鍵の保存」ボタンをクリックして、それぞれの鍵ファイルを保存します。

＃ 3

3. 鍵生成ダイアログを閉じます。

保存後、「キャンセル」ボタンをクリックしてダイアログを閉じます。

1.4.16　Tera Term を使ったファイル転送

Tera Term は SSH SCP 機能でファイルの送受信が行えます。作成した公開鍵をサーバに設置するために、公開鍵（id_rsa.pub）をサーバにコピーします。

1. Secure File Copy ダイアログを呼び出します。

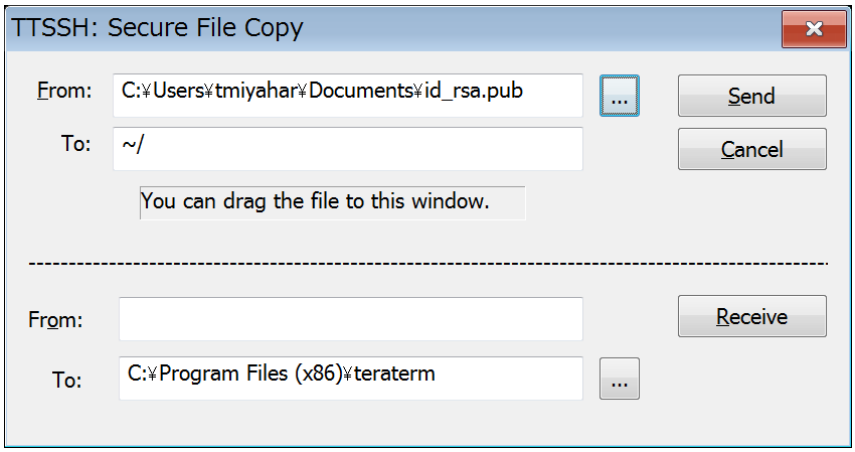


図 8　「From:」に作成した公開鍵を指定します

TeraTerm でサーバにログインした状態のまま、「ファイル」メニューから「SSH SCP」を選択します。

＃ 2

2. 公開鍵ファイルを選択します。「TTSSH: Secure File Copy」ダイアログが表示されます。上側の「From:」の右横にある「...」ボタンをクリックしてファイルダイアログを開き、保存した公開鍵ファイル（id_rsa.pub）ファイルを選択します。
3. 公開鍵ファイルをコピーします。「Send」ボタンをクリックすると、ファイルがサーバ側のユーザのホームディレクトリにコピーされます。
4. 公開鍵ファイルを確認します。

```
[sshuser@server ~]$ ls
id_rsa.pub  sftptestfile  testdir
```

5. 公開鍵ファイルを設置します。コピーした公開鍵は、Linux での公開鍵の設置と同じ手順で authorized_keys に追加しておきます。以下は、初めてサーバに公開鍵を登録する場合のコマンド例です。

```
[sshuser@server ~]$ mkdir .ssh
[sshuser@server ~]$ chmod 700 .ssh
[sshuser@server ~]$ touch .ssh/authorized_keys
[sshuser@server ~]$ chmod 600 .ssh/authorized_keys
[sshuser@server ~]$ cat id_rsa.pub >>  .ssh/authorized_keys
```

1.4.17　Tera Term を使った Windows クライアントからの公開鍵認証による接続

Tera Term を使って、公開鍵認証でサーバに接続します。

1. Tera Term を起動し、サーバに接続します。
2.「SSH 認証」ダイアログで、ユーザ名、パスフレーズ（秘密鍵に設定したもの）を入力します。
3.「RSA/DSA/EC DSA 鍵を使う」を選択し、「秘密鍵」ボタンをクリックして保存しておいた秘密鍵ファイル（id_rsa）を選択し、「OK」ボタンをクリックします。

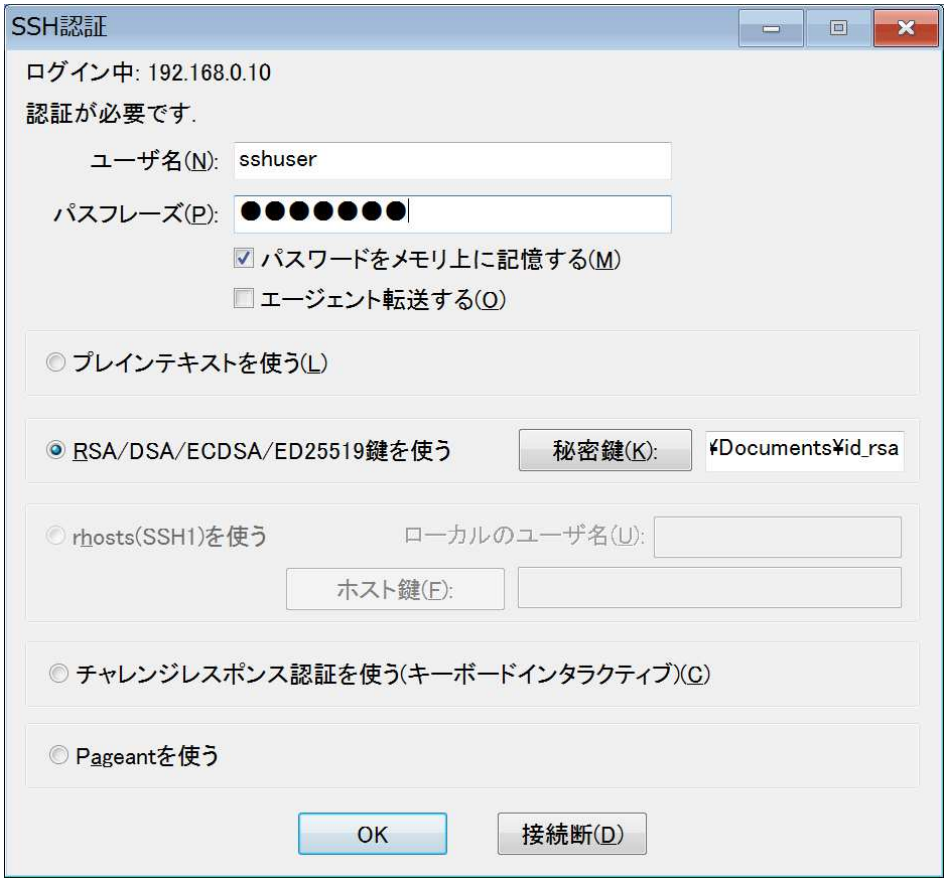


図 9　「RSA/DSA/EC DSA 鍵を使う」を選択して、秘密鍵を指定します

これで、TeraTerm を使って公開鍵認証でログインができました。

1.4.18　パスワード認証の禁止と管理者ユーザ root のログインの禁止

公開鍵認証による接続ができるようになったら、OpenSSH サーバの設定を変更してパスワード認証による接続を禁止しておきます。

OpenSSH サーバの設定ファイル/etc/ssh/sshd_config を以下のように設定変更します。

```
[root@server ~]# vi /etc/ssh/sshd_config

PasswordAuthentication ※no ←no に変更
```

また、管理者ユーザ root の外部からの直接ログインを禁止することもできます。root ユーザの直接ログインを許すかどうかは後述します。root ユーザの SSH ログインを禁止するには、以下のように変更します。

```
PermitRootLogin ※no ←no に変更
```

設定を保存したら、service コマンドで sshd を再起動します。

```
[root@server ~]# service sshd restart
sshd を停止中:                                             [  OK  ]
sshd を起動中:                                             [  OK  ]
```

これで、外部からのパスワード認証を使ったログインが禁止され、かつ root ユーザでの SSH ログインが禁止されました。

1.5 root 権限の管理

管理者ユーザである root は最も高い権限を持っているアカウントとなるため、管理方法には注意が必要です。root 権限を取得するには、以下の 3 つの方法があります。

- root で直接ログインする
- 一般ユーザでログインした後、su コマンドを実行して管理者ユーザ root に切り替える
- 一般ユーザでログインした後、sudo コマンドを使って root 権限でコマンドを実行する

どの方法も一長一短がありますが、root で直接ログインするのは許さず、一般ユーザでログインした後、su コマンドか sudo コマンドを使用させることが多いようです。

また、root 権限を使った作業を中断して席を離れる際には、ログアウトするか画面をロックするなどして、他人に勝手に操作されないようにするなど十分注意を払う必要があります。もちろん、一般ユーザでのログイン時も同様に気を付けましょう。

サーバの設置場所も重要です。サーバに物理的にアクセスされてしまうと、システムの様々なセキュリティ対策も用を為さなくなってしまいます。サーバはロックのかかったマシンルームやサーバラック内に設置し、許可された作業者のみアクセスできるようにすることが望ましいでしょう。

1.5.1 root ユーザで直接ログインする

root ユーザで直接ログインすると、まず誰がログインしたのかログに残らなくない問題があります。たとえば last コマンドを実行すると、以下のようにログインしたのは root ユーザであることが分かりますが、作業者は誰なのか分かりません。

```
# last
root     ttyS0                        Mon Aug 11 12:56   still logged in
root     ttyS0                        Mon Aug 11 12:23 - 12:56  (00:32)
root     ttyS0                        Mon Aug 11 01:11 - 12:23  (11:11)
```

また、root パスワードの管理も煩雑になります。たとえば、担当者が異動や退職したタイミングで全部のサーバの root パスワードを変更すべきですし、サーバの管理台数が多くなると root パスワードを変更するための作業に大変な手間がかかります。

OpenSSH の設定で root ユーザの直接ログインを許すと、パスワードの総当たり攻撃 (ブルートフォース攻撃) の標的とされてしまう危険性もあります。SSH 接続のパスワード認証を禁止にし、SSH 公開鍵認証方式を利用する、ファイアウォールで OpenSSH サーバに対して接続できる IP アドレスを制限する等の対策が必要です。

1.5.2 一般ユーザから su コマンドでユーザ root に切り替える

一般ユーザでログインした後、su コマンドを実行して管理者ユーザ root に切り替えると、どのユーザが root ユーザに切り替えたかログで確認できます。

以下の例では、uid が 501 のユーザ suzuki が su コマンドを実行したことが分かります。

```
$ su -
パスワード:
# tail /var/log/secure
（略）
Jan  6 11:33:55 server su: pam_unix(su-l:session): session opened for user root by
    suzuki(uid=501)
```

ただし、root として実行したコマンドのログは記録されないので、複数の管理者が root として作業すると、ファイルの削除やシステムの設定変更などが行われても、誰がいつ行ったのか後からログで調査することが難しくなってしまいます。

su コマンドは、システムの初期設定時など一人のユーザが操作を行い、操作のログを残しておく必要が無いような場合に向いているといえます。

1.5.3 su コマンドを実行できるユーザを制限する

su コマンドは root パスワードを知っているユーザなら誰でも root になれる事が問題になる場合があります。PAM（Pluggable Authentication Modules）の設定を変更して、su コマンドを実行できるユーザを制限します。

wheel グループに所属しているユーザのみ su コマンドを実行して root に切り替えられるように設定します。

PAM の設定ファイル/etc/pam.d/su を vi エディタで開いて、行頭のコメントアウトを 2 カ所外します。上の設定行は、wheel グループに所属しているユーザはパスワード無しで su コマンドを実行できる、という設定です。下の設定行は、wheel グループに所属しているユーザのみ su コマンドを実行できる、という設定です。

```
#  vi /etc/pam.d/su

#%PAM-1.0
auth            sufficient      pam_rootok.so
# Uncomment the following line to implicitly trust users in the "wheel" group.
auth           sufficient      pam_wheel.so trust use_uid ※←行頭の#を削除
# Uncomment the following line to require a user to be in the "wheel" group.
auth           required        pam_wheel.so use_uid ※←行頭の#を削除
auth            include         system-auth
account         sufficient      pam_succeed_if.so uid = 0 use_uid quiet
account         include         system-auth
password        include         system-auth
session         include         system-auth
session         optional        pam_xauth.so
```

設定変更はすぐに反映されるので、システムの再起動などは必要ありません。

確認のため、一般ユーザ suzuki で su コマンドを実行してみます。正しい root のパスワードを入力しても、root の切り替えることができません。

```
$ id suzuki
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki),5000(eigyou)
$ su -
パスワード:
su: パスワードが違います
```

root ユーザで gpasswd コマンドを実行して、ユーザ suzuki を wheel グループに所属させます。

```
# gpasswd -a suzuki wheel
Adding user suzuki to group wheel
```

所属グループはログイン時に設定されます。ユーザ suzuki でログインし直して、再度 su コマンドを実行します。今度はパスワード無しで root に切り替わります。

```
$ id suzuki
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki),10(wheel),5000(eigyou)
$ su -
[root@server ~]#
```

1.5.4 一般ユーザが sudo コマンドを実行する

sudo コマンドを使うと、一般ユーザはコマンドを root 権限で実行できます。特定のユーザやグループに対して、特定のコマンドのみ実行できるように設定するなど細かい制御ができます。su コマンドと異なり、sudo コマンドの実行には、実行したユーザのパスワードを入力して認証を行う必要があります。

sudo コマンドは実行履歴がログに残るので、後からいつ、誰が、どのコマンドを実行したのか調査できる点にメリットがあります。また、実行時の認証が実行ユーザのパスワードなので、root ユーザのパスワードを共有したり、管理したりする必要がなくなります。

CentOS では、デフォルトでは一般ユーザは sudo コマンドを実行する権限が与えられていません。

```
$ id suzuki
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki),10(wheel),5000(eigyou)
    context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
$ sudo cat /etc/shadow

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for suzuki: ※ユーザsuzukiのパスワードを入力
suzuki は sudoers ファイル内にありません。この事象は記録・報告されます。
```

sudo コマンドの設定を行って、wheel グループに所属しているユーザは sudo コマンドを実行できるように設定します。管理者ユーザ root で visudo コマンドを実行し、設定ファイル/etc/sudoers を編集して、wheel グループに所属しているユーザのみ sudo コマンドを使えるように設定します。

```
# visudo
```

下記の行の行頭のコメントを外して有効にします。「%wheel」は wheel グループを指定し、「ALL=(ALL) ALL」ですべてのコマンドを実行可能としています。

```
%wheel ALL=(ALL)       ALL ※←行頭の#を削除
```

visudo コマンドは vi エディタを呼び出しただけですので、「:wq」と入力して編集を終了します。終了時に文法のチェックが行われ、間違いがあった場合にはエラーが表示されます。

確認のため、sudo コマンドで useradd コマンドを実行してユーザ作成を行います。

```
$ sudo useradd testuser
[sudo] password for suzuki: ※ユーザsuzukiのパスワードを入力
[suzuki@server ~]$ id testuser
uid=503(testuser) gid=503(testuser) 所属グループ=503(testuser)
```

1.5.5 sudo で実行できるコマンドの制限

sudo コマンドでは、ある特定のグループに対して、一部のコマンドのみ実行できるように制限できます。

たとえば、グループ webadm に所属しているユーザに対して Web サーバ（httpd）の起動や停止、再起動ができるような権限を付与するには、visudo を実行して以下の 1 行を追加します。実行可能なコマンドをカンマ区切りで記述していきます。

```
$ sudo visudo

%webadm        ALL=NOPASSWD:  /sbin/service httpd start, /sbin/service httpd stop,
     /sbin/service httpd restart
```

webadm グループに所属するユーザアカウントを作成します。useradd コマンドに-G（大文字）オプションを付与して実行すると、ユーザアカウント作成時にサブグループへの所属を指定できます。

```
$ sudo groupadd webadm
$ sudo useradd -G webadm httpdtest
```

su -コマンドで作成したユーザ httpdtest に切り替えます。

```
$ sudo su - httpdtest
$ id
uid=504(httpdtest) gid=504(httpdtest) 所属グループ=504(httpdtest),5001(webadm)
```

sudo コマンドを使って Web サーバ を起動します。

```
$ sudo service httpd start
httpd を起動中: httpd: Could not reliably determine the server's fully qualified domain
     name, using 192.168.0.10 for ServerName
                                                           [  OK  ]
```

警告が出ていますが、ここでは無視して構いません。

Web サーバが起動したか、プロセスを確認します。

```
$ ps ax | grep httpd
28608 pts/0    S      0:00 su - httpdtest
31175 ?        Ss     0:00 /usr/sbin/httpd
31176 ?        S      0:00 /usr/sbin/httpd
31177 ?        S      0:00 /usr/sbin/httpd
31179 ?        S      0:00 /usr/sbin/httpd
31180 ?        S      0:00 /usr/sbin/httpd
31181 ?        S      0:00 /usr/sbin/httpd
31182 ?        S      0:00 /usr/sbin/httpd
31183 ?        S      0:00 /usr/sbin/httpd
31184 ?        S      0:00 /usr/sbin/httpd
31198 pts/0    S+     0:00 grep httpd
```

Web サーバの停止を行います。

```
$ sudo service httpd stop
httpd を停止中:                                            [  OK  ]
$ ps ax | grep httpd
28608 pts/0    S      0:00 su - httpdtest
31325 pts/0    S+     0:00 grep httpd
```

一般ユーザで Web サーバの起動や停止ができるようになりました。#ネットワークの管理

1.6 ネットワークインターフェイスの設定

ネットワークインターフェースには、IP アドレスなどネットワーク通信のための各種設定が必要となります。ネットワーク環境に合わせて設定を変更します。

1.6.1 ネットワーク設定の確認

ifconfig を使ってネットワーク設定の確認を行います。ifconfig コマンドは、ネットワークインターフェースの状態を表示します。

```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:1C:42:DC:25:92
          inet addr:192.168.0.10  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::21c:42ff:fedc:2592/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:6267 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3120 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:706436 (689.8 KiB)  TX bytes:472809 (461.7 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:45 errors:0 dropped:0 overruns:0 frame:0
          TX packets:45 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:5792 (5.6 KiB)  TX bytes:5792 (5.6 KiB)
```

1.6.2 デフォルトゲートウェイの確認

デフォルトゲートウェイの確認には route コマンドまたは netstat -rn コマンドを使います。

```
# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.0     *               255.255.255.0   U     1      0        0 eth0
default         192.168.0.1     0.0.0.0         UG    0      0        0 eth0
```

Destination が default になっている行がデフォルトゲートウェイの設定です。この例では、192.168.0.1 がネットワークインターフェース eth0 から通信する場合のデフォルトゲートウェイとして設定されています。

1.6.3 ネットワークインターフェースの設定ファイル

ネットワークインターフェースの設定ファイルは、/etc/sysconfig/network-scripts ディレクトリ以下に格納されています。ファイル名は「ifcfg-ネットワークインターフェース名」です。たとえば、最初のネットワークインターフェースは eth0 というネットワークインターフェース名で認識されるので、設定ファイルは ifcfg-eth0 となります。

```
# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
TYPE=Ethernet
UUID=c9eaa5e8-a31a-4d36-8dc7-2fc8de8350b3
ONBOOT=yes
NM_CONTROLLED=yes
BOOTPROTO=none
HWADDR=00:1C:42:DC:25:92
IPADDR=192.168.0.10
PREFIX=24
GATEWAY=192.168.0.1
DNS1=192.168.0.1
DEFROUTE=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME="System eth0"
```

1.6.4 ip コマンドを使ったネットワークインターフェースの設定

ip コマンドは、ネットワークインターフェースの状態の確認や設定、ルーティングテーブルの表示や追加、削除、ARP テーブルの確認や削除など、ネットワークにおける操作全般を行うことができます。CentOS では今後、ifconfig/route/arp/netstat コマンドなど net-tools パッケージに含まれていたコマンドは標準ではなくなり、ip コマンドに置き換わっていきます。

■IP アドレス、MAC アドレスの確認

IP アドレスや MAC アドレスの確認には ip address show コマンドを実行します。これは従来の ifconfig コマンドに相当します。

```
# ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:1c:42:dc:25:92 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.10/24 brd 192.168.0.255 scope global eth0
    inet6 fe80::21c:42ff:fedc:2592/64 scope link
       valid_lft forever preferred_lft forever
```

■ルーティングテーブルの確認

ルーティングテーブルの確認を行うには ip route show コマンドを実行します。これは従来の route コマンドに相当します。

```
# ip route show
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.10  metric 1
default via 192.168.0.1 dev eth0  proto static
```

■ARP テーブルの確認

ARP テーブルの確認を行うには ip neighbor show コマンドを実行します。これは従来の arp コマンドに相当します。neighbor は neigh に省略できます。

```
#  ip neigh show
192.168.0.1 dev eth0 lladdr 00:1c:42:00:00:18 STALE
192.168.0.2 dev eth0 lladdr 00:1c:42:00:00:08 REACHABLE
```

1.6.5 netstat コマンドを使った設定確認

netstat コマンドはネットワークの各種状態を確認することができます。よく使用するオプションは以下の通りです。

オプション	説明
-i	ネットワークインターフェースの統計情報を表示
-n	コンピューター名の名前解決をせずに IP アドレスで表示
-a	すべての接続を表示
-l	リッスンしているポートの統計情報を表示
-t	TCP の統計情報を表示
-u	UDP の統計情報を表示

■ネットワークインターフェースの統計情報の表示

netstat -i コマンドは、各ネットワークインターフェースのパケット転送量を分かりやすく表示します。

```
# netstat -i
Kernel Interface table
Iface       MTU Met    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0       1500   0    47780      0      0      0    16784      0      0      0 BMRU
lo        65536   0     2366      0      0      0     2366      0      0      0 LRU
```

■TCP 通信の状態の表示

すべての TCP 通信の状態を表示したい場合には、netstat -nat コマンドを実行します。

```
# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
tcp        0      0 0.0.0.0:22                  0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:631               0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:25                0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:37729               0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:111                 0.0.0.0:*                   LISTEN
tcp        0      0 :::22                       :::*                        LISTEN
tcp        0      0 ::1:631                     :::*                        LISTEN
tcp        0      0 ::1:25                      :::*                        LISTEN
tcp        0      0 :::37114                    :::*                        LISTEN
tcp        0      0 :::111                      :::*                        LISTEN
```

■待ち受け TCP ポートの表示

待ち受けているすべての TCP のポートを表示したい場合には、netstat -nlt コマンドを実行します。

```
# netstat -nlt
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
```

```
tcp        0      0 0.0.0.0:22                  0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:631               0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:25                0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:37729               0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:111                 0.0.0.0:*                   LISTEN
tcp        0      0 :::22                       :::*                        LISTEN
tcp        0      0 ::1:631                     :::*                        LISTEN
tcp        0      0 ::1:25                      :::*                        LISTEN
tcp        0      0 :::37114                    :::*                        LISTEN
tcp        0      0 :::111                      :::*                        LISTEN
```

■待ち受け UDP ポートの表示

待ち受けているすべての UDP のポートを表示したい場合には、netstat -nlu コマンドを実行します。

```
# netstat -nlu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
udp        0      0 0.0.0.0:68                  0.0.0.0:*
udp        0      0 127.0.0.1:708               0.0.0.0:*
udp        0      0 0.0.0.0:111                 0.0.0.0:*
udp        0      0 0.0.0.0:631                 0.0.0.0:*
udp        0      0 192.168.0.10:123            0.0.0.0:*
udp        0      0 127.0.0.1:123               0.0.0.0:*
udp        0      0 0.0.0.0:123                 0.0.0.0:*
udp        0      0 0.0.0.0:44415               0.0.0.0:*
udp        0      0 0.0.0.0:655                 0.0.0.0:*
udp        0      0 :::111                      :::*
udp        0      0 fe80::21c:42ff:fedc:2592:123 :::*
udp        0      0 ::1:123                     :::*
udp        0      0 :::123                      :::*
udp        0      0 :::39182                    :::*
udp        0      0 :::655                      :::*
```

1.6.6 ping コマンドを使用した疎通の確認

リモートのホストに IP のパケットが到達できるか確認するためには ping コマンドを実行します。Ctrl+C を実行するまで無制限に実行されます。

```
# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=128 time=6.26 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=128 time=3.28 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=128 time=2.85 ms
※^C Ctrl+Cキーを入力する
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 62.780/64.980/66.416/1.579 ms
```

-c オプションで実行回数を指定できます。以下の例では、5 回だけ疎通確認を行っています。

```
# ping -c 5 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
```

```
64 bytes from 8.8.8.8: icmp_seq=1 ttl=128 time=3.39 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=128 time=3.12 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=128 time=3.44 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=128 time=2.85 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=128 time=3.10 ms

--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4012ms
rtt min/avg/max/mdev = 2.856/3.185/3.440/0.225 ms
```

ping コマンドは ICMP プロトコルを使いますので、経路の途中にあるルーターやファイアーウォールなどで ICMP プロトコルがブロックされていると、ping コマンドを実行してもうまく結果が返ってこない場合があります。また、対象となるリモートのホストが ping コマンドに反応を返さない場合もあります。

レスポンス結果 (RTT) の目安としては、同じネットワークセグメント上のホストの場合は 1ms(ミリ秒) 以内、国内のインターネット上の他のホストの場合、10ms〜30ms、地球の裏側で 500ms 程度かかかります。

1.6.7 ethtool コマンドを使ったネットワークインターフェース情報の確認

ethtool コマンドは、ネットワークインターフェースに対するハードウェアスペックやファームウェア、リンク状態、リンクスピードなどの確認、およびアクセラレーション機能の有効化・無効化を制御することができます。

ネットワークインターフェースに対するハードウェアスペックやファームウェア、リンク状態、リンクスピードなどの確認をするには ethtool コマンドを実行します。

```
# ethtool eth0
Settings for eth0:
   Supported ports: [ TP ]
   Supported link modes:   10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Full
   Supported pause frame use: No
   Supports auto-negotiation: Yes
   Advertised link modes:  10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Full
   Advertised pause frame use: No
   Advertised auto-negotiation: Yes
   Speed: 1000Mb/s
   Duplex: Full
   Port: Twisted Pair
   PHYAD: 1
   Transceiver: internal
   Auto-negotiation: on
   MDI-X: Unknown
   Supports Wake-on: g
   Wake-on: g
   Link detected: yes
```

ファームウェアのバージョンなどを確認するには ethtool -i コマンドを実行します。

```
# ethtool -i eth0
driver: bnx2
version: 2.2.3
firmware-version: bc 4.6.4 NCSI 1.0.3
```

```
bus-info: 0000:02:00.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: yes
supports-register-dump: yes
supports-priv-flags: no
```

仮想マシンが利用している仮想ネットワークインターフェースの場合、多くの情報が取得できない場合があります。

```
# ethtool eth0
Settings for eth0:
    Link detected: yes
```

```
# ethtool -i eth0
driver: virtio_net
version:
firmware-version:
bus-info: virtio0
supports-statistics: no
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```

1.7 network サービスと NetworkManager

CentOS 6 上のネットワークの管理は、network サービス、もしくは NetworkManager サービスで行います。network サービスは、Linux で従来から使われているネットワーク管理の仕組みです。起動時に設定ファイルからネットワークの情報を読み込み、ネットワークインターフェースに対して IP アドレスなどの設定を行ったり、デフォルトゲートウェイや DNS サーバの情報を起動時に反映させます。中身はシェルスクリプトの集まりです。

一方、NetworkManager は Linux におけるネットワークの管理を抽象化して新たに作った仕組みです。また、NetworkManager は、Linux のプロセス間通信でよく使用される D-Bus の API を持っており、ネットワークを利用するアプリケーションと連携を行うことができます。

CentOS 6 は NeworkManager を標準で使用していますが、Minimal でインストールした場合や、NetworkManager に対応していないアプリケーションを使うために network サービスを使用したい場合には、NetworkManager を停止し、network サービスを有効にします。

1.7.1 NetworkManager の停止と network サービスの有効化

network サービスを有効化したい場合には、事前に NetworkManager を停止、無効化した上で、network サービスを起動、有効化します。なお、これらの作業は SSH にてネットワーク経由でログインしている時には行わないでください。接続が切れてしまいシステムを制御できなくなる場合があります。

service コマンドを使って NetworkManager サービスを停止し、chkconfig コマンドでシステム起動時の NetworkManager サービスの自動実行を無効にしておきます。

```
# service NetworkManager stop
NetworkManager デーモンを停止中:                          [  OK  ]
# chkconfig NetworkManager off
# chkconfig --list NetworkManager
NetworkManager  0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

service コマンドで network サービスを実行し、chkconfig コマンドでシステム起動時の network サービスの自動実行を有効にしておきます。

```
# service network start
ループバックインターフェイスを呼び込み中                    [  OK  ]
インターフェース eth0 を活性化中:                           [  OK  ]
# chkconfig network on
# chkconfig --list network
network         0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

1.8 各種ネットワーク設定ファイル

Linux のネットワーク関連の設定は、いくつかの設定ファイルに分散して設定されています。

1.8.1 ネットワーク設定に関する情報を指定/etc/sysconfig/network

設定ファイル/etc/sysconfig/network には、ネットワークの有効化やホスト名などの情報が含まれています。

```
# cat /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=server.example.com
NTPSERVERARGS=iburst
```

ホスト名を変更したい場合には、HOSTNAME の値を変更して、システムを再起動します。

1.8.2 静的な名前解決/etc/hosts

設定ファイル/etc/hosts には、静的な名前解決のために IP アドレスとホスト名が対になって列挙された情報が記述されています。

```
# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.0.10    server.example.com server
192.168.0.101   client.example.com client
```

1.8.3 参照 DNS 設定ファイル/etc/resolv.conf

設定ファイル/etc/resolv.conf には、DNS を使って名前解決を行う場合に参照する DNS サーバの情報が記述されています。先に記述されてある DNS サーバを優先 DNS サーバとして最初に参照します。その優先 DNS サーバが応答しない場合には、次に記述されている代替 DNS サーバを参照します。

```
# cat /etc/resolv.conf
# Generated by NetworkManager
search example.com
nameserver 192.168.0.1
```

■/etc/resolv.conf は直接編集しない

/etc/resolv.conf ファイルを直接編集して、参照する DNS の設定を変更することは推奨されていません。なぜなら、NetworkManager や network サービス、DHCP クライアントの設定ファイルなどにより、/etc/resolv.conf の設定情報は自動的に変更されます。そのため、/etc/resolv.conf を手動で変更後にシステムや各サービスを再起動すると、/etc/resolv.conf の設定が古い設定で上書きされてしまい、名前解決ができなくなるなど予期せぬトラブルが発生してしまいます。

DNS サーバの設定は、インターフェース設定ファイルに記述しておきます。ネットワークインターフェースが有効になる際に、記述しておいた値に基づいて/etc/resolv.conf ファイルが設定されます。

/etc/sysconfig/network-scripts/ifcfg-eth0 に以下を追記します。DNS1 で優先 DNS サーバ、DNS2 で代替 DNS サーバを指定します。

```
DNS1=192.168.0.1
DNS2=192.168.0.2
```

この設定は、システム再起動時、または NetworkManager サービス、あるいは network サービスの再起動を行うと/etc/resolv.conf に反映されます。

1.8.4 名前解決設定ファイル/etc/nsswitch.conf

設定ファイル/etc/nsswitch.conf には、名前解決などを行う場合に参照する仕組みのリストと優先順位が記述されています。/etc/hosts を参照するのか、DNS や NIS を参照させるのかなどが細かく設定できます。

```
# cat /etc/nsswitch.conf
 (略)
#hosts:     db files nisplus nis dns
hosts:      files dns
 (略)
```

ホストの名前解決は、左から順に「files」、「dns」で優先度が記述されています。まず設定ファイル/etc/hosts を参照し、次に DNS を参照して名前解決を行います。この設定ファイルには他に、ユーザ認証を行う際の参照先の指定なども記述されます。

1.8.5 ポート番号とサービスの対応リスト/etc/services

設定ファイル/etc/services には、各種 TCP/UDP のポート番号と対応するサービスの名前が記述されています。

たとえば、HTTP プロトコルは以下のように記述されています。

```
http            80/tcp          www www-http    # WorldWideWeb HTTP
```

各種コマンドがポート番号を表示する際、TCP のポート番号 80 番を表示する時にはプロトコル名に置き換えて http と表示します。netstat の-n オプションは表示を数値で表示し、-n オプションが指定されないとプロトコル名などを名前で表示します。

この設定ファイルは、あくまで各種コマンドがポート番号をプロトコル名に置き換えて表示するために参照されます。実際には他のプロトコルがポートを使用している場合もあります。

```
# netstat -nat | grep 80
tcp        0      0 :::80                       :::*                        LISTEN
# netstat -at | grep http
tcp        0      0 *:http                      *:*                         LISTEN
```

ポート番号 80 が http に置き換えられているのが分かります。ポート番号が数字のまま表示される場合があるのは、/etc/services に記述が無いためです。また、1 つ目の例は表示が IPv6 での表示になっていますが、これは IPv6 が有効な場合の Apache Web サーバの動作によるものです。この状態でも IPv4 で接続できます。

1.8.6 プロトコル定義ファイル/etc/protocols

設定ファイル/etc/protocols には各種プロトコルの名前とプロトコル番号が記述されています。たとえば、よく使用しているプロトコル番号は以下の通りです。

```
ip  0   IP      # internet protocol, pseudo protocol number
icmp    1   ICMP        # internet control message protocol
tcp 6   TCP     # transmission control protocol
udp 17  UDP     # user datagram protocol
```

1.9 iptables によるパケットフィルタリング

iptables は Linux カーネルに実装されたパケットフィルタリングの仕組みです。パケットフィルタリングとは、ネットワークに対してどのようなパケットの通過を許可および拒否するか判定する機能のことです。ファイアーウォールの基本的な機能であるため、ファイアーウォール機能と説明される場合もあります。iptables は、カーネル内の NF(netfilter) によって実装されており、ユーザーランドのパケットフィルタリングのルール定義を行うコマンドとして iptables コマンドが用意されています。

1.9.1 iptables の NAT 機能

iptables にはパケットフィルタリング機能の他に、NAT(Network Address Translation) というパケットの送信元または宛先の IP アドレスを変換する機能があります。NAT には以下の種類があります。ここでは内部ネットワークをプライベート IP アドレスが割り振られた LAN、外部ネットワークをグローバル IP アドレスが割り振られたインターネットを想定して説明します。

■スタティック NAT

内部ホストの IP アドレスと外部向け IP アドレスを 1 対 1 で結びつけます。内部から外部へアクセスするパケットの送信元 IP アドレスを、内部ホストの IP アドレスから結びつけられている外部向け IP アドレスに書き換えて通信を行います。外部と通信できるホストは用意された外部向け IP アドレスの数とあらかじめ決まっています。外部から内部への通信を許可して、外部から内部のホストへアクセスさせることもできます。

■ダイナミック NAT

複数の内部ホストの IP アドレスと複数の外部向け IP アドレスを N 対 N で結びつけます。内部から外部へアクセスするパケットの送信元 IP アドレスを、内部ホストの IP アドレスから外部向け IP アドレスのプールから選択された IP アドレスに書き換えて通信を行います。IP アドレスの対応付けは通信開始時に行われるので、外部向け IP アドレスが余っていないと通信が行えませんが、他のホストが通信を終了して外部向け IP アドレスが解放されると通信が行えるようになります。内部のホストの数が多い場合には、外部向け IP アドレスが不足することになるので、次の NAPT を使用した方が外部との通信が行いやすいでしょう。

■NAPT(IP マスカレード)

複数の内部ホストの IP アドレスと 1 つの外部向け IP アドレスを結びつけます。内部向け IP アドレスは外部に出る際に外部向け IP アドレスに書き換えられて通信を行います。その際に NAPT ではポート番号の変換も行います。ポート番号は最大 65535 番まであるので、1 つの外部 IP アドレスで沢山の内部ホストを外部と通信させることができます。

1.9.2 iptables の起動と停止

service コマンドで iptables の起動と停止が行えます。

```
# service iptables start
iptables: チェインをポリシー ACCEPT へ設定中filter          [  OK  ]
iptables: ファイアウォールルールを消去中:                    [  OK  ]
iptables: モジュールを取り外し中:                            [  OK  ]
iptables: ファイアウォールルールを適用中:                    [  OK  ]
```

service コマンドで iptables を停止します。

```
# service iptables stop
iptables: チェインをポリシー ACCEPT へ設定中filter          [  OK  ]
iptables: ファイアウォールルールを消去中:                    [  OK  ]
iptables: モジュールを取り外し中:                            [  OK  ]
```

1.9.3 iptables のステータス確認

service コマンドで iptables のステータスを確認します。

```
# service iptables start
iptables: ファイアウォールルールを適用中:                    [  OK  ]
# service iptables status
テーブル: filter
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state
     RELATED,ESTABLISHED
2    ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
3    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
4    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
5    REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with
     icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination
1    REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with
     icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
```

iptables -L コマンドでも iptables のステータスの確認ができます。

```
# iptables -L
```

iptables-save コマンドは、iptables の設定を iptables コマンドの設定オプションの形式で出力します。

```
# iptables-save
# Generated by iptables-save v1.4.7 on Fri Jan  9 16:51:47 2015
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [33:4180]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
# Completed on Fri Jan  9 16:51:47 2015
```

1.9.4 パケットフィルタリングの設定

パケットフィルタリングは iptables-A コマンドで設定します。コマンドの書式は以下の通りです。

```
iptables -A チェーン 条件 -j ターゲット
```

条件は様々なオプションで設定しますが、基本的な設定は後述します。チェーンとターゲットで設定できる値は以下の通りです。

チェーンの種類	説明
INPUT	受信パケット
OUTPUT	送信パケット
FORWARD	フォワードするパケット
PREROUTING	受信時に変換するチェーン
POSTROUTING	送信時に変換するチェーン

ターゲットの種類	説明
ACCEPT	パケットの通過を許可
DROP	パケットを破棄
REJECT [–reject-with]	パケットを拒否し、ICMP で通知
LOG	パケットの情報を syslog に出力

1.9.5 パケットの通過を許可する iptables 設定ルール

パケットの通過を許可するには、INPUT チェーンに対して許可したいプロトコルやポート番号を指定します。コマンドの書式は以下の通りです。

```
iptables -A INPUT -m tcp -p tcp --dport ポート番号 -j ACCEPT
```

以下の例では TCP プロトコルの 80 番ポート (HTTP) の通信を許可する設定を行っています。ただし、iptables のルールは設定された順番に適用されます。デフォルト設定ではすべてのパケットを REJECT するルールが設定されているため、このコマンドで一番最後に追加されたルールは実際には機能しません。

実際の運用では、後述する設定ファイル/etc/sysconfig/iptables に設定を記述して、iptables に読み込ませる方法で設定を行います。

```
# iptables -A INPUT -m tcp -p tcp --dport 80 -j ACCEPT
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
REJECT     all  --  anywhere             anywhere            reject-with
    icmp-host-prohibited ※すべて拒否
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:http ※到達しません
（略）
```

1.9.6 iptables の設定を保存する

iptables の設定をシステム起動時に再度設定したい場合には、変更内容を保存しておきます。

```
# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables: [  OK  ]
```

保存された iptables の設定ルールは/etc/sysconfig/iptables に保存されます。iptables サービスは起動時にこの設定ファイルを読み込んで設定を行います。

新しい iptables のルールを設定したい場合には、この設定ファイルを変更して iptables に読み込ませます。

1.9.7　iptables の設定ルールをリロードする

設定ファイル/etc/sysconfig/iptables を直接変更した場合は、設定ルールをリロード (再読み込み) してルールを適用する必要があります。リロードするには、service iptables reload コマンドを実行します。

iptables サービスは service iptables restart で再度設定を行うこともできますが、プロトコルによっては iptables の追加モジュールを使用している場合があり、restart すると一度モジュールがアンロードされて接続が切れてしまうことがあります。reload を指定すれば、接続を切らずに新しいルールを適用できます。

```
# service iptables reload
iptables: Trying to reload firewall rules:                 [  OK  ]
```

1.9.8　system-config-firewall-tui を使用した iptables の設定

system-config-firewall-tui は、iptable の設定を CUI で行えるツールです。

もしインストールされていなかったら、以下の通りインストールを実行します。

```
# yum install system-config-firewall-tui
```

1. system-config-firewall-tui を実行します。

```
# system-config-firewall-tui
```

＃ 2

2. カスタマイズを選択します。

図 10　「カスタマイズ」を選択します

system-config-firewall-tui を実行すると、設定画面が表示されます。ファイアウォールの設定を変更したい場合、「カスタマイズ」を選択して Enter キーを押します。選択の変更は TAB キー、またはカーソルキーで行えます。

3

3. 許可するサービスを選択します。

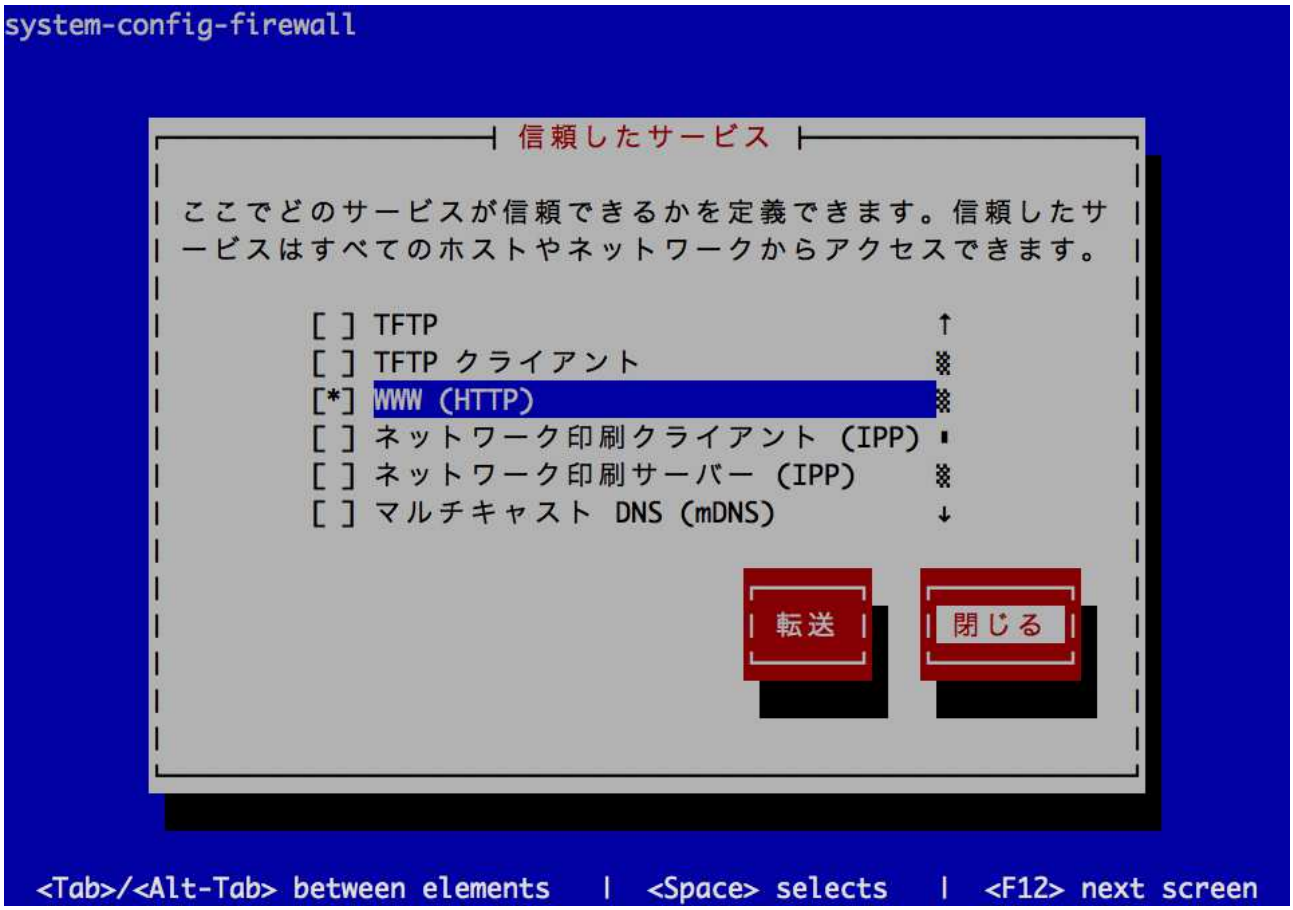


図 11 許可するサービスを選択します

カーソルキーの上下で許可したいサービスを選択し、スペースキーで有効にします。ここでは「WWW (HTTP)」を追加で有効にします。設定が終わったら「閉じる」を選択します。

4. 設定がすぐに反映されることを確認します。

元の画面に戻るので、「OK」を選択します。現在の iptables の設定を上書きするか確認されるので、「はい」を選択して終了します。

5

5. 設定を確認します。

system-config-firewall-tui で設定を変更すると、/etc/sysconfig/iptables の設定が上書きされ、現在設定されている iptables のルールも変更されます。

たとえば、WWW(HTTP) の通信を許可した場合、下記のように 80 番ポートの通信許可設定が追加されます。

```
# cat /etc/sysconfig/iptables
# Firewall configuration written by system-config-firewall
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
```

図 12　設定はすぐに反映されます

```
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

1.10　DHCP サーバの構築

DHCP は、IP アドレスを自動的に取得するためのプロトコルです。DHCP から DHCP クライアントに対して、IP アドレスなどの設定を自動的に提供する仕組みになっています。

1.10.1　DHCP サーバは 1 つだけ設置

DHCP サーバは、同一ネットワークセグメント内に 1 つだけ設置します。DHCP サーバを同一ネットワークセグメント内に複数設置すると、クライアントは先に受け取った DHCP サーバからの設定で IP アドレスなどを設定します。

DHCP プロトコルはルーターを越えることができないので、ルーターで分割された別のネットワークには別の DHCP サーバを設置できます。

たとえば、本番運用されている DHCP サーバが存在するネットワークにテスト目的のための DHCP サーバを設置すると、ユーザーが誤った IP アドレスを取得してしまうおそれがあります。このような場合には、テスト目的のホストは手動で IP アドレスを設定するか、VLAN や L3 スイッチなどを使って本番用とテスト用を別々のネットワークとして分離します。

1.10.2 DHCP サーバのインストール

DHCP サーバを動作させるには dhcp パッケージをインストールします。

```
# yum install dhcp
```

1.10.3 DHCP サーバの設定

DHCP サーバの設定ファイル/etc/dhcp/dhcpd.conf の設定は最小限、以下の記述があれば動作します。設定行の最後には「;」（セミコロン）を記述します。

```
ddns-update-style none;

subnet 192.168.0.0 netmask 255.255.255.0 {
   range 192.168.0.200 192.168.0.254;
}
```

この設定では、以下のような値を設定しています。

設定項目	意味	値
ddns-update-style	ダイナミック DNS 機能との連動	none（連動しない）
subnet	ネットワークアドレス	192.168.0.0
netmask	ネットマスク	255.255.255.0
range	開始 IP アドレスと終了 IP アドレス	192.168.0.200 から 192.168.0.254

1.10.4 その他の DHCP サーバの設定パラメーター

DHCP サーバでは、基本的な IP アドレスのほかに、以下のようなパラメーターを DHCP クライアントに対して提供できます。

```
ddns-update-style none;

subnet 192.168.0.0 netmask 255.255.255.0 {
   range 192.168.0.200 192.168.0.254;
   option routers 192.168.0.1;
   option domain-name-servers 192.168.0.1,192.168.0.2;
   default-lease-time 18000;
   max-lease-time 36000;
}
```

設定項目	意味	値
option routers	デフォルトゲートウェイを設定	192.168.0.1
option domain-name-servers	DNS サーバを指定。複数指定はカンマ区切りで記述	192.168.0.1 と 192.168.0.2
default-lease-time	デフォルトのリース時間 (秒) を指定	18000（5 時間）
max-lease-time	最大のリース時間 (秒) を指定	36000（10 時間）

リース時間が過ぎると DHCP クライアントは DHCP サーバに対して IP アドレスの再要求を行います。クライアントが IP アドレス要求時にリース時間を指定しなかった場合には、default-lease-time の時間がリース時間として設定されます。クライアントがリース時間を指定しても、max-lease-time より長い時間のリース時間は設定されません。

1.10.5 特定のクライアントに固定 IP アドレスを割り当てる

特定のクライアントに固定 IP アドレスを割り当てたい場合は、host 設定を行い、hardware ethernet でクライアントの MAC アドレスを指定し、fixed-address で固定 IP アドレスを割り当てます。

```
host client1 {
    hardware ethernet FA:16:3E:01:DB:D0;
    fixed-address 192.168.0.10;
}
```

1.10.6 DHCP サービスの開始

service コマンドで DHCP サービスを起動します。システム起動時に DHCP サービスを自動的に起動するためには、chkconfig コマンドで有効化します。

```
# service dhcpd start
# chkconfig dhcpd on
```

1.10.7 Linux での DHCP クライアントの設定

Linux クライアントで DHCP を有効にするには、ネットワークインターフェースの設定ファイルに「BOOTPROTO= dhcp」を記述します。DHCP サーバから DNS の情報を受け取った場合、デフォルトでクライアントの/etc/resolv.conf が自動的に変更される仕様になっています。/etc/resolv.conf の変更を有効にするためには PEERDNS オプションで yes を指定します。

```
$ cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE= eth0
BOOTPROTO= dhcp
ONBOOT= yes
PEERDNS=yes
```

network サービスを再起動します。

```
# service network restart
```

設定された IP アドレス、デフォルトゲートウェイ、DNS などを確認します。

```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:1C:42:D0:CA:A0
          inet addr:192.168.0.200  Bcast:192.168.0.255  Mask:255.255.255.0
 （略）
```

```
$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.0     *               255.255.255.0   U     1      0        0 eth0
default         192.168.0.1     0.0.0.0         UG    0      0        0 eth0
```

```
$ cat /etc/resolv.conf
# Generated by NetworkManager
nameserver 192.168.0.1
nameserver 192.168.0.2
```

1.10.8 Windows クライアントでの DHCP クライアントの設定

Windows クライアントで DHCP を有効にするには、ネットワークアダプターの設定を変更して、「IP アドレスを自動的に取得する」を有効にします。

1. コントロールパネルから「ネットワークと共有センター」を実行します。

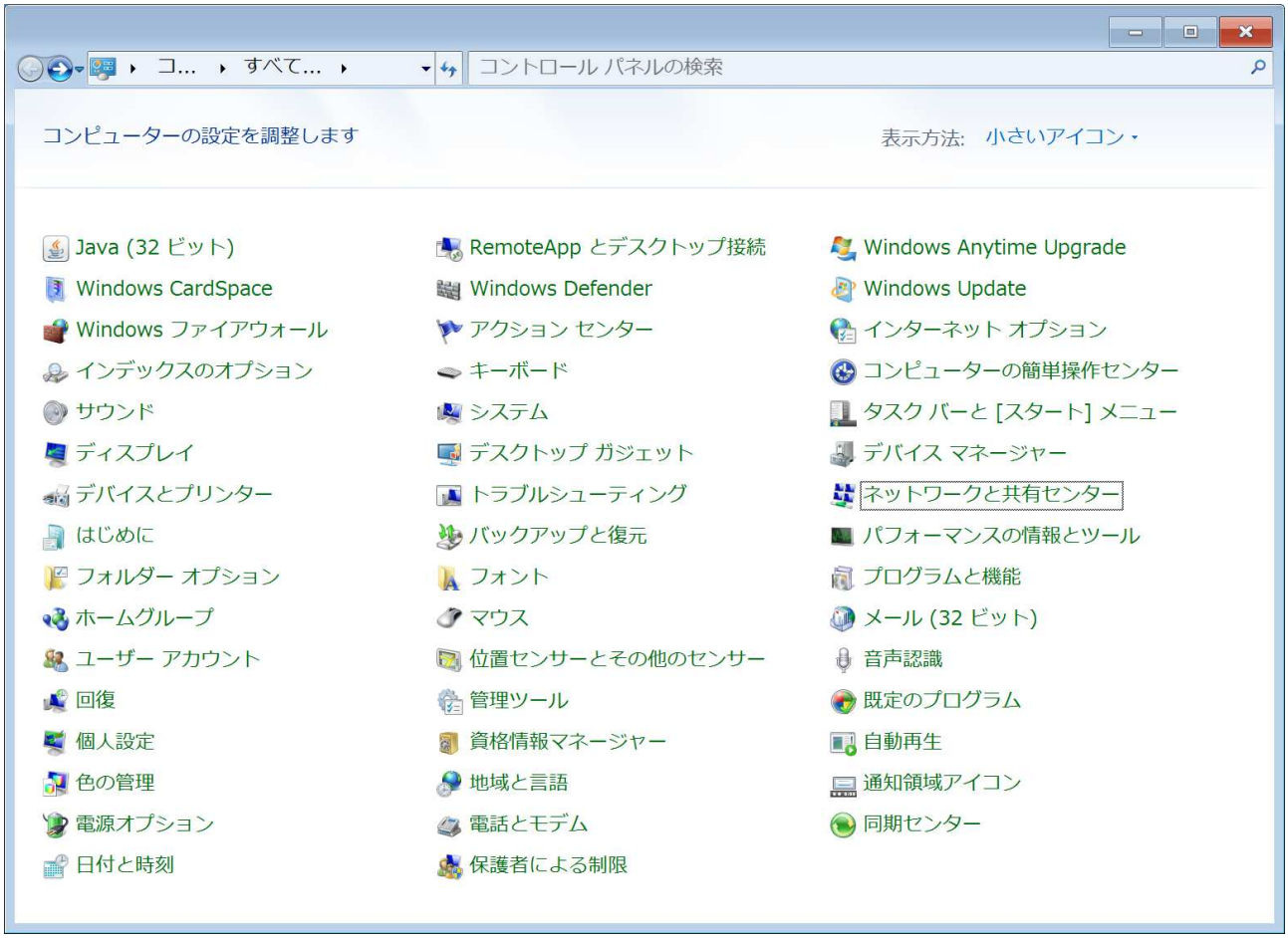


図 13 「ネットワークと共有センター」を実行します

Windows7 の場合、「コントロールパネル」から「ネットワークと共有センター」を実行します。

＃ 2

2. ネットワーク接続の一覧画面を呼び出します。

「アダプターの設定の変更」をクリックします。

＃ 3

3. アダプターのプロパティダイアログを呼び出します。

使用するネットワークアダプターを右クリックして、ポップアップメニューから「プロパティ」を選択します。

＃ 4

4. TCP/IPv4 のプロパティダイアログを呼び出します。

「インターネットプロトコルバージョン 4 (TCP/IPv4)」を選択し、「プロパティ」ボタンをクリックします。

＃ 5

5. DHCP クライアントとして設定します。

図 14　「アダプターの設定の変更」をクリックします

「IP アドレスを自動的に取得する」を選択します。DNS サーバの設定も DHCP サーバから取得する場合には、「DNS サーバのアドレスを自動的に取得する」を選択します。「OK」ボタンをクリックします。

＃ 6

6. アダプターのプロパティダイアログを閉じます。「閉じる」ボタンをクリックします。

7. 接続状態を確認します。

使用するネットワークアダプターをダブルクリックします。接続状態によって「IPv4 接続」の状態表記が変わります。

＃ 8

8. 設定の詳細を確認します。

設定された IP アドレス、デフォルトゲートウェイ、DNS などを確認するには、「詳細」ボタンをクリックします。

1.10.9　DHCP サーバが提供している IP アドレスの確認

DHCP サーバが DHCP クライアントに提供している IP アドレスの一覧は/var/lib/dhcpd/dhcpd.leases のリースデータベースファイルの中に記録されています。

```
[root@server ~]# cat /var/lib/dhcpd/dhcpd.leases
# The format of this file is documented in the dhcpd.leases(5) manual page.
# This lease file was written by isc-dhcp-4.1.1-P1

server-duid "\000\001\000\001\034H\217F\000\034B\334%\222";

lease 192.168.0.200 {
```

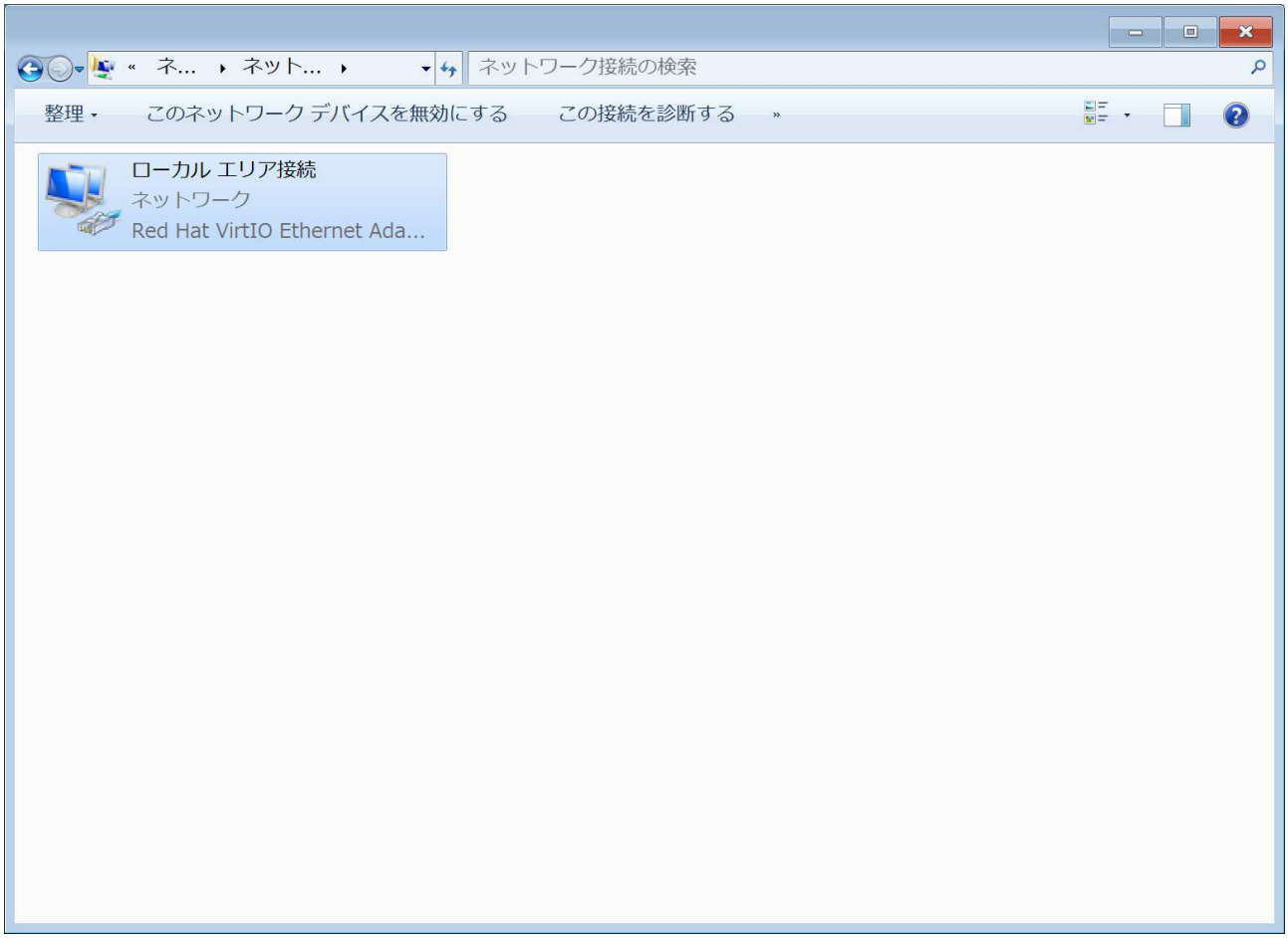


図 15　アダプターを右クリックして、プロパティを選択します

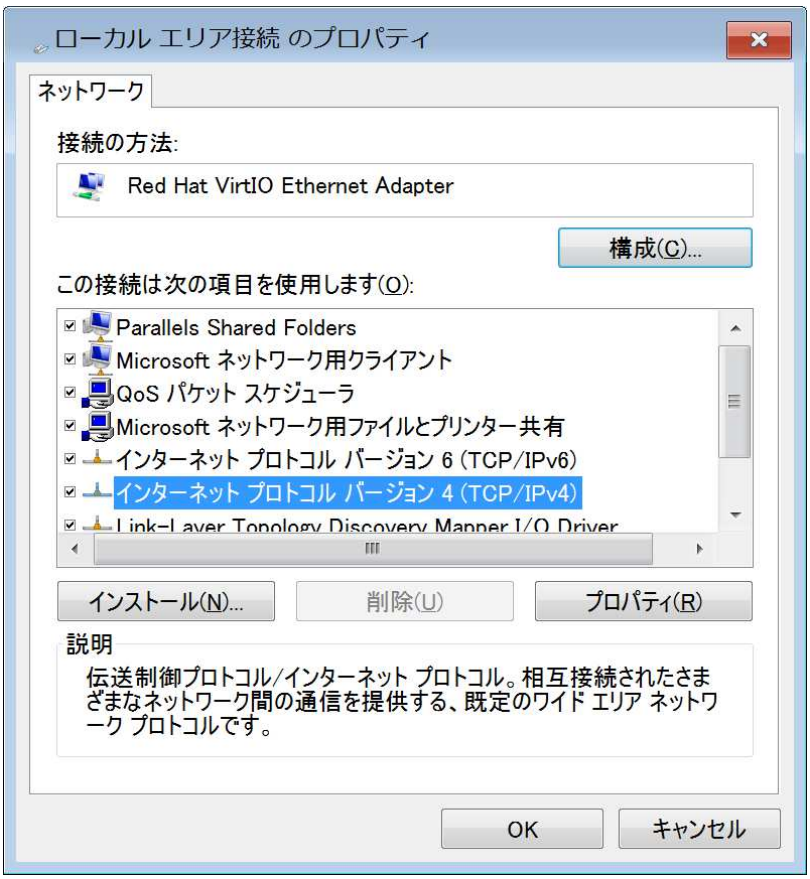


図 16　TCP/IPv4 を選択して、「プロパティ」ボタンをクリックします

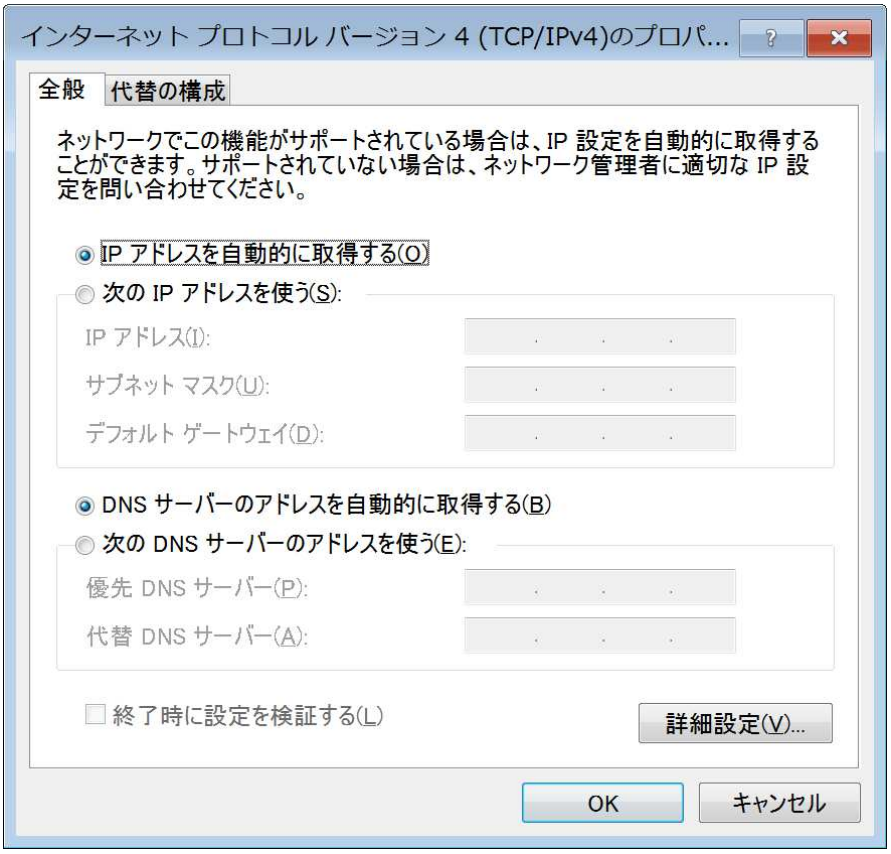


図 17　自動的に取得するように設定します

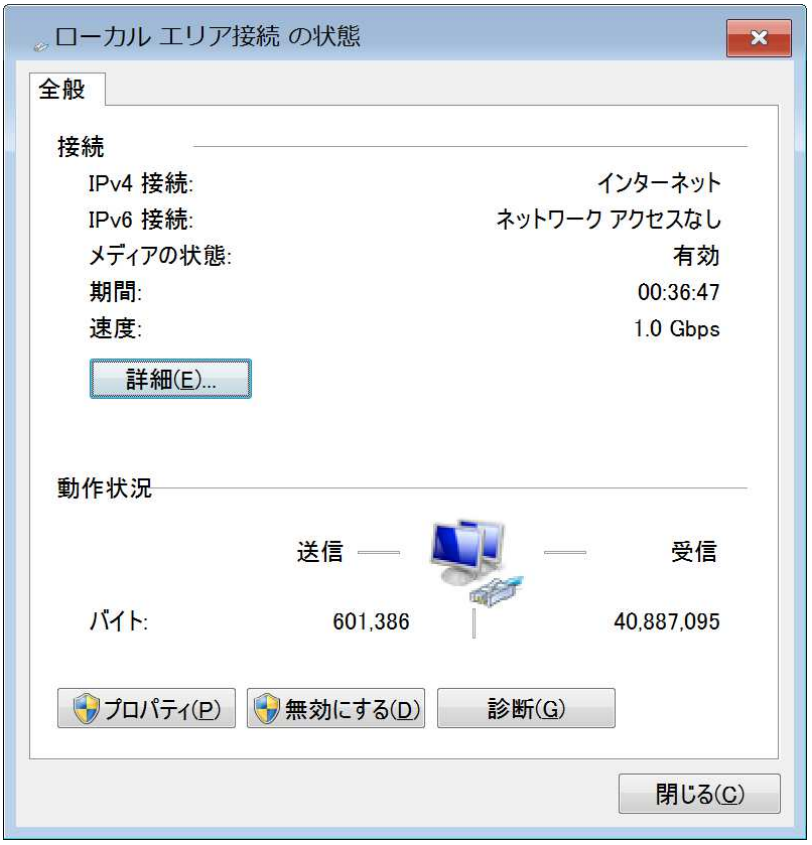


図 18　接続状態が確認できます

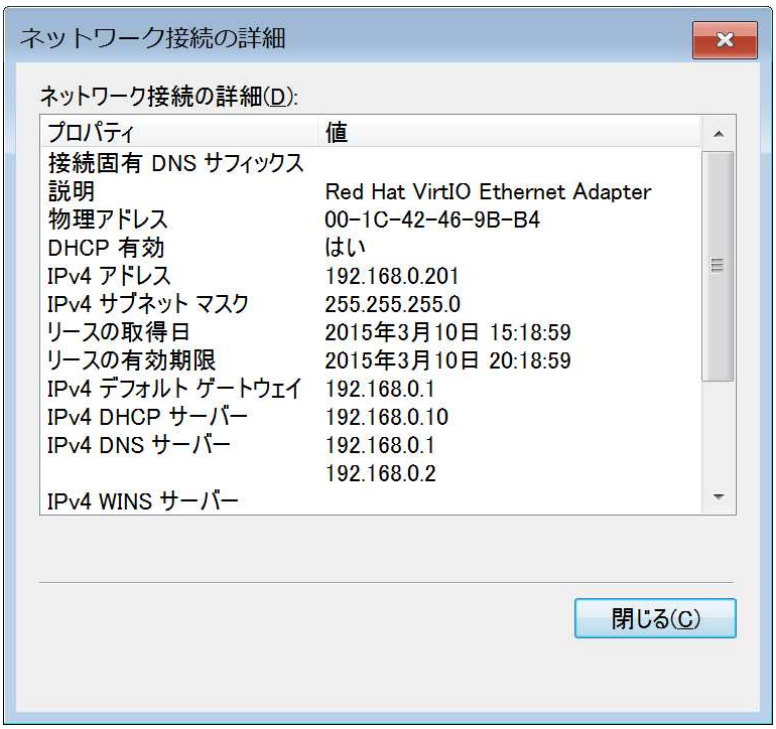


図 19　IP アドレスなどが確認できます

```
  starts 3 2015/01/14 02:22:17;
  ends 3 2015/01/14 07:22:17;
  cltt 3 2015/01/14 02:22:17;
  binding state active;
  next binding state free;
  hardware ethernet 00:1c:42:d0:ca:a0;
  client-hostname "client";
}
lease 192.168.0.201 {
  starts 3 2015/01/14 02:22:40;
  ends 3 2015/01/14 07:22:40;
  cltt 3 2015/01/14 02:22:40;
  binding state active;
  next binding state free;
  hardware ethernet 00:1c:42:46:9b:b4;
  uid "\001\000\034BF\233\264";
  client-hostname "TORUWIN7MACPRO";
}
```

2　サービスの管理

2.1　OS が起動するまでのプロセス

マシンに電源を入れた後、以下のような順番でシステムの初期化が行われ、OS が起動します。

1. 電源オン
2. BIOS 起動とハードウェアの初期化
3. ブートローダー（GRUB）の起動
4. Linux カーネルイメージの読み込み
5. init プロセスの起動

6. 各種サービスの起動
7. OS 起動

2.1.1 ブートローダー GRUB の起動

マシンの電源をオンにすると、BIOS が起動してハードウェアの初期化が行われ、起動に使用するブートデバイス（ハードディスクなど）が決定します。ブートデバイスからブートローダーである GRUB が読み込まれ、起動処理が引き継がれます。GRUB は、Linux カーネルのイメージをメモリ上にロードする役割を持っています。

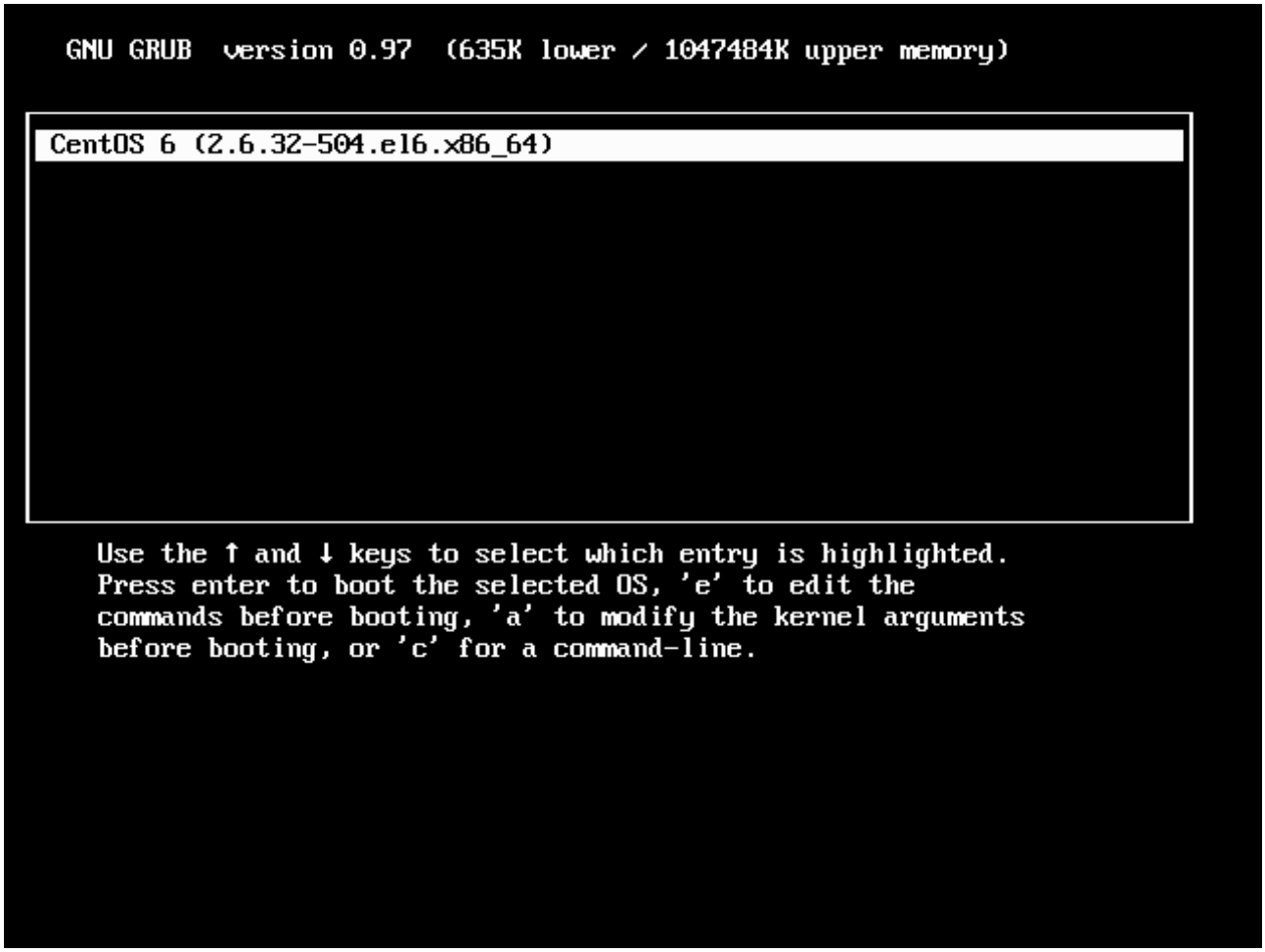


図 20 GRUB 選択画面

Linux カーネルイメージが複数ある場合は、GRUB の初期画面が表示されている時に何かキーを入力すると、GRUB のメニュー画面が表示されます。ロードしたいイメージを選択して、Enter キーを押します。

2.1.2 GRUB 設定ファイル

GRUB の設定ファイルは、/boot/grub/grub.conf です。普段は設定変更をすることはまずありませんが、システムに何かトラブルが発生した場合には設定を変更します。また、GRUB メニューから一時的にパラメーターを変更することもできます。たとえばシングルユーザモードで起動する処理はトラブルシューティングを行う際によく行う操作です。

```
# cat /boot/grub/grub.conf
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/mapper/vg_server-lv_root
```

```
#          initrd /initrd-[generic-]version.img
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title CentOS 6 (2.6.32-504.el6.x86_64)
    root (hd0,0)
    kernel /vmlinuz-2.6.32-504.el6.x86_64 ro root=/dev/mapper/vg_server-lv_root
         rd_LVM_LV=vg_server/lv_swap rd_NO_LUKS rd_LVM_LV=vg_server/lv_root rd_NO_MD
         crashkernel=auto  KEYBOARDTYPE=pc KEYTABLE=jp106 LANG=ja_JP.UTF-8 rd_NO_DM
         rhgb quiet
    initrd /initramfs-2.6.32-504.el6.x86_64.img
```

設定の意味は以下の通りです。

■default=0

timeout 行で指定したタイムアウト値が経過したときの、デフォルトのカーネルイメージを選択します。「0」の場合、最初の title 設定以降で指定されているカーネルイメージがデフォルトになります。

■timeout=5

GRUB 選択画面のタイムアウトを設定します。「5」の場合、タイムアウト値は 5 秒間になり、それを過ぎるとデフォルトのカーネルイメージが選択されます。

■splashimage=(hd0,0)/grub/splash.xpm.gz

GRUB メニューの背景画面を指定します。不要の場合はコメントアウトします。

■hiddenmenu

キーが押されるまで GRUB のメニューを表示しないようにします。

■title CentOS 6 (2.6.32-504.el6.x86_64)

GRUB のメニューに表示されるタイトルを設定します。title 設定から、次の title 設定までが 1 つの括りとして扱われます。

■root (hd0,0)

起動するパーティションを指定します。パーティションの指定方法は以下のようになります。GRUB では、デバイス番号やパーティション番号は 1 からではなく 0 からカウントされることに注意してください。

```
（デバイスのタイプ デバイス番号,パーティション番号）
```

■デバイスのタイプ

HDD や SSD の場合、接続インターフェース規格（SATA、SCSI、IDE など）に関係なく、全て「hd」で始まります。現在は殆ど使用することはありませんが、フロッピーディスクの場合は「fd」で始まります。

■デバイス番号

BIOS デバイス番号を指定します。0 から始まり、1 番目のディスクは「hd0」、2 番目のディスクは「hd1」、3 番目は「hd2」のように指定します。

■パーティション番号

パーティションを指定します。「(hd0,0)」は一番目のドライブにある、1 番目のパーティションを指定しています。

■kernel /vmlinuz-2.6.32-504.el6.x86_64

起動するカーネルを指定します。後ろには各種カーネルパラメータの指定が記述されています。

■initrd /initramfs-2.6.32-504.el6.x86_64.img

初期化 RAM ディスクを指定します。

2.1.3　カーネルバージョンの読み方

カーネルバージョンが「2.6.32-431.11.2.el6.x86_64」であった場合、バージョンは以下のように読みます。

```
メジャーバージョン.マイナーバージョン.リビジョン番号-リリース番号
```

■メジャーバージョン：※ 2 ※.6.32-504.el6.x86_64

カーネルのメジャーバージョンです。最新のメージャーバージョンは 3 です。

■マイナーバージョン：2. ※ 6 ※.32-504.el6.x86_64

バージョン 2 系では偶数が安定版、奇数の場合は開発用カーネルを意味していました。バージョン 3 系では安定版のバージョンアップのたびにマイナーバージョン番号が 1 つずつ上がるルールに変更されました。

■リビジョン番号：2.6. ※ 32 ※-504.el6.x86_64

機能追加などがあったときにリビジョン番号が上がります。

■リリース番号：2.6.32-※ 504.el6 ※.x86_64

パッケージが新しくなった時にリリース番号が上がります。リリース番号の後ろには Linux ディストリビューションの識別子が含まれます。CentOS 6 のベースになっているディストリビューションが Red Hat Enterprise Linux 6 であるため、「el6」となっています。

■アーキテクチャー: 2.6.32-504.el6. ※ x86_64 ※

CPU のアーキテクチャーです。「x86_64」は 64 ビットを表しています。

2.1.4　カーネルの起動

GRUB で指定された Linux カーネルイメージがメモリに読み込まれて、カーネルが起動します。カーネルはハードウェアを初期化し、カーネルの各種機能を有効にしていきます。

カーネルは必要に応じてモジュールを読み込みますが、モジュールは初期化 RAM ディスク（initramfs）に含まれています。カーネルは初期化 RAM ディスクをメモリに読み込み、仮のルートファイルシステムとして利用可能にすることで、必要となるモジュールのファイルが読み込めるようになります。

■dmesg によるカーネル起動時の動作の確認

カーネルが起動する際の動作の様子は、dmesg コマンドで確認できます。

```
# dmesg
Initializing cgroup subsys cpuset
Initializing cgroup subsys cpu
Linux version 2.6.32-504.el6.x86_64 (mockbuild@c6b9.bsys.dev.centos.org) (gcc version
    4.4.7 20120313 (Red Hat 4.4.7-11) (GCC) ) #1 SMP Wed Oct 15 04:27:16 UTC 2014
Command line: ro root=/dev/mapper/vg_server-lv_root rd_LVM_LV=vg_server/lv_swap
    rd_NO_LUKS rd_LVM_LV=vg_server/lv_root rd_NO_MD crashkernel=auto  KEYBOARDTYPE=pc
    KEYTABLE=jp106 LANG=ja_JP.UTF-8 rd_NO_DM rhgb quiet
KERNEL supported cpus:
  Intel GenuineIntel
  AMD AuthenticAMD
  Centaur CentaurHauls
Disabled fast string operations
BIOS-provided physical RAM map:
 BIOS-e820: 0000000000000000 - 000000000009ec00 (usable)
 (略)
```

2.1.5 init プロセスの起動とランレベル

カーネルイメージが起動された後は、init プロセスが起動されます。init プロセスには全てのプロセスを起動する役割があります。

init プロセスが起動するプロセスは、ランレベルによって区別されています。ランレベルとは、下表の通り OS の動作モードを表します。デフォルトのランレベルは/etc/inittab ファイルで設定され、システム起動時に設定されたランレベルで指定されている各種サービスが起動します。サービスは、システムを初期化するためのプロセスを実行したり、サーバのためのデーモンプロセスが起動したりします。

ランレベルの動作モードは、Linux のディストリビューションにより多少異なります。以下では CentOS 6 でのランレベルを紹介します。

ランレベル	状態
0	システムの停止
1	シングルユーザモード
2	未使用（ユーザ定義可能）
3	マルチユーザモード（コンソールログイン）
4	未使用（ユーザ定義可能）
5	マルチユーザモード（グラフィカルログイン）
6	システムの再起動

■ランレベル 0

システムを停止します。ランレベル 0 の状態にするには、以下のコマンドを実行します。

```
# init 0
# telinit 0
```

shutdown コマンドに–h オプションを付け、シャットダウンする時間を指定する方法もあります。今すぐサーバを停止する場合は以下のコマンドを実行します。

```
# shutdown -h now
```

halt コマンドでも同様に、すぐにシャットダウンを実行します。shutdown コマンドの-h オプションは halt を意味しています。

```
# halt
```

■ランレベル 1

シングルユーザモードと呼ばれ、主にメンテナンス時に使用されます。シングルユーザモードでは root ユーザのみログインが許可されます。ネットワークやデーモンは起動されず、コンソール上での操作となります。ユーザのローカルファイルシステムはマウントされます。

シングルユーザモードにするには、root ユーザで以下のコマンドを実行します。

```
# telinit 1
# init 1
```

root のパスワードを忘れてしまった場合や、システムの設定変更後に正常に起動できなくなってしまった場合などのトラブルシューティングのためにシングルユーザーモードで起動することもできます。起動方法は 6 章で解説します。

■ランレベル 2、ランレベル 4

基本的には未使用です。ユーザが独自に定義することも可能です。

■ランレベル 3

全ユーザがログイン可能なマルチユーザモードです。デーモンもネットワークも起動されている状態で、CUI による操作が可能です。

ランレベル 3 にするには、以下のコマンドを実行します。

```
# telinit 3
# init 3
```

■ランレベル 5

ランレベル 3 のマルチユーザモードに加え、グラフィカルログインが可能な状態です。ランレベル 3 からランレベル 5 にするには、以下のコマンドで X Window System を起動します。

```
# startx
```

または、以下のコマンドでも実行可能です。

```
# telinit 5
# init 5
```

■ランレベル 6

システムを再起動します。再起動後は/etc/inittab ファイルで指定されたデフォルトのランレベルで起動されます。

以下のコマンドはいずれも再起動をするためのコマンドです。

```
# telinit 6
# init 6
# reboot
# shutdown -r now
```

2.1.6 ランレベルの確認

現在のランレベルは runlevel コマンドを使って確認します。

```
# runlevel
N 5
```

左から、直前のランレベルと現在のランレベルになります。起動時のランレベルが 5 に指定されているのが分かります。起動直後など直前のランレベルがない場合は「N」と表示されます。

次にランレベルを変更します。telinit コマンドを使用します。ランレベルを「5」から「3」に変更します。GUI でログインしている場合、ログアウトして CUI のログインプロンプトが表示されます。

```
# telinit 3
```

CUI ログイン後、runlevel コマンドを実行すると、左から直前のランレベル 5、現在のランレベル 3 が表示されます。

```
# runlevel

5 3
```

2.1.7 デフォルトのランレベルを変更する

デフォルトのランレベルを変更したいとき、/etc/inittab ファイルのデフォルト設定を変更します。変更の際、/etc/inittab ファイルの編集ミスに十分注意して下さい。設定を間違えてしまうと、システムにログインできなくなる危険性があります。

```
# vi /etc/inittab
id:※3※:initdefault: ※←5を3に変更
```

上記の設定にすると、起動時のランレベルが CUI になります。設定変更後、システムを再起動します。

```
# reboot
```

デフォルトのランレベルが CUI になったことを確認できたら、元の設定であるランレベル 5 をに戻します。

```
# vi /etc/inittab
id:※5※:initdefault: ※←3を5に変更
```

```
# reboot
```

2.2 サービスの管理

Linux はバックグラウンドで動作する様々なサービスによって、サーバなどの機能を実現しています。必要なサービスを起動し、不要なサービスを停止することで、CPU やメモリなどのリソースが節約でき、セキュリティも向上します。

2.2.1 サービスの手動制御

サービスを手動で制御するときは、service コマンドを使用します。以下の例では、Web サーバである httpd サービスを制御します。

■サービスの起動

service コマンドに、起動するサービス名 httpd と start を引数として指定して実行すると、Web サーバのサービスが起動します。

```
# service httpd start
httpd を起動中:                                            [  OK  ]
```

■サービスステータスの確認

httpd サービスのステータスを確認します。

```
# service httpd status
httpd (pid  5234) を実行中...
```

■サービスの再起動

httpd サービスを再起動するときは restart オプションを使用します。たとえば、設定ファイルを書き換えた場合などには、サービスの再起動が必要となります。

```
# service httpd restart
httpd を停止中:                                            [  OK  ]
httpd を起動中:                                            [  OK  ]
```

■サービスの停止

httpd サービスを停止します。

```
# service httpd stop
httpd を停止中:                                            [  OK  ]
```

2.2.2 サービス自動起動の一覧表示

OS 起動時に自動的にサービスが起動するように設定するには、chkconfig コマンドを使用します。chkconfig コマンドに–list オプションをつけて実行すると、どのサービスが OS 起動後に自動的に起動するのか、ランレベル別にリスト表示されます。

```
# chkconfig --list
NetworkManager  0:off   1:off   2:off   3:off   4:off   5:off   6:off
abrt-ccpp       0:off   1:off   2:off   3:on    4:off   5:on    6:off
abrtd           0:off   1:off   2:off   3:on    4:off   5:on    6:off
 (略)
```

特定のサービスについて確認するときは、サービス名を指定します。

```
# chkconfig --list httpd
httpd           0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

上記の場合、httpd はどのランレベルでも自動起動せず停止している状態になっています。

2.2.3 サービスの自動起動の有効化

サービスの自動起動を有効にするには、chkconfig コマンドの引数に対象のサービスと on を指定して実行します。

```
# chkconfig httpd on
# chkconfig --list httpd
httpd           0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

chkconfig コマンドはランレベル 2、3、4、5 をまとめて on に設定します。Linux は通常ランレベル 3 かランレベル 5 で起動するので、どちらの場合でもサービスは自動起動することになります。これで Linux 起動時に httpd が自動的に起動するようになります。

2.2.4 サービスの自動起動の無効化

Linux の起動時にサービスを無効化した状態にしておきたい場合、chkconfig コマンドの引数に対象のサービスと off を指定して実行します。

```
# chkconfig httpd off
# chkconfig --list httpd
httpd           0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

これで httpd サービスは起動時に自動的に起動しないようになりました。

2.2.5 サービス起動スクリプト

service コマンドを実行すると、引数として指定された名前の起動スクリプトを呼び出して制御処理を行っています。起動スクリプトは、/etc/rc.d/init.d ディレクトリ以下に配置されています。

```
# ls /etc/rc.d/init.d/
NetworkManager    dhcpd         kdump         portreserve  single
abrt-ccpp         dhcpd6        killall       postfix      smartd
abrt-oops         dhcrelay      lvm2-lvmetad  psacct       snmpd
abrtd             dhcrelay6     lvm2-monitor  quota_nld    snmptrapd
acpid             dnsmasq       mdmonitor     rdisc        spice-vdagentd
atd               firstboot     messagebus    restorecond  sshd
auditd            functions     netconsole    rngd         sssd
autofs            haldaemon     netfs         rpcbind      sysstat
blk-availability  halt          network       rpcgssd      udev-post
bluetooth         htcacheclean  nfs           rpcidmapd    wdaemon
certmonger        httpd         nfslock       rpcsvcgssd   winbind
cpuspeed          ip6tables     ntpd          rsyslog      wpa_supplicant
crond             iptables      ntpdate       sandbox      ypbind
cups              irqbalance    oddjobd       saslauthd
```

2.2.6 ランレベルと起動スクリプトの関係

/etc/rc.d/init.d ディレクトリ以下にある起動スクリプトだけでは、指定したランレベルでのサービス起動の有無や起動、停止の順序を決めることができません。Linux では、/etc/rc.d ディレクトリ以下にランレベル毎にディレクトリを作成し、その中に起動スクリプトへのシンボリックリンクを配置しています。たとえば、ランレベル 5 の場合には rc5.d という名前でディレクトリが作成されます。

```
# ls /etc/rc.d
init.d  rc.local    rc0.d  rc2.d  rc4.d  rc6.d
rc      rc.sysinit  rc1.d  rc3.d  rc5.d
```

/etc/rc.d/rc5.d ディレクトリ以下の内容を確認します。

```
# ls /etc/rc.d/rc5.d/
K01smartd         K69rpcsvcgssd      S08iptables           S26haldaemon
K02oddjobd        K73winbind         S10network            S26udev-post
K05wdaemon        K75ntpdate         S11auditd             S28autofs
K10psacct         K75quota_nld       S11portreserve        S50bluetooth
K10saslauthd      K76ypbind          S12rsyslog            S55sshd
K15htcacheclean   K80kdump           S13cpuspeed           S58ntpd
K15httpd          K84NetworkManager  S13irqbalance         S70spice-vdagentd
K35dhcpd          K84wpa_supplicant  S13rpcbind            S80postfix
K35dhcpd6         K87restorecond     S15mdmonitor          S82abrt-ccpp
K35dhcrelay       K88sssd            S22messagebus         S82abrtd
K35dhcrelay6      K89rdisc           S24nfslock            S90crond
K50dnsmasq        K95firstboot       S24rpcgssd            S95atd
K50netconsole     K99rngd            S25blk-availability   S99certmonger
K50snmpd          S01sysstat         S25cups               S99local
K50snmptrapd      S02lvm2-monitor    S25netfs
K60nfs            S08ip6tables       S26acpid
```

たとえば、/etc/rc.d/rc5.d/S55sshd を確認してみると、/etc/rc.d/init.d/sshd スクリプトのシンボリックリンクになっています。

```
# ls -l /etc/rc.d/rc5.d/S55sshd
lrwxrwxrwx. 1 root root 14  1月  6 06:18 2015 /etc/rc.d/rc5.d/S55sshd -> ../init.d/sshd
```

シンボリックリンク名の頭文字のアルファベットで起動（Start）と停止（Kill）を判別しています。

シンボリックリンク名の数字は実行順です。サービスを起動、停止するための順番として利用されています。telinit コマンド等でランレベルが指定されると、以下のように動作します。

1. 指定されたランレベルに対応するディレクトリが決定されます。
2. 先頭が K で始まる起動スクリプトを引数に stop を指定して実行します。ただし、/var/lock/subsys ディレクトリ内に作成されているファイル名に該当するサービス名のみが対象となります。
3. 先頭が S で始まる起動スクリプトを引数に start を指定して実行します。ただし、/var/lock/subsys ディレクトリ内に作成されているファイル名に該当するサービス名は対象から除外されます。

2.2.7 シェルスクリプトとしてのサービス起動スクリプト

シェルスクリプトとは、複数のコマンドをまとめて記述したスクリプトのことです。シェルスクリプトで処理を自動化し、システム運用の効率化を図ることができます。サービス起動スクリプトも、シェルスクリプトで記述されています。サービス起動時に内部でどのような処理が行われているか、シェルスクリプトを読んでみるとよいでしょう。

Apache Web サーバのサービス起動スクリプト/etc/rc.d/init.d/httpd を読んでみる際のポイントをまとめてみました。該当する部分をスクリプト内から探してみましょう。

- シェルスクリプト/etc/rc.d/init.d/functions を読み込みます。この functions スクリプトでは、サービス起動スクリプトで使用される共通の処理が記述されています。
- スクリプトの引数として、start、stop、status、restart などが与えられます。スクリプト内では、指定された引数に応じて case 分で処理を条件分岐させています。
- 引数に restart を与えた場合には、単純にスクリプト内の stop と start のファンクションを呼び出します。
- 引数に reload を与えた場合には、killproc 関数を呼び出して httpd のプロセスに対して HUP シグナルを送っている事が分かります。プロセスは HUP シグナルを受け取ると、停止せずに設定ファイルを読み込み直します。
- 引数に configtest を与えた場合には、apachectl configtest コマンドを実行し、Apache Web サーバの設定ファイルをチェックできます。

2.2.8 init から systemd への移行

2014 年 6 月にリリースされた Red Hat Eenterprise Linux 7、そして CentOS 7 から、これまでのサービス管理である SysV init、または Upstart から Linux 向けの新しいサービス管理マネージャーである「systemd」に置き換えられました。systemd でのサービス管理については 7 章で解説します。

2.3 cron によるコマンドの自動実行

Linux では、コマンドを定期的に実行する仕組みとして cron があります。システムのバックアップ処理など、定期的に実行したい処理を自動的に行わせるのに適しています。

2.3.1 crond について

cron を使用するには、crond がバックグラウンドで実行されている必要があります。crond は毎分 cron ジョブの有無をチェックしています。もし crond が停止していると、cron ジョブは実行されません。また、システムメンテナンスなどでマシンが停止しているときも、該当時間の cron ジョブは実行されないことに注意してください。

crond が実行されているか、システム起動時に自動起動されるように設定されているかを確認します。

```
# service crond status
crond (pid  1720) を実行中...
# chkconfig --list crond
crond           0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

2.3.2 cron ジョブの実行ユーザ

cron ジョブには、システム管理者が設定を行うシステム全体の cron ジョブと、ユーザが個別に設定を行う cron ジョブの 2 種類があります。

システム全体の cron ジョブは、設定ファイル/etc/crontab に記述されています。

ユーザが個別に設定した cron ジョブは、/var/spool/cron ディレクトリ内にユーザ名で設定ファイルが作成されます。ただし、/var/spool/cron ディレクトリは root ユーザしかアクセスできないようにパーミッションが設定されているため、一般ユーザは crontab コマンドを実行して cron ジョブを設定します。

以下の例は、後述する cron ジョブ実行の設定を行った後の結果です。

```
# ls -ld /var/spool/cron/
drwx------. 2 root root 4096 11月 23 21:43 2013 /var/spool/cron/
# ls -l /var/spool/cron/
合計 8
-rw-------. 1 root     root     28  1月 14 13:38 2015 root
-rw-------. 1 testuser testuser 37  1月 14 13:40 2015 testuser
```

2.3.3 cron ジョブ実行ユーザの制御

システム管理者は、設定ファイル/etc/cron.allow および/etc/cron.deny を使用して crontab コマンドを実行できるユーザを制御することができます。

- /etc/cron.allow に登録されたユーザのみ、crontab コマンドの実行できます。
- /etc/cron.deny に登録されたユーザは、crontab コマンドの実行ができません。
- /etc/cron.allow、/etc/cron.deny のどちらのファイルも存在しない場合、すべてのユーザが crontab コマンドを実行できません。
- /etc/cron.allow が存在する場合、/etc/cron.deny は無視され、/etc/cron.allow で許可されたユーザのみが crontab コマンドを実行できます。
- /etc/cron.allow が存在するが何も記述されていない場合、すべてのユーザが crontab コマンドを実行できません。

デフォルトでは、/etc/cron.allow は無く、/etc/cron.deny は存在しますが何も記述されていないので、すべてのユーザが crontab コマンドを実行できます。

/etc/cron.allow	/etc/cron.deny	実行可能なユーザ
無し	記述無し	全てのユーザ（デフォルト）
無し	有り	/etc/cron.deny に記述されていないユーザ
無し	無し	無し
有り	無視される	/etc/cron.allow に記述されたユーザ
記述無し	無視される	無し

2.3.4　cron ジョブの書式

crond に実行させる cron ジョブは、次のような書式で記述します。

```
分　時　日　月　曜日　[ユーザ名]　コマンド
```

ユーザ名の指定は、システム全体の cron ジョブで有効です。指定されたユーザの実行権限でコマンドが実行されます。

項目	設定できる値
分	0-59
時	0-23
日	1-31
月	1-12
曜日	0-7（0,7 は日曜日）
ユーザ名	実行ユーザのユーザ名
コマンド	実行コマンド

それぞれの設定値は「,」（カンマ）で区切ることで複数指定できます。また、「/」（スラッシュ）で間隔を指定できます。

■毎時 0 分と 30 分に example.sh を実行する

```
0,30 * * * * example.sh
```

■10 分間隔で example.sh を実行する

```
*/10  * * * example.sh
```

2.3.5　crontab コマンドを使った cron ジョブの設定

cron ジョブの設定は、crontab コマンドを使って行います。

crontab コマンドオプション	説明
-e	crontab を編集
-l	登録されている crontab を表示
-r	登録されている crontab を削除
-u ユーザ名	ユーザを指定して crontab コマンドを実行（root ユーザのみ実行可能）

crontab コマンドに-e オプションを付けて実行すると、cron ジョブの編集を行うためにエディタ（通常は vi エディタ）が起動します。cron ジョブの書式に従って記述し、最後に保存を行うと反映されます。保存時に書式がチェックされ、誤りがあると再度編集を行うか確認されるので、再度編集する場合には「y」を入力します。

```
# crontab -e
```

vi エディタが起動するが起動するので、以下のように入力し、:wq で保存、終了します。

```
0 0 * * * /root/crontest.sh
```

正しい書式で cron ジョブを記述した場合、以下のメッセージが表示されます。

```
crontab: installing new crontab
```

間違った書式で cron ジョブを記述した場合、以下のメッセージが表示されます。

```
crontab: installing new crontab
"/tmp/crontab.2dEukI":1: bad day-of-week
errors in crontab file, can't install.
Do you want to retry the same edit? ※y ←yと入力すると編集に戻る※
```

ユーザ testuser でも同様に cron ジョブを設定しておきます。su コマンドでユーザ testuser に切り替え、crontab -e コマンドを実行します。

```
# su - testuser
$ crontab -e
```

記述する内容

```
0 0 * * * /home/testuser/crontest.sh
```

exit コマンドで root ユーザに戻します。

```
$ exit
logout
#
```

2.3.6 cron ジョブの一覧

登録されている cron ジョブを表示します。

```
# crontab -l
0 0 * * * /root/crontest.sh
```

root ユーザは、-u オプションでユーザを指定することでその他のユーザの cron ジョブを表示できます。

```
# crontab -u testuser -l
0 0 * * * /home/testuser/crontest.sh
```

さらに、root ユーザは-e オプションをつけて実行すると、その他のユーザの cron ジョブを編集することもできます。

```
# crontab -u testuser -e
```

2.3.7 cron ジョブの削除

crontab -r コマンドで、登録されている cron ジョブをすべて削除できます。

```
# ls /var/spool/cron/
root  testuser
# crontab -r
# ls /var/spool/cron/
testuser
```

-r オプションも、-u オプションでユーザを指定できます。

```
# crontab -u testuser -r
# ls /var/spool/cron/
※すべて削除されたので何も表示されません
```

crontab -r コマンドを間違えて実行して cron ジョブを削除してしまった場合、再度 cron ジョブを登録しなおす必要があります。

編集オプションの-e と削除オプションの-r は、キーボードで隣同士になっているので打ち間違いに注意が必要です。以下のように、crontab -l コマンドで登録されている cron ジョブを表示して、バックアップを取得しておくことも対策の一つになります。

```
# crontab -l > ~/crontab_backup
```

2.3.8 システム全体の cron ジョブについて

システム全体の cron ジョブの設定は、以下のファイルに分かれています。

ファイルおよびディレクトリ	用途	実行ユーザの指定
/var/spool/cron/root	root ユーザ用 cron ジョブ	root のみ
/etc/crontab	システムジョブ用	root 以外を指定可
/etc/cron.d ディレクトリ以下	サービス単位の cron ジョブ	root 以外を指定可
/etc/anacrontab	システムジョブ用	root のみ

/etc/crontab および/etc/cron.d ディレクトリ以下に配置する cron ジョブは、root 以外の実行ユーザを指定することができます。

2.3.9 root ユーザ固有の cron ジョブの使用

root ユーザが crontab コマンドで cron ジョブを編集すると、/var/spool/cron/root に保存されます。前述の通り、crontab コマンド実行時の操作ミスで設定した cron ジョブが全て消えてしまう危険もありますので、後述するサービス単位のジョブや、anacron を利用したシステムジョブを活用するとよいでしょう。

2.3.10 /etc/crontab による cron ジョブの設定

/etc/crontab は、最も基本的な cron ジョブを設定するためのファイルです。システムの運用上、決められた時間に実行する必要がある cron ジョブを記述します。

ただし、CentOS 6 では 1 時間おき、1 日おき、1 週間おき、1 ヶ月おきに定期的に実行する仕組みが別途用意されているので、それらの間隔で cron ジョブを実行したい場合には、後述する仕組みを利用する方が良いでしょう。

2.3.11 サービス単位の cron ジョブ

/etc/cron.d ディレクトリ内には、各種サービスのために実行される cron ジョブが記述された設定ファイルが配置されています。crond が起動する際に読み込まれて、cron ジョブが設定されます。

```
# ls /etc/cron.d
0hourly  raid-check  sysstat
```

/etc/cron.d/0hourly には、1 時間おき（毎時 1 分）に/etc/cron.hourly ディレクトリ内のシェルスクリプトが実行される cron ジョブが記述されています。1 時間おきに実行したい cron ジョブはこのファイルに記述しておくか、/etc/cron.hourly ディレクトリ内にシェルスクリプトを配置しておくとよいでしょう。

```
# cat /etc/cron.d/0hourly
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/
01 * * * * root run-parts /etc/cron.hourly
```

/etc/cron.hourly ディレクトリ内には、後述する anacron を呼び出すスクリプトが用意されています。

```
# ls /etc/cron.hourly/
0anacron
# cat /etc/cron.hourly/0anacron
#!/bin/bash
# Skip excecution unless the date has changed from the previous run
if test -r /var/spool/anacron/cron.daily; then
    day=`cat /var/spool/anacron/cron.daily`
fi
if [ `date +%Y%m%d` = "$day" ]; then
    exit 0;
fi

# Skip excecution unless AC powered
if test -x /usr/bin/on_ac_power; then
    /usr/bin/on_ac_power &> /dev/null
    if test $? -eq 1; then
    exit 0
    fi
fi
/usr/sbin/anacron -s
```

/etc/cron.d/raid-check は、ソフトウェア RAID のチェックを行う raid-check コマンドを、毎週日曜日の午前 1 時に呼び出しています。

```
# cat /etc/cron.d/raid-check
# Run system wide raid-check once a week on Sunday at 1am by default
0 1 * * Sun root /usr/sbin/raid-check
```

/etc/cron.d/sysstat は、システムの利用状況を記録する sar を呼び出しています。10 分おきに/usr/lib64/sa/sa1 が実行され、23 時 53 分に/usr/lib64/sa/sa2 が実行されます。

```
# cat /etc/cron.d/sysstat
# Run system activity accounting tool every 10 minutes
*/10 * * * * root /usr/lib64/sa/sa1 1 1
# 0 * * * * root /usr/lib64/sa/sa1 600 6 &
# Generate a daily summary of process accounting at 23:53
53 23 * * * root /usr/lib64/sa/sa2 -A
```

2.3.12　anacron によるジョブの実行

cron を使って決められた時刻に一斉に cron ジョブを実行すると、システムの負荷が集中してしまう欠点があります。特にクラウド環境において同じ時刻に cron ジョブが実行されてしまうと、CPU やメモリ、I/O などの共有リソースを複数の仮想マシン

が一斉に取り合うことになります。そこで anacron を使ってジョブを実行すると、ジョブが実行されるタイミングがランダムに決められるので、ジョブ実行が同時発生しないようになります。

anacron で実行させたいジョブはシェルスクリプトとして作成し、実行したい時間間隔に応じて以下の表のディレクトリ内に配置します。シェルスクリプトのファイルを配置するだけでジョブが定期実行されるようになるので、定期実行するジョブをパッケージのインストール時に簡単に登録できるというメリットもあります。

実行する時間間隔	ディレクトリ
1 日おき	/etc/cron.daily
1 週間おき	/etc/cron.weekly
1 ヶ月おき	/etc/cron.monthly

2.3.13 anacron の設定

anacron は、1 時間おきに crond から起動されます。起動時に設定ファイルとして/etc/anacrontab を読み込み、実行が必要なジョブを実行します。

デフォルトの設定ファイルは以下の通りです。

```
# cat /etc/anacrontab
# /etc/anacrontab: configuration file for anacron

# See anacron(8) and anacrontab(5) for details.

SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
# the maximal random delay added to the base delay of the jobs
RANDOM_DELAY=45
# the jobs will be started during the following hours only
START_HOURS_RANGE=3-22

#period in days   delay in minutes   job-identifier   command
1       5       cron.daily              nice run-parts /etc/cron.daily
7       25      cron.weekly             nice run-parts /etc/cron.weekly
@monthly 45     cron.monthly            nice run-parts /etc/cron.monthly
```

ジョブの実行頻度は、ジョブ設定の最初の数字で実行間隔を日数で記述します。デフォルトでは 1 日おきと 7 日おきのジョブが設定されています。1 ヶ月毎の設定のように、実行間隔の設定は数値以外にマクロが用意されています。

マクロ	設定値
[@daily]	1（毎日 1 回）
[@weekly]	7（毎週 1 回）
[@monthly]	毎月 1 回

各ジョブは、それぞれの基準遅延時間に最大 45 分のランダムに決められた遅延時間（RANDOM_DELAY）を足して実行されます。基準遅延時間は、ジョブ定義の 2 番目の数字です。デフォルトでは、1 日おきのジョブが 5 分、1 週間おきのジョブが 25 分、1 ヶ月おきのジョブが 45 分です。仮想マシン間で同時にジョブが実行されないようにしたい場合には、基準遅延時間を大きくずらす必要があります。

anacron がジョブを実行するのは、START_HOURS_RANGE で設定されている 3 時から 22 時の間です。anacron はシステムが停止していた場合、実行していなかったジョブを再起動後に実行する仕組みがあります。そのため、このようにジョブ実行時間が広く指定されています。ただし、この設定では日中でもジョブが実行される可能性があります。もし、夜間にだけジョブ

を実行したい場合には、/etc/anacrontab に以下のように指定するといいでしょう。ここでは夜間の 23 時から翌日の朝 6 時までを指定しています。

```
START_HOURS_RANGE=23-6
```

2.4 NTP による時刻管理

コンピューターの時刻は、意外と精度が低く 1 日ごとに数秒狂っていきます。しかも、電源 OFF にした状態だとさらに狂いやすくなります。認証やデータベース、ログを集中管理するような環境の場合には、この時刻のずれが大きな問題になる場合があります。

システムの時刻を合わせる仕組みとして、NTP（Network Time Protocol）があります。NTP を利用することで、ネットワーク上の NTP サーバから時刻を取得し、システムの時刻を正確な時刻に合わせる事ができます。

2.4.1 NTP サービスのインストール

NTP クライアントに対して時刻を提供する NTP サーバを実行するには、NTP サービスをインストールします。NTP サービスは、NTP サーバとしての機能と、自分自身の時刻を NTP サーバに同期させる NTP クライアントの機能の両方を備えています。

NTP サービスがインストールされていない場合には、yum コマンドでインストールします。

```
# yum install ntp
```

2.4.2 NTP サービスの起動と自動起動の有効化

NTP サービス（ntpd）を起動します。

```
# service ntpd start
```

chkconfig コマンドで自動起動を有効化します。

```
# chkconfig ntpd on
# chkconfig --list ntpd
ntpd            0:off   1:off   2:off   3:on    4:off   5:off   6:off
```

NTP サーバを起動してから、しばらくすると上位の NTP サーバと時刻同期が始まります。時刻同期は徐々に行われるため、すぐには完了しません。

2.4.3 上位 NTP サーバの設定

同期する時刻を提供してくれる上位 NTP サーバの設定は/etc/ntp.conf に記述します。サーバは複数指定できます。CentOS では、デフォルトで pool.ntp.org の NTP サーバに同期するように設定されています。pool.ntp.org はインターネット上の NTP サーバのアドレスをランダムに返すようになっています。

```
server 0.centos.pool.ntp.org iburst
server 1.centos.pool.ntp.org iburst
server 2.centos.pool.ntp.org iburst
server 3.centos.pool.ntp.org iburst
```

ntpq コマンドを実行して、外部の NTP サーバとの時刻同期の状態を確認します。

```
# ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*219x123x70x91.a 192.168.7.123    2 u  424 1024  377    2.296   -0.851   1.985
-balthasar.gimas 65.32.162.194    3 u  764 1024  377    4.574    3.282   1.737
```

```
+ntp-v6.chobi.pa 61.114.187.55    2 u  960 1024  337    1.012    0.546   1.170
+the.platformnin 22.42.17.250     3 u   46 1024  377    3.686    0.123   2.642
```

一番左に表示されているステータスの読み方は以下の通りです。

表示	意味
*	同期している
+	いつでも同期可能
x	クロックが不正確なため無効
空白（スペース）	使用不可（通信不可、同期に時間が掛かっている等）

2.4.4 NTP クライアントからの時刻同期リクエストの制御

NTP サービスは、デフォルトでは NTP クライアントからの時刻同期リクエストを受け付けないように設定されています。

以下の例では、NTP サーバの設定ファイル/etc/ntp.conf に「192.168.0.0/255.255.255.0」のネットワークに属している NTP クライアントからの時刻同期リクエストを許可するように設定しています。

```
# vi /etc/ntp.conf

# Hosts on local network are less restricted.
#restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap
※restrict 192.168.0.0 mask 255.255.255.0 nomodify notrap ←この行を追加
```

設定を変更したら ntp サービスを再起動します。

```
# service ntpd restart
ntpd を停止中:                                             [  OK  ]
ntpd を起動中:                                             [  OK  ]
```

2.4.5 ファイアーウォールの設定変更

NTP サーバは UDP のポート番号 123 番で NTP クライアントからの時刻同期リクエストを待ち受けています。iptables でパケットフィルタリングを行っている場合、ルールを追加する必要があります。

/etc/sysconfig/iptables を編集してルールを追加し、iptables サービスをリロードします。

```
# vi /etc/sysconfig/iptables

# Firewall configuration written by system-config-firewall
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT
※-A INPUT -m state --state NEW -m udp -p udp --dport 123 -j ACCEPT ←この行を追加※
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

service コマンドで、iptables サービスをリロードします。

```
# service iptables reload
iptables: Trying to reload firewall rules:                 [  OK  ]
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
 (略)
ACCEPT     udp  --  anywhere             anywhere            state NEW udp dpt:ntp
REJECT     all  --  anywhere             anywhere            reject-with
    icmp-host-prohibited
 (略)
```

2.4.6 NTP クライアントの NTP サービスを使って NTP サーバと時刻を同期する

クライアントの NTP サービスは、/etc/ntp.conf に server 設定で指定された NTP サーバと時刻同期を行います。

クライアントで、デフォルトで設定されている pool.ntp.org の server 設定をコメントアウトし、構築した NTP サーバ（192.168.0.10）と時刻を同期するように設定します。

クライアントに NTP サービスがインストールされていない場合には、yum コマンドでインストールします。

```
[root@client ~]# yum install ntp
[root@client ~]# vi /etc/ntp.conf

※#※server 0.centos.pool.ntp.org iburst ※←行頭でコメントアウト
※#※server 1.centos.pool.ntp.org iburst ※←行頭でコメントアウト
※#※server 2.centos.pool.ntp.org iburst ※←行頭でコメントアウト
※#※server 3.centos.pool.ntp.org iburst ※←行頭でコメントアウト
※server 192.168.0.10 iburst ←この行を追加
```

クライアントの NTP サービスを再起動します。

```
# service ntpd restart
ntpd を停止中:                                             [  OK  ]
ntpd を起動中:                                             [  OK  ]
```

ntpq コマンドで、時刻同期の状態を確認します。

```
[root@client ~]# ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*server          157.7.154.29     3 u    2   64    1    0.152    0.108   0.007
```

#ファイルシステムの管理

2.5 アクセス権の管理

Linux は POSIX で示されているアクセス制御に準拠しています。POSIX とは「Portable Operating System Interface for UNIX」の略で、IEEE によって定められた、UNIX ベースの OS の仕様セットです。ユーザ ID（uid）/グループ ID（gid）とパーミッションの組み合わせでファイルに対するアクセス権を管理しています。

2.5.1 UID と GID

ユーザ ID（uid：User Identifier) は Linux システムでユーザを識別するためのユニークな番号です。Linux で追加されたユーザカウントには、それぞれ個別に uid が割り振られます。uid は 0 から 65535 までの値をとります。0 は特別なユーザ ID で、管理

者権限を持つ root ユーザに付与されています。

グループ ID（gid: Group Identifier）はグループを識別するためのユニークな番号です。Linux のユーザは、1 つ以上のグループに所属することができます。gid は 0 から 65535 までの値をとります。

2.5.2 検証用ユーザ、グループの確認

アクセス制御の動作確認のため、検証用のユーザを用意します。すでに 1 章で作成していますが、作成されていない場合には useradd コマンド、grooupadd コマンドなどを使用して作成して下さい。

ユーザ sato とユーザ suzuki が作成されており、ユーザ suzuki は wheel グループと eigyou グループに所属しています。

```
# id sato
uid=500(sato) gid=500(sato) 所属グループ=500(sato)
# id suzuki
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki),10(wheel),5000(eigyou)
```

2.5.3 別々のユーザとして作業する

ユーザ sato とユーザ suzuki での操作をスムーズに行うため、それぞれ別々のユーザでログインします。

Linux サーバとは別の端末で操作を行っている場合には、それぞれのユーザでログインします。

Linux サーバ上の X Window System で操作を行っている場合には、root ユーザでログインした後、別々のターミナルを起動し、su コマンドを使ってユーザを切り替えるとよいでしょう。

ターミナル A でユーザ sato に切り替えます。

```
[root@server ~]# su - sato
[sato@server ~]$ id
uid=500(sato) gid=500(sato) 所属グループ=500(sato)
    context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

ターミナル B でユーザ suzuki に切り替えます。

```
[root@server ~]# su - suzuki
[suzuki@server ~]$ id
uid=501(suzuki) gid=501(suzuki) 所属グループ=501(suzuki),10(wheel),5000(eigyou)
    context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

2.5.4 プロセスの実行権の管理

Linux では、root ユーザを除いて他のユーザが起動したプロセスを停止させることはできません。

以下の例では、ユーザ sato で vi エディタ（vim）を起動して/tmp にファイルを作成しようとしているプロセスをユーザ suzuki が kill コマンドで停止しようとしますが、停止できません。

ユーザ sato が vi エディタで/tmp/sato を作成します。

```
[sato@server ~]$ vi /tmp/sato
```

ユーザ suzuki が vim プロセスを確認します。

```
[suzuki@server ~]$ ps aux | grep vim
sato      6456  0.1  0.3 148100  3692 pts/2    S+   19:46   0:00 vim /tmp/sato
suzuki    6462  0.0  0.0 107464   916 pts/3    S+   19:46   0:00 grep vim
```

ユーザ suzuki がユーザ sato が実行中の vim エディタのプロセスを kill コマンドで停止しようとしますが、停止できません。指定するプロセス ID は、ps コマンドの 2 番目の表示項目です。

```
[suzuki@server ~]$ kill 6456
-bash: kill: (6456) - 許可されていない操作です
```

ユーザ sato は「:q!」と入力して vim エディタを終了します。

2.5.5 ファイルのアクセス権の管理

ユーザ sato が作成したファイル/tmp/sato を使って、アクセス権の動作を検証します。

ユーザ sato でファイル/tmp/sato のアクセス権を確認します。その他のユーザへのアクセス権は読み取りのみ与えられています。

```
[sato@server ~]$ ls -l /tmp/sato
-rw-rw-r--. 1 sato sato 5 12月  9 17:51 2014 /tmp/sato
```

ユーザ suzuki で cat コマンドを実行し、ファイル/tmp/sato の内容を確認します。その他のユーザへの読み取りは許可されているので、内容を確認できます。

```
[suzuki@server ~]$ cat /tmp/sato
sato
```

ユーザ suzuki でファイル/tmp/sato に追記してみます。書き込みのアクセス権は与えられていないのでエラーとなります。

```
[suzuki@server ~]$ echo "suzuki" >> /tmp/sato
-bash: /tmp/sato: 許可がありません
```

2.5.6 umask とデフォルトのパーミッションの関係

umask とは、ファイルやディレクトリが新規に作成される際にデフォルトのパーミッションを決定するための値です。umask コマンドで確認できます。

```
[sato@server ~]$ umask
0002
```

umask の設定値には、新しくファイルを作成する際に設定しない（許可しない）パーミッションを 8 進数で指定します。

	読み取り	書き込み	実行
パーミッション	r	w	x
8 進数値	4	2	1

ファイルとディレクトリでは設定されるデフォルトのパーミッションが変わるので、それぞれ確認してみましょう。

2.5.7 ファイル作成のパーミッションと umask

ファイルが新規作成される際にはファイルの実行パーミッション (eXecute) は設定しないので、0666(rw-rw-rw-) に対して umask の値が適用されます。

umask が 0002 と設定されていると、その他のユーザの書き込みのパーミッション（w）が設定されていないファイル（-rw-rw-r–、0664）が作成されます。

```
[sato@server ~]$ umask
0002
[sato@server ~]$ touch testfile
[sato@server ~]$ ls -l testfile
```

```
-rw-rw-r--. 1 sato sato 0  1月 14 19:51 2015 testfile
```

2.5.8 ディレクトリ作成のパーミッションと umask

ディレクトリが新規作成される際には、実行パーミッション (eXecute) が必要になるので、0777(rwxrwxrwx) に対して umask の値が適用されます。実行パーミッションが必要になるのは、1 章でも説明したとおり、そのディレクトリをカレントディレクトリにするためには実行パーミッションが必要になるからです。

umask が 0002 と設定されていると、その他のユーザの書き込みのパーミッション（w）が設定されないディレクトリ（-rwxrwxr-x、0775）が作成されています。

```
[sato@server ~]$ umask
0002
[sato@server ~]$ mkdir testdir
[sato@server ~]$ ls -ld testdir
drwxrwxr-x. 2 sato sato 4096  1月 14 19:52 2015 testdir
```

2.5.9 umask が 4 桁の理由

パーミッションは通常、ユーザ、グループ、その他のユーザの 3 つに対するアクセス権が設定されますが、umask の値は 4 桁になっています。これは、通常のパーミッションの先頭に、setUID/setGID/スティッキービットを表す桁が含まれるためです。setUID などについては後述します。また、通常 setUID などをデフォルトパーミッションとして設定することはないので、umask は先頭を省略して 3 桁で設定することもできます。以下の例では、umask を 022 と 3 桁で設定していますが、umask コマンドの結果は 0022 になっています。

```
[sato@server ~]$ umask 022
[sato@server ~]$ umask
0022
```

2.5.10 umask を変更する

umask を変更したい場合には、umask コマンドで設定した umask 値を引数として与えます。以下の例では、umask の値を 0022 に変更したので、新規に作成したファイルのアクセス権は 644(-rw-r–r–) に設定されています。

```
[sato@server ~]$ umask 0022
[sato@server ~]$ touch umasktest
[sato@server ~]$ ls -l umasktest
-rw-r--r--. 1 sato sato 0  1月 14 19:53 2015 umasktest
```

2.5.11 root ユーザの umask とデフォルトの umask

一般ユーザの umask の値は 0002 ですが、root ユーザの umask の値は 0022 に設定されています。

```
[root@server ~]# umask
0022
```

これは、bash シェルを起動する際に読み込まれるシェルスクリプト/etc/bashrc の中でデフォルトの umask が設定されているためです。以下のように、uid が 200 以上で、かつ uid と gid が同じ場合には umask の値は 0002（002 と 3 桁表記）、それ以外は 0022 に設定されるように処理されています。

同様の処理は/etc/profile でも行われています。

```
# cat /etc/bashrc
 （略）
    # By default, we want umask to get set. This sets it for non-login shell.
```

```
    # Current threshold for system reserved uid/gids is 200
    # You could check uidgid reservation validity in
    # /usr/share/doc/setup-*/uidgid file
    if [ $UID -gt 199 ] && [ "`id -gn`" = "`id -un`" ]; then
       umask 002
    else
       umask 022
    fi
（略）
```

uid と gid が同じであることを確認しているのは、useradd コマンドでユーザアカウントを新規に作成すると、特別設定しない限り指定されたユーザ名と同じ名前のグループを作成し、uid と gid が同じになるためです。つまり、uid と gid が同じユーザは、useradd コマンドを使ってシンプルに作成されたユーザアカウントと判定できるということになります。

2.5.12 setUID の確認

setUID が実行ファイルに設定されていると、その実行ファイルは所有ユーザの権限で実行されます。setUID が設定されている場合、ls コマンドの出力で所有ユーザの実行パーミッションが「s」と表示されます。

setUID が設定されている例として、passwd コマンドがあります。一般ユーザがパスワードを変更するには、root ユーザだけが書き込める/etc/shadow ファイルに対する変更が必要です。パスワードを変更する passwd コマンドは、所有ユーザが root ユーザで setUID が設定されているので、一般ユーザが passwd コマンドを実行すると、root ユーザの権限で実行されて/etc/shadow ファイルに変更を加えることができます。

コマンドを実行したユーザを「実行ユーザ」、setUID で権限が変更されたユーザを「実効ユーザ」と呼びます。

以下の例では、passwd コマンドを一時停止して、ps コマンドで実効ユーザを確認しています。

setUID が設定されていることを確認します。

```
[sato@server ~]$ ls -l /usr/bin/passwd
-rwsr-xr-x. 1 root root 30768  2月 22 20:48 2012 /usr/bin/passwd
```

passwd を実行し、Ctrl+Z キーで一時停止します。一時停止後、シェルプロンプトに戻すためには Enter キーを押す必要があります。

```
[sato@server ~]$ passwd
ユーザー sato のパスワードを変更。
sato 用にパスワードを変更中
現在のUNIXパスワード: ※Ctrl+Zキーを入力後、Enterキーを押す
[1]+  停止                 passwd
```

ps コマンドで実効ユーザを確認します。passwd コマンドの実効ユーザが root であることが確認できます。

```
[sato@server ~]$ ps aux | grep passwd
root     15052  0.0  0.2 164012  2068 pts/1    T    10:47   0:00 passwd
sato     15178  0.0  0.0 107464   916 pts/1    S+   10:48   0:00 grep passwd
```

fg コマンドで一時停止した passwd コマンドをフォアグラウンドプロセスに戻し、Ctrl+C キーで終了します。

```
[sato@server ~]$ fg
passwd
※^C ←Ctrl+Cキーを入力
[sato@server ~]$
```

2.5.13 setGID の確認

setGID が設定されていると、所有グループの権限で実行されます。setGID は所有グループの実行パーミッションが「s」と表示されます。

setGID が設定されている例として、write コマンドや slocate コマンドがあります。

```
$ ls -l /usr/bin/write
-rwxr-sr-x  1 root tty 10124 2月 18日  2011 /usr/bin/write
$ ls -l /usr/bin/slocate
-rwxr-sr-x  1 root slocate 38516 11月 17日  2007 /usr/bin/slocate
```

write コマンドは、ログインしている他のユーザに対してメッセージを送るコマンドです。以下の例では、write コマンドを一時停止して、ps コマンドで実効グループを確認しています。

2 つのユーザアカウントでログインします。同じユーザアカウントでも構いません。write コマンドを実行し、Ctrl+Z キーで一時停止します。

```
[sato@server ~]$ write suzuki
※^Z ←Ctrl+Z キー を 入 力
[1]+  停 止                  write suzuki
```

ps コマンドで実効グループを確認します。

```
[sato@server ~]$ ps a -eo "%p %u %g %G %y %c" | grep write
23400 sato     sato     ※tty※      pts/1    write
```

表示は左から、プロセス ID（%p）、実行ユーザ（%u）、実行グループ（%g）、実効グループ（%G）、実効端末（%y）、コマンド（%c）となっています。実行したのはユーザ sato ですが、setGID されているため tty グループとして動作していることが確認できます。

tty とは「Tele-TYpewriter」の意味で、端末を表します。write コマンドはログインしている他のユーザの端末にメッセージを表示するために setGID を行って実効グループを tty グループにしているわけです。

2.5.14 スティッキービット

スティッキービットが設定されたファイルやディレクトリは、「すべてのユーザが書き込めるが、所有者しか削除できない」というアクセス権限が設定されます。

たとえば/tmp ディレクトリに対してスティッキービットが設定されています。/tmp ディレクトリは全てのユーザやアプリケーションが書き込めるディレクトリとして、一時ファイルの作成などに使用されています。しかし/tmp ディレクトリのパーミッションを 777（rwxrwxrwx）に設定すると、作成したファイルを他のユーザが削除できてしまいます。そこで/tmp ディレクトリにスティッキービットを設定すると、そのファイルを削除できるのは作成したユーザのみとなります。

スティッキービットが設定されていると、ls コマンドの出力でその他のユーザの実行パーミッションが「t」と表示されます。

```
[sato@server ~]$ ls -ld /tmp
drwxrwxrwt. 16 root root 4096  1月 14 20:26 2015 /tmp
```

ユーザ sato で/tmp/sbittest を作成し、パーミッションを 666 に設定します。

```
[sato@server ~]$ touch /tmp/sbittest
[sato@server ~]$ chmod 666 /tmp/sbittest
[sato@server ~]$ ls -l /tmp/sbittest
-rw-rw-rw-. 1 sato sato 0  1月 14 20:28 2015 /tmp/sbittest
```

ユーザ suzuki で/tmp/sbittest に書き込みをします。その他のユーザーに対する書き込みのパーミッションが付与されているので書き込みが行えます。

```
[suzuki@server ~]$ echo "suzuki" >> /tmp/sbittest
[suzuki@server ~]$ cat /tmp/sbittest
suzuki
```

ユーザ suzuki で/tmp/sbittest を削除しようとしますが、スティッキービットが働いて削除できません。

```
[suzuki@server ~]$ rm /tmp/sbittest
rm: cannot remove `/tmp/sbittest': 許可されていない操作です
```

ユーザ sato で/tmp/sbittest を削除します。所有ユーザは削除が行えます。

```
[sato@server ~]$ rm /tmp/sbittest
```

2.6 POSIX ACL

ACL(Access Control List。POSIX 準拠の ACL のため、POSIX ACL とも呼ばれる) は、Linux カーネル 2.6 から標準採用されており、従来の Linux でのアクセス権限よりも細かにアクセス権限を設定できる技術です。Linux 以外の OS、たとえば Windows などでも ACL をサポートしており、付加できる権限の種類もより細やかなものになっています。Linux でも、Windows 向けのファイルサーバとして Samba を利用する場合などには、同様のアクセス権限設定を行うために ACL が必要です。

2.6.1 ACL を有効にする条件と確認

ACL はファイルシステムの拡張属性を利用しているため、拡張属性がサポートされているファイルシステムを用いる必要があります。ext3 や ext4、XFS などほとんどのファイルシステムでは拡張属性がサポートされています。また、ファイルシステムによってはマウントする際に mount コマンドに acl オプションを指定する必要がありますが、CentOS 6 で標準で利用している ext4 ではデフォルトで ACL が有効になっているので、acl オプションの指定は必要ありません。

ACL が使用できるかは、ls コマンドでパーミッションを確認した時に、パーミッションの最後に “.” が表示されていることで判別できます。

“.” は、そのファイルに ACL が設定されていないことを意味しています。ACL が設定されると “+” に表示が変更されます。

2.6.2 ACL の設定例

実際に ACL を設定してみます。getfacl コマンドはファイルやディレクトリに対して、設定されている ACL を表示するコマンドです。また、setfacl はファイルやディレクトリに対して、ACL を設定するコマンドです。

ユーザ sato で/tmp/acltest ファイルを作成します。

```
[sato@server ~]$ touch /tmp/acltest
```

getfacl コマンドで/tmp/acltest ファイルの ACL を確認します。

```
[sato@server ~]$ getfacl /tmp/acltest
getfacl: Removing leading '/' from absolute path names
# file: tmp/acltest
# owner: sato
# group: sato
user::rw-
group::r--
other::r--
```

ユーザ suzuki で/tmp/acltest ファイルに書き込みをします。その他のユーザにはパーミッションが付与されていないので書き込みが行えません。

```
[suzuki@server ~]$ echo "suzuki" >> /tmp/acltest
-bash: /var/tmp/acltest: 許可がありません
```

ユーザ sato で setfacl コマンドを実行して、ユーザ suzuki の/tmp/acltest に対する読み書きの ACL を設定します。

```
[sato@server ~]$ setfacl -m u:suzuki:rw /tmp/acltest
[sato@server ~]$ getfacl /tmp/acltest
getfacl: Removing leading '/' from absolute path names
# file: tmp/acltest
# owner: sato
# group: sato
user::rw-
※user:suzuki:rw- ユーザsuzukiに対するACLが設定されている
group::rw-
mask::rw-
other::r--
```

ユーザ suzuki で再度/tmp/acltest ファイルに書き込みをします。ACL が設定されたので書き込みが行えます。

```
[suzuki@server ~]$ echo "suzuki" >> /tmp/acltest
[suzuki@server ~]$ cat /tmp/acltest
suzuki
```

ユーザ sato で setfacl コマンドを実行して、ユーザ suzuki の/tmp/acltest に対する読み書きの ACL を削除します。

```
[sato@server ~]$ setfacl -x u:suzuki /tmp/acltest
[sato@server ~]$ getfacl /tmp/acltest
getfacl: Removing leading '/' from absolute path names
# file: tmp/acltest
# owner: sato
# group: sato
user::rw-
group::rw-
mask::rw-
other::r--
```

ユーザ suzuki で/tmp/acltest ファイルに再度書き込みをします。ACL が削除されたので書き込みが行えません。

```
[suzuki@server ~]$ echo "suzuki" >> /tmp/acltest
-bash: /var/tmp/acltest: 許可がありません
```

2.6.3 Samba と ACL の関係

Samba で Windows クライアントに対してファイル共有を提供した際に、Windows で設定した細かな権限は Linux 上では ACL を用いて実現されています。

例として、/home/sato 以下に samba_ACL_test ディレクトリを作成し、ACL の設定を行ってみます。

■Samba のインストールと設定

Samba をインストールします。

```
# yum install samba
```

Samba の設定ファイル/etc/samba/smb.conf の workgroup 設定を変更します。ワークグループ名は Windows クライアントの参加しているワークグループに合わせます。Windows クライアントのデフォルトのワークグループは WORKGROUP です。

```
vi /etc/samba/smb.conf

        workgroup = ※WORKGROUP ←ワークグループ名を変更※
```

Samba を起動します。smb サービスと nmb サービスを起動します。

```
# service smb start
SMB サービスを起動中:                                      [  OK  ]
# service nmb start
NMB サービスを起動中:                                      [  OK  ]
```

■iptables の設定変更

iptables の設定を変更します。system-config-firewall-tui コマンドを実行してカスタマイズ設定で Samba を許可するか、/etc/sysconfig/iptables に以下の 4 行を設定して iptables サービスを reload します。Samba の使用している SMB/CIFS プロトコルは TCP と UDP の 2 種類である点に注意してください。

```
-A INPUT -m state --state NEW -m udp -p udp --dport 137 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 138 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 139 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 445 -j ACCEPT
```

■SELinux の設定変更

SELinux が有効な場合、SELinux の設定を変更します。以下の setsebool コマンドを実行して、Samba 経由でユーザのホームディレクトリへのアクセスを許可します。SELinux の設定については後述します。

```
# setsebool -P samba_enable_home_dirs on
```

■Samba ユーザの登録

smbpasswd コマンドで Samba ユーザを登録します。ユーザアカウントはあらかじめ Linux に登録されているユーザアカウントを指定する必要があります。ここではユーザ sato を指定しています。入力したパスワードは、Windows クライアントからファイル共有へアクセスする際の認証に使用します。

```
# smbpasswd -a sato
New SMB password: ※パスワードを入力
Retype new SMB password: ※パスワードを再入力
Added user sato.
```

■Windows クライアントから Samba へのアクセス

Windows クライアントから Samba のファイル共有にアクセスします。

1. Samba へのアクセスを指定します。

「スタート」ボタン→「プログラムとファイルの検索」に「\server」、あるいは「\192.168.0.10」と入力します。

＃ 2

2. ユーザ認証を行います。

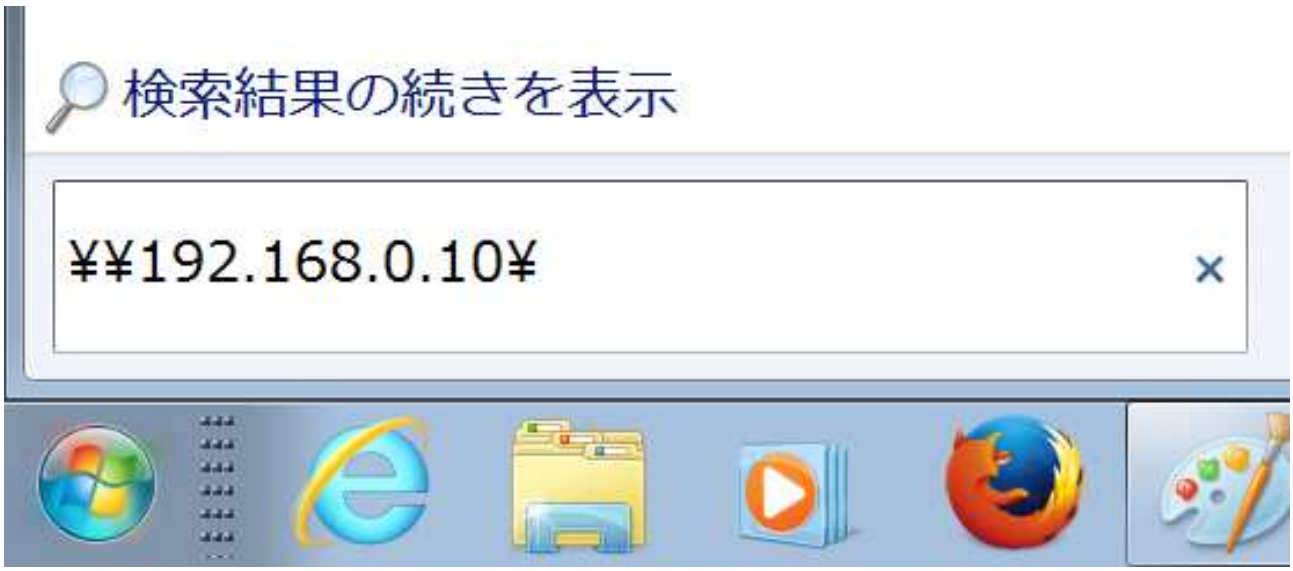


図 21　Samba へのアクセス

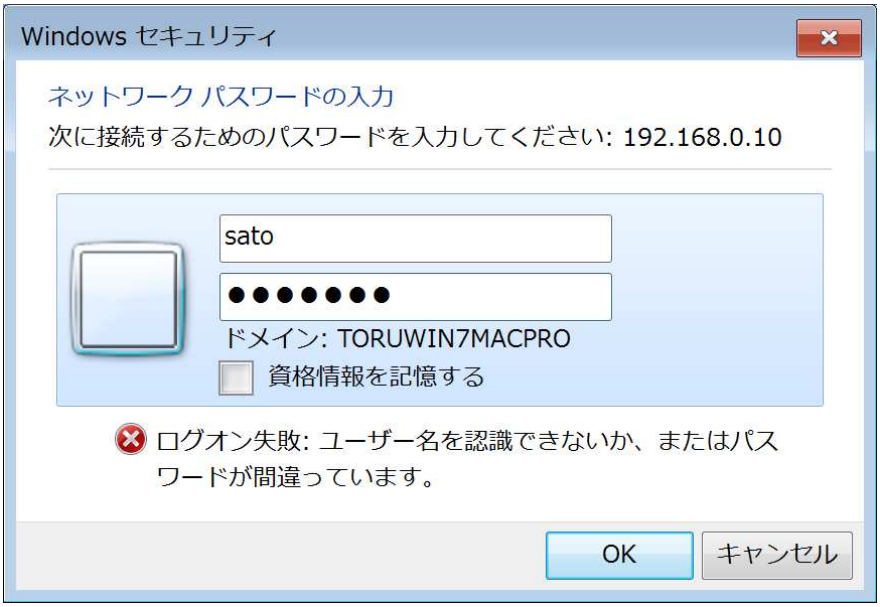


図 22　ユーザ認証

ユーザ認証が要求された場合には、前の手順で登録したユーザ名、パスワードで認証を行います。

＃ 3

3. ユーザホーム共有にアクセスします。

ユーザアカウント名のファイル共有（sato）のアイコンをダブルクリックで開きます。これは Samba がユーザのホームディレクトリを自動的に共有として扱う、ユーザホーム共有の機能を使っています。

＃ 4

4. テスト用のフォルダを作成します。

samba_acl_test フォルダを作成します。

＃ 5

5. フォルダのプロパティウインドウを呼び出します。

Windows クライアントで samba_acl_test フォルダを右クリックして、「プロパティ」を選択します。「セキュリティ」タブをク

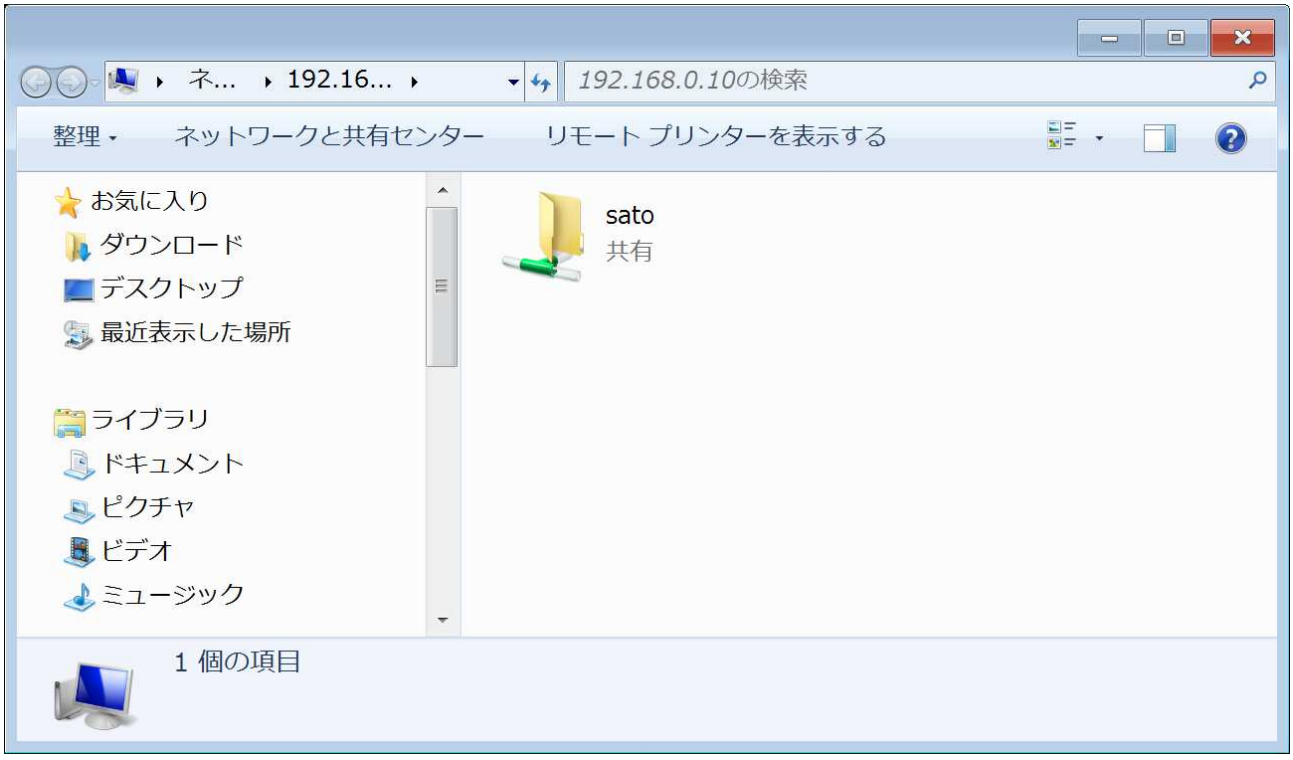


図 23　ユーザホーム共有

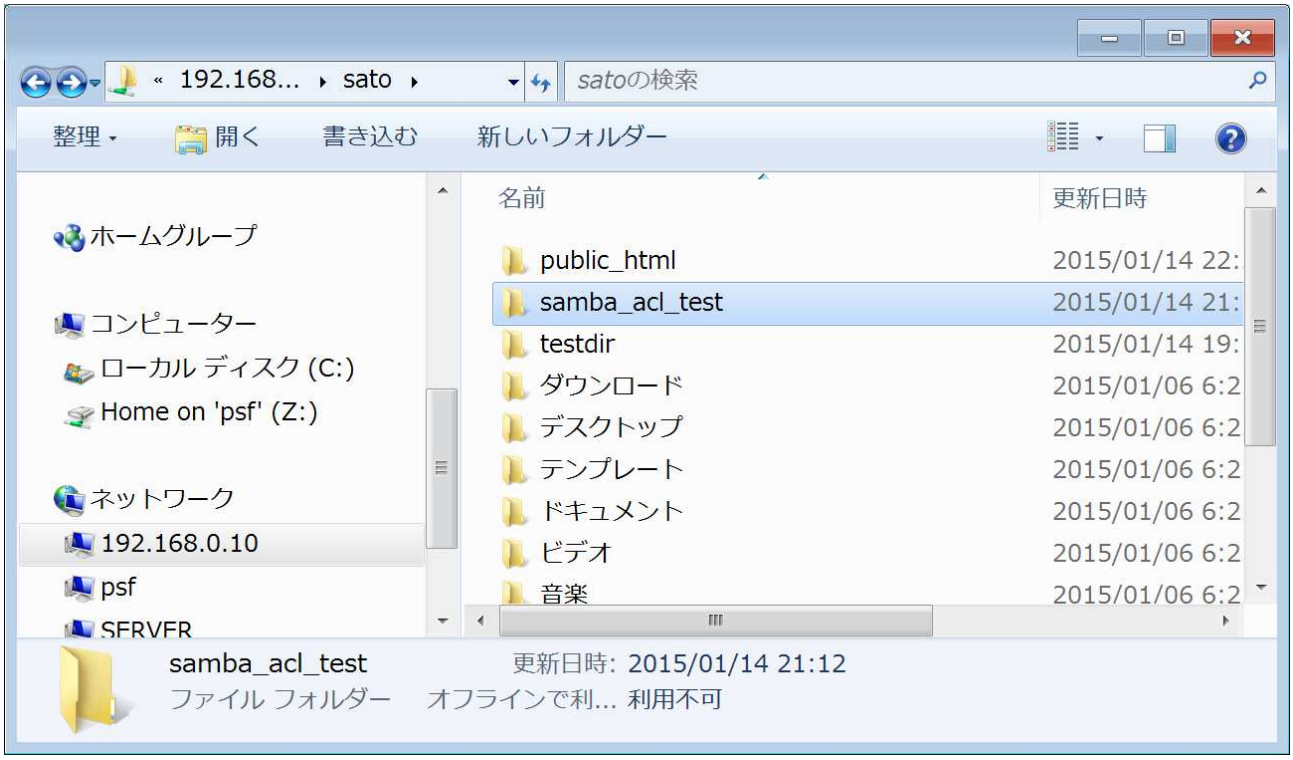


図 24　samba_acl_test フォルダ

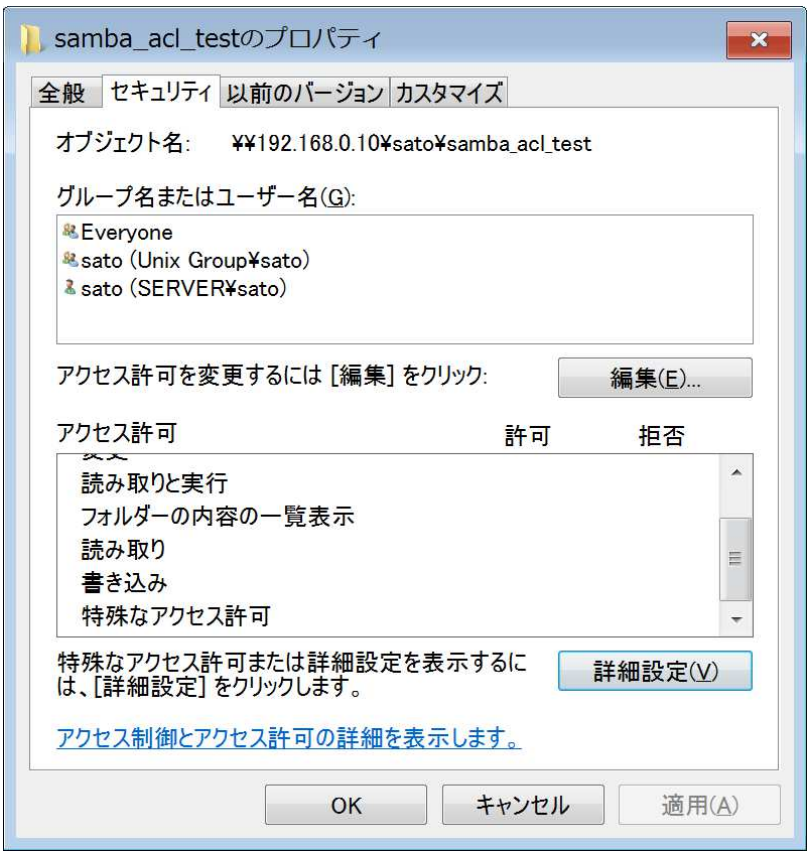


図 25　プロパティ

リックして、「詳細設定」ボタンをクリックします。

6

6. アクセス許可エントリを確認します。

「Everyone」をダブルクリックして、「許可」が 5 つチェックされていることを確認します。「OK」ボタンをクリックします。さらに OK ボタンをクリックして、プロパティのウインドウに戻ります。

■Linux から ACL を確認

1. ユーザ sato でログインし、ホームディレクトリに作られた samba_acl_test ディレクトリの ACL を確認します。

```
[sato@server ~]$ getfacl samba_acl_test/
# file: samba_acl_test/
# owner: sato
# group: sato
user::rwx
group::r-x
other::r-x
```

2

2. setfacl コマンドを実行して、samba_acl_test ディレクトリに対して、その他のユーザに書き込みの ACL を付与します。

```
[sato@server ~]$ setfacl -m o::rwx samba_acl_test
[sato@server ~]$ getfacl samba_acl_test/
# file: samba_acl_test/
# owner: sato
# group: sato
```

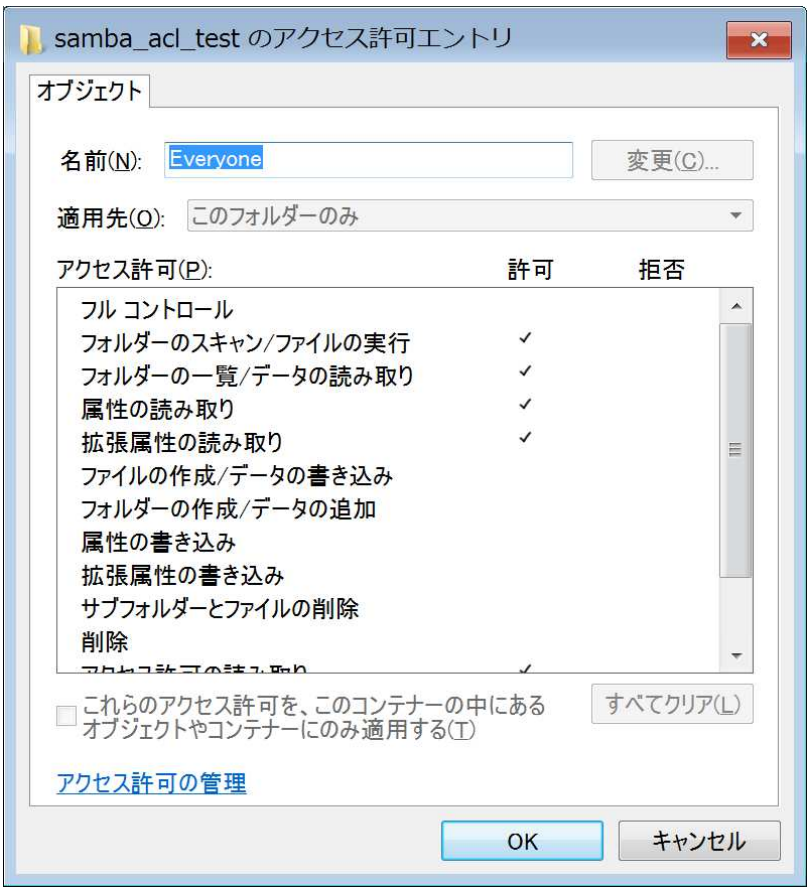


図 26　読み書きのみ

```
user::rwx
group::r-x
other::r※w※x ※←書き込み権限が付与されている
```

3

3. Windows クライアントで再度アクセス許可エントリを確認します。

Windows クライアントで再度「詳細設定」ボタンをクリックします。「Everyone」をダブルクリックして、すべてのアクセス許可項目がチェックされていることを確認します。

2.7　SELinux

SELinux は Linux カーネル 2.6 から実装された、root ユーザの特権に対しても制限を掛けることができる強制アクセス制御（MAC、Mandatory Access Control）の仕組みです。

本教科書では、SELinux の基本的な管理について解説します。SELinux のより詳しい説明については、『Linux セキュリティ標準教科書』を参照してください。

2.7.1　SELinux の仕組み

SELinux では、プロセスやファイルなど Linux の全てのリソースに対して「コンテキスト」（contexts）と呼ばれるラベルを付加し、「サブジェクト」（subject。アクセスする側。主にプロセス）が「オブジェクト」（object。アクセスされる側。主にファイルやディレクトリ、プロセス）に対してアクセスを行う際に、そのコンテキストを比較することによりアクセス制御を行います。

複数のコンテキストを組み合わせて、アクセスの可否を行うルールを SELinux では「ポリシー」と呼びます。ポリシーの詳細な説明と修正に関しては、『Linux セキュリティ標準教科書』を参照してください。

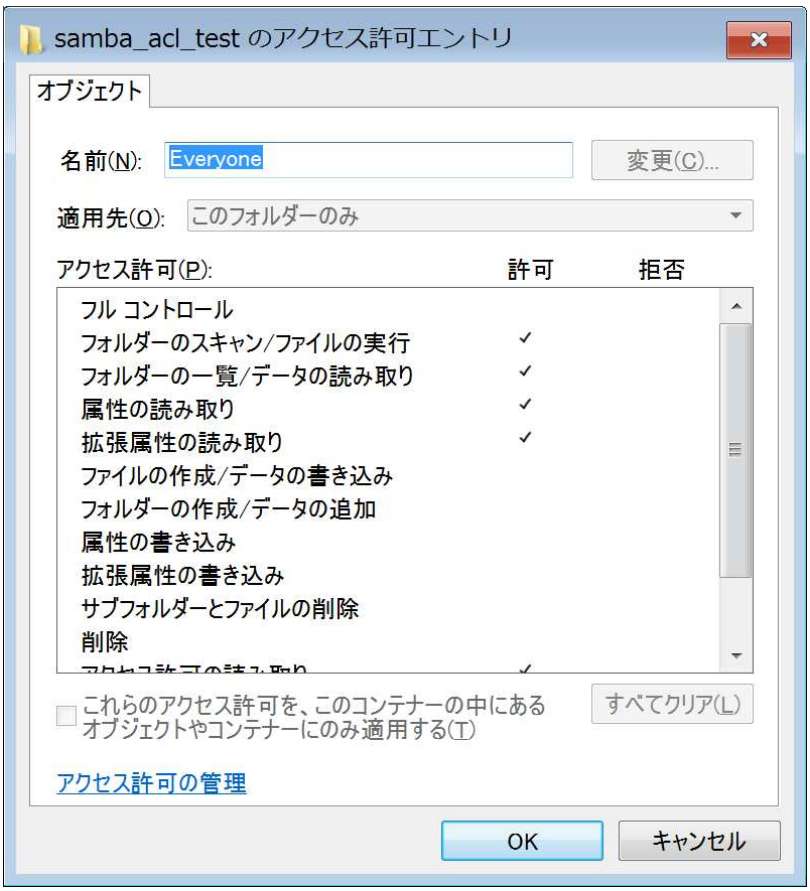


図 27　すべてのアクセス許可

2.7.2　SELinux の有効、無効の確認

SELinux の状態は getenforce コマンドで確認できます。

```
[root@server ~]# getenforce
Enforcing
```

getenforce コマンドの結果は以下の通りです。

結果	状態
Enforcing	SELinux によるアクセス制御が有効
Permissive	SELinux は有効であるが動作拒否は行わない
Disabled	SELinux によるアクセス制御が無効

SELinux の状態は、setenforce コマンドによる動的な変更か、設定ファイル/etc/selinux/config による静的な変更のいずれかで変更できます。

2.7.3　setenforce コマンドによる SELinux の動的な変更

setenforce コマンドで SELinux の状態を動的に変更できます。変更は root ユーザで実行する必要があります。

ただし、動的に変更できるのは Enforcing と Permissive の切り替えのみで、SELinux を有効から無効（Disabled）に、あるいは無効から有効に変更することはできません。有効、無効の切り替えは、後述する設定ファイルによる静的な変更とシステムの再起動が必要です。

```
setenforce [ Enforcing | Permissive | 1 | 0 ]
```

たとえば、システムの SELinux によるアクセス制御を一時的に適用しないようにしたいときには状態を Permissive に変更します。SELinux によるアクセス制御での動作の拒否は行われなくなりますが、デバッグなどの用途のために SELinux のポリシー違反が発生するとログは出力されます。システムが思ったように動作せず、SELinux が原因と思われる時などに Permissive に設定して、SELinux が原因かどうかの切り分け作業を行います。

```
# getenforce
Disabled
```

/etc/selinux/config を編集し、設定項目 SELINUX の値を enforcing に変更します。

```
# vi /etc/selinux/config

SELINUX=enforcing ※←行頭の#を削除
※#※SELINUX=disabled ※←行頭に#を追加
```

システムを再起動します。

```
# reboot
```

getenforce コマンドで SELinux が有効（Enforcing）になったことを確認します。

```
# getenforce
Enforcing
```

2.7.5 コンテキストの確認

コンテキストはファイルなどに設定され、アクセス制御に利用されます。コンテキストは、次の 4 つの識別子で構成されています。

- ユーザ (user)
- ロール (role)
- タイプ (type)：プロセスの場合には特に「ドメイン」とも言います
- MLS：高度な Multi Level Security を提供できますが、通常のシステムではあまり使われません

コンテキストは、これらの識別子を組み合わせて、以下の形式で表されます。

```
ユーザ:ロール:タイプ:MLSレベル
```

SELinux でのアクセス制御は、タイプ／ドメインに対して許可する動作を定義した「ポリシー」に基づいて行われます。タイプ／ドメインの名前は、役割やプロセス名からつけられています。たとえば、Apache Web サーバのプロセスである httpd には「httpd_t」というドメインがつけられています。

2.7.6 コンテキストの確認

SELinux のアクセス制御で用いられるコンテキストは、プロセスやファイルを参照するコマンドに-Z オプションをつけて実行することで確認できます。

たとえば、ファイルやディレクトリに付与されているコンテキストを確認するには ls -lZ コマンドを実行します。例として、Apache Web サーバ（httpd）に関するファイルを確認してみます。

```
# ls -lZ /var/www
drwxr-xr-x. root root system_u:object_r:httpd_sys_script_exec_t:s0 cgi-bin
drwxr-xr-x. root root system_u:object_r:httpd_sys_content_t:s0 error
drwxr-xr-x. root root system_u:object_r:httpd_sys_content_t:s0 html
drwxr-xr-x. root root system_u:object_r:httpd_sys_content_t:s0 icons
```

/var/www/html ディレクトリや/var/www/icons ディレクトリなど、Web サーバのコンテンツを含むディレクトリには「httpd_sys_content_t」というタイプが付与されています。この/var/www/html ディレクトリ内にファイルを作成すると、親ディレクトリのコンテキストに従ってファイルにコンテキストが付与されます。

確認のために、/var/www/html ディレクトリ以下に index.html ファイルを作成してみます。親ディレクトリからコンテキストを継承し、index.html ファイルに「httpd_sys_content_t」というタイプが付与されています。

```
# touch /var/www/html/index.html
# ls -lZ /var/www/html/index.html
-rw-r--r--. root root unconfined_u:object_r:※httpd_sys_content_t※:s0
    /var/www/html/index.html
```

また、プロセスのコンテキストの情報を確認するには、ps axZ コマンドを実行します。

以下の例では、httpd のプロセスを確認すると、httpd_t ドメインが付与されていることが分かります。

```
[root@server ~]# service httpd start
httpd を起動中:                                            [  OK  ]
[root@server ~]# ps axZ | grep httpd
unconfined_u:system_r:httpd_t:s0 27104 ?       Ss     0:00 /usr/sbin/httpd
unconfined_u:system_r:httpd_t:s0 27106 ?       S      0:00 /usr/sbin/httpd
 (略)
```

SELinux のポリシーでは、httpd プロセスに付与されている httpd_t ドメインが、「httpd_sys_content_t」などのタイプが付与されているファイルに read（読み取り）などが行えるように権限が設定されています。

2.7.7 Boolean を使った SELinux の制御

SELinux を有効にしてアプリケーションがうまく動作しない場合には、SELinux のアクセス制御によってプロセスがファイルやディレクトリにアクセスできないことが原因の場合があります。そのような時には、SELinux のポリシーを設定する必要があります。

一般的なポリシーの設定は「Boolean」（ブーリアン）と呼ばれる設定の有効、無効で対応できます。Boolean で設定できる項目は、CentOS 6 では、200 種ほどあります。

もし、独自のアプリケーションを使用したり、アプリケーションの設定を大幅に変更した場合には、ポリシーを追加、修正する必要があります。ポリシーの追加、修正方法については『Linux セキュリティ標準教科書』を参照してください。

以下の例では、Apache Web サーバ (httpd) に関するポリシーを設定しています。

getsebool コマンドで Boolean の設定状況一覧を確認します。Boolean 名には関係するプロセス名が含まれているので、grep コマンドで「httpd」をキーワードにして検索します。

```
# getsebool -a | grep httpd
allow_httpd_anon_write --> off
allow_httpd_mod_auth_ntlm_winbind --> off
 (略)
httpd_enable_homedirs --> off
 (略)
```

後の作業で httpd_enable_homedirs の Boolean を設定します。この Boolean は、Apache Web サーバのユーザホームディレクトリ機能に関する設定です。ユーザホームディレクトリ機能は、各ユーザのホームディレクトリに作成された public_html ディレクトリ内を Web コンテンツとして公開する仕組みです。

Apache Web サーバの設定ファイル/etc/httpd/conf/httpd.conf を修正し、UserDir ディレクティブを設定してユーザホームディレクトリ機能を有効にします。

```
# vi /etc/httpd/conf/httpd.conf

 (略)
<IfModule mod_userdir.c>
    #
    # UserDir is disabled by default since it can confirm the presence
    # of a username on the system (depending on home directory
```

```
    # permissions).
    #
    ※#※UserDir disabled ※←行頭に#を追加

    #
    # To enable requests to /~user/ to serve the user's public_html
    # directory, remove the "UserDir disabled" line above, and uncomment
    # the following line instead:
    #
    UserDir public_html ※←行頭の#を削除
 (略)
```

httpd サービスを再起動します。

```
# service httpd restart
httpd を停止中:                                            [  OK  ]
httpd を起動中:                                            [  OK  ]
```

ユーザ sato でログインし、ホームディレクトリに public_html ディレクトリを作成します。

```
$ pwd
/home/sato
$ mkdir public_html
```

/home/sato ディレクトリ、/home/sato/public_html ディレクトリのパーミッションを 711 に設定します。

```
$ chmod 711 /home/sato
$ chmod 711 /home/sato/public_html/
```

public_html ディレクトリに index.html ファイルを作成します。

```
[sato@server ~]$ echo "SELinux test" > /home/sato/public_html/index.html
```

ブラウザを起動し、「http://192.168.0.10/~sato/」にアクセスします。SELinux のアクセス制御が有効になっているため、「Forbidden」が表示されます。

root ユーザでログファイル/var/log/audit/audit.log を確認します。httpd(httpd_t) がユーザホームディレクトリ (user_home_dir_t) にアクセスできなかったというログが出力されています。

```
[root@server ~]# tail /var/log/audit/audit.log
 (略)
type=AVC msg=audit(1421241819.317:804): avc:  ※denied  { search }※ for  pid=7357
     comm="httpd" name="sato" dev=dm-2 ino=130305
     scontext=unconfined_u:system_r:※httpd_t※:s0
     tcontext=unconfined_u:object_r:※user_home_dir_t※:s0 tclass=dir
type=SYSCALL msg=audit(1421241819.317:804): arch=c000003e syscall=4 success=no exit=-13
     a0=7f7f0adf26e8 a1=7fff803d37c0 a2=7fff803d37c0 a3=1999999999999999 items=0
     ppid=7352 pid=7357 auid=0 uid=48 gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48
     fsgid=48 tty=(none) ses=87 comm="httpd" exe="/usr/sbin/httpd"
     subj=unconfined_u:system_r:※httpd_t※:s0 key=(null)
type=AVC msg=audit(1421241819.317:805): avc:  ※denied  { getattr }※ for  pid=7357
     comm="httpd" ※path="/home/sato"※ dev=dm-2 ino=130305
     scontext=unconfined_u:system_r:※httpd_t※:s0
     tcontext=unconfined_u:object_r:※user_home_dir_t※:s0 tclass=dir
```

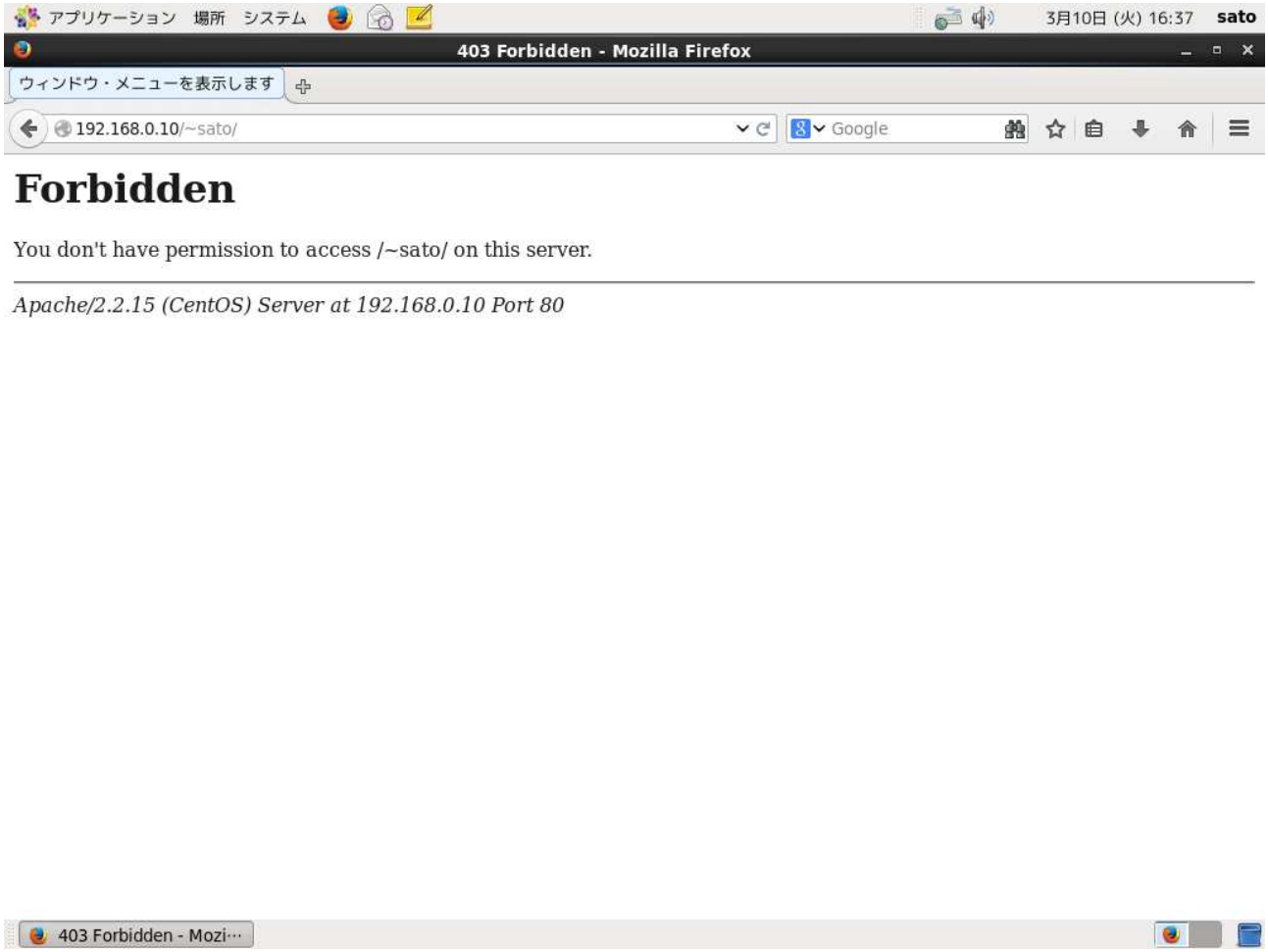


図 28　Forbidden

```
type=SYSCALL msg=audit(1421241819.317:805): arch=c000003e syscall=6 success=no exit=-13
    a0=7f7f0adf2798 a1=7fff803d37c0 a2=7fff803d37c0 a3=1 items=0 ppid=7352 pid=7357
    auid=0 uid=48 gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none)
    ses=87 comm="httpd" exe="/usr/sbin/httpd"
    subj=unconfined_u:system_r:※httpd_t※:s0 key=(null)
```

setsebool コマンドを実行して、Boolean「httpd_enable_homedirs」を有効に設定します。

```
[root@server ~]# getsebool httpd_enable_homedirs
httpd_enable_homedirs --> off
[root@server ~]# setsebool httpd_enable_homedirs on
[root@server ~]# getsebool httpd_enable_homedirs
httpd_enable_homedirs --> on
```

再度ブラウザで「http://192.168.0.10/~sato/」にアクセスします。Boolean でアクセスが許可されたので、作成したページが表示されます。

2.8　LVM の設定

LVM（Logical Volume Manager）は、ハードディスクなどの記憶媒体の物理的な状態を隠蔽し、論理的なイメージで管理するための技術です。

LVM を使うことで、複数のハードディスクにまたがったボリュームが作成できるようになり、ファイルシステムの容量が足りなくなった場合の容量の追加が簡単になります。ボリュームの操作は、システムを再起動することなく行うことができます。

また、ハードディスクに障害が発生した時には、新しい HDD を追加して、壊れている HDD を外すなどの障害対応が容易になります。他にも、スナップショットを取ることができるなどのメリットがあります。

現在の一般的な Linux ディストリビューションでは、インストール時に LVM でパーティションを作成できます。CentOS では、インストール時に自動パーティション設定を選択すると、デフォルトで LVM を使用してストレージを設定します。

LVM の詳しい説明に関しては、『高信頼システム構築標準教科書』を参照してください。

LVM は、物理ボリューム（PV: Physical Volume）、ボリュームグループ（VG: Volume Group）、論理ボリューム（LV: Logical Volume）の 3 つで構成されています。

2.8.1 物理ボリューム（PV）

物理ボリューム (PV) は、物理ディスクのパーティション単位で扱われます。一つの物理ディスクすべてを一つの PV として扱うこともできますし、一つの物理ディスク内にパーティションを複数作成し、それぞれのパーティションを別々の PV として扱うこともできます。

PV を作成するには、パーティションを作成し、パーティションタイプを 8E に設定します。

以下の例では、Linux マシンに新規に追加した/dev/sdb として認識されているハードディスクを LVM で使用できるよう、fdisk でパーティションを作成して PV として設定しています。同時に、後の作業で領域拡張を行うための追加パーティションも作成しておきます。

```
# fdisk /dev/sdb
デバイスは正常な DOS 領域テーブルも、Sun, SGI や OSF ディスクラベルも
含んでいません
（略）

コマンド（m でヘルプ）: ※n ←新規パーティション作成のnを入力
コマンドアクション
   e   拡張
   p   基本パーティション（1-4）
※p ←基本パーティションのpを入力
パーティション番号（1-4）: ※1 ←パーティション番号1を入力
最初 シリンダ（1-8354, 初期値 1）: ※1 ←パーティション番号1を入力
Last シリンダ, +シリンダ数 or +size{K,M,G}（1-8354, 初期値 8354）: ※+2G
     ←容量として+2GBを入力

コマンド（m でヘルプ）: ※n ←新規パーティション作成のnを入力
コマンドアクション
   e   拡張
   p   基本パーティション（1-4）
※p ←基本パーティションのpを入力
パーティション番号（1-4）: ※2 ←パーティション番号2を入力
最初 シリンダ（263-8354, 初期値 263）: ※Enterキーを入力
初期値 263 を使います
Last シリンダ, +シリンダ数 or +size{K,M,G}（263-8354, 初期値 8354）: ※+2G
     ←容量として+2GBを入力

コマンド（m でヘルプ）: ※t ←パーティションタイプ変更のtを入力
パーティション番号（1-4）: ※1 ←パーティション番号1を入力
16進数コード（L コマンドでコードリスト表示）: ※8e ←LVM用の8eを入力
領域のシステムタイプを 1 から 8e（Linux LVM）に変更しました

コマンド（m でヘルプ）: ※t ←パーティションタイプ変更のtを入力
パーティション番号（1-4）: ※2 ←パーティション番号2を入力
16進数コード（L コマンドでコードリスト表示）: ※8e ←LVM用の8eを入力
```

```
領域のシステムタイプを 2 から 8e (Linux LVM) に変更しました

コマンド (m でヘルプ): ※w ←パーティション情報を書き込むwを入力
パーティションテーブルは変更されました!

ioctl() を呼び出してパーティションテーブルを再読込みします。
ディスクを同期しています。
```

2.8.2 ボリュームグループ（VG）

ボリュームグループ (VG) は、1 つ以上の物理ボリューム（PV）をひとまとめにしたものです。これは仮想的なディスクに相当します。

ボリュームグループは vgcreate コマンドで作成します。

```
vgcreate ボリューム名 PVデバイス名 [PVデバイス名 ...]
```

たとえば、物理ボリューム（PV）として作成した/dev/sdb1 を使って Volume00 という名前のボリュームグループを作成するには、以下の vgcreate コマンドを実行します。

```
# vgcreate Volume00 /dev/sdb1
  Physical volume "/dev/sdb1" successfully created
  Volume group "Volume00" successfully created
```

また、ボリュームグループの情報は vgscan コマンドで確認できます。

```
# vgscan
  Reading all physical volumes.  This may take a while...
  Found volume group "Volume00" using metadata type lvm2
  Found volume group "vg_server" using metadata type lvm2
```

2.8.3 論理ボリューム（LV）

論理ボリューム（LV）は、ボリュームグループ（VG）上に作成する仮想的なパーティションです。Linux からはデバイスとして認識されます。ハードディスクに物理パーティションを作成する場合と同様に、ボリュームグループをすべて一つの論理ボリュームとすることもできますし、一つのボリュームグループを複数の論理ボリュームに分割して使用することもできます。

論理ボリュームは lvcreate コマンドを使って作成します。

```
lvcreate -L サイズ -n 論理ボリューム名 ボリュームグループ名
```

たとえば、ボリュームグループ Volume00 にサイズ 1GB、論理ボリューム名「LogVol01」の論理ボリュームを作成するには、以下の lvcreate コマンドを実行します。

```
# lvcreate -L 1024M -n LogVol01 Volume00
```

2.8.4 論理ボリュームにファイルシステムの作成

作成した論理ボリュームを利用するには、通常のパーティションと同じく論理ボリューム上にファイルシステムを作成します。論理ボリュームは、以下のようなデバイスとして扱うことができます。

```
/dev/ボリュームグループ名/論理ボリューム名
```

/dev/Volume00/LogVol01 上に ext4 ファイルシステムを作成するために、mkfs コマンドを実行します。

```
# mkfs -t ext4 /dev/Volume00/LogVol01
mke2fs 1.41.12 (17-May-2010)
Discarding device blocks: done
Filesystem label=
OS type: Linux
 (略)
This filesystem will be automatically checked every 33 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

mount コマンドを使って、/dev/Volume00/LogVol01 をマウントします。

```
# mkdir /mnt/LVMtest
# mount -t ext4 /dev/Volume00/LogVol01 /mnt/LVMtest/
# mount /mnt/LVMtest/
mount: /dev/mapper/Volume00-LogVol01 は マウント済か /mnt/LVMtest が使用中です
mount: mtab によると、/dev/mapper/Volume00-LogVol01 は /mnt/LVMtest にマウント済です
```

2.8.5 ボリュームグループへのディスクの追加

既存のボリュームグループ Volume00 に物理ボリューム/dev/sdb2 を追加します。

vgextend コマンドを実行して、物理ボリューム/dev/sdb2 をボリュームグループ Volume00 に追加します。

```
# vgextend Volume00 /dev/sdb2
  Physical volume "/dev/sdb2" successfully created
  Volume group "Volume00" successfully extended
```

vgdisplay コマンドを実行して、ボリュームグループ Volume00 の情報を確認します。PV（Physical volume）の数が 2 となっており、/dev/sdb2 が加わっていることが分かります。

```
# vgdisplay Volume00
  --- Volume group ---
  VG Name               Volume00
  System ID
  Format                lvm2
  Metadata Areas        2
  Metadata Sequence No  3
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               1
  Max PV                0
  Cur PV                ※2
  Act PV                ※2
  VG Size               4.01 GiB
  PE Size               4.00 MiB
  Total PE              1026
  Alloc PE / Size       256 / 1.00 GiB
  Free  PE / Size       770 / 3.01 GiB
  VG UUID               yTTwWd-G5tb-FzNb-Ow0L-ebvr-1n9I-ikLWo2
```

2.8.6 論理ボリュームの拡張

LVM では、論理ボリュームのサイズを変更できます。また、LVM の論理ボリューム上に作成された ext4 ファイルシステムは、ファイルシステムをマウントしたまま拡張できます。

df コマンドを実行して、現在のファイルシステムの容量を確認します。現在の容量は 1GB です。

```
# df /mnt/LVMtest/
Filesystem           1K-blocks  Used Available Use% Mounted on
/dev/mapper/Volume00-LogVol01
                        999320  1284    945608   1% /mnt/LVMtest
```

lvextend コマンドを実行して、論理ボリューム LogVol01 のサイズを 2G まで拡大します。

```
# lvextend -L 2G /dev/Volume00/LogVol01
 Size of logical volume Volume00/LogVol01 changed from 1.00 GiB (256 extents) to 2.00
     GiB (512 extents).
  Logical volume LogVol01 successfully resized
```

resize2fs コマンドを実行して、ファイルシステムを拡大します。

```
# resize2fs /dev/Volume00/LogVol01
resize2fs 1.41.12 (17-May-2010)
Filesystem at /dev/Volume00/LogVol01 is mounted on /mnt/LVMtest; on-line resizing
     required
old desc_blocks = 1, new_desc_blocks = 1
Performing an on-line resize of /dev/Volume00/LogVol01 to 524288 (4k) blocks.
The filesystem on /dev/Volume00/LogVol01 is now 524288 blocks long.
```

df コマンドで再度容量を確認します。容量が 2GB に増えていることが確認できます。

```
# df /mnt/LVMtest/
Filesystem           1K-blocks  Used Available Use% Mounted on
/dev/mapper/Volume00-LogVol01
                       2031440  1536   1925060   1% /mnt/LVMtest
```

2.8.7 論理ボリュームの縮小

運用上、他の論理ボリュームを拡大したい等の理由で使用率の低い論理ボリュームの縮小を行う場合があります。論理ボリュームを縮小するには、先にファイルシステムを縮小し、その後に論理ボリュームを縮小する必要があります。ファイルシステムの縮小はマウントしたままでは行えないので、作業中は一度アンマウントしておく必要があります。

縮小したいボリュームをアンマウントします。umount コマンドを実行して、/mnt/LVMtest をアンマウントします。

```
# umount /mnt/LVMtest/
```

縮小したい論理ボリューム/dev/Volume00/LogVol01 に対して fsck コマンドを実行します。強制的にチェックを行うために-f オプションを付与して実行します。

```
# fsck -f /dev/Volume00/LogVol01
fsck from util-linux-ng 2.17.2
e2fsck 1.41.12 (17-May-2010)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
```

```
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/mapper/Volume00-LogVol01: 11/131072 files (0.0% non-contiguous), 16812/524288
     blocks
```

resize2fs コマンドを実行して、ファイルシステムを縮小します。例として、1GB まで縮小します。

```
# resize2fs /dev/Volume00/LogVol01 1G
resize2fs 1.41.12 (17-May-2010)
Resizing the filesystem on /dev/Volume00/LogVol01 to 262144 (4k) blocks.
The filesystem on /dev/Volume00/LogVol01 is now 262144 blocks long.
```

lvreduce コマンドを実行して、論理ボリューム/dev/Volume00/LogVol01 を縮小します。

```
# lvreduce -L 1G /dev/Volume00/LogVol01
  WARNING: Reducing active logical volume to 1.00 GiB
  THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce LogVol01? [y/n]: ※y ←yを入力
  Size of logical volume Volume00/LogVol01 changed from 2.00 GiB (512 extents) to 1.00
       GiB (256 extents).
  Logical volume LogVol01 successfully resized
```

/mnt/LVMtest に再マウントして、容量を確認します。

```
# mount -t ext4 /dev/Volume00/LogVol01 /mnt/LVMtest/
# df /mnt/LVMtest/
Filesystem           1K-blocks  Used Available Use% Mounted on
/dev/mapper/Volume00-LogVol01
                        999320  1284    945616   1% /mnt/LVMtest
```

2.9 バックアップ／リストア

システムを運用していく際には、バックアップは重要です。特に障害からシステムを復旧させるときや、重要なファイルを誤って削除したりするリスクを考えると、きちんとバックアップ／リストアのプランを立てて、前もってテストをしておくことが重要です。

2.9.1 バックアップメディアについて

最近のストレージの大容量化を考えると、システムとは別のハードディスクにバックアップを取るのが一番簡単な手段です。また、容量を簡単に増やすことができるファイルサーバは、ネットワークを経由したバックアップ先の対象としてもよく利用されています。また、昔から使われているテープは現在でもバックアップメディアとしてよく利用されています。他にも、バックアップ対象の容量が多くない場合には CD や DVD もバックアップメディアの候補となります。ただし、媒体の寿命が比較的短いため、長期保存する場合には保管場所などに注意が必要です。

2.9.2 代表的なバックアップツール

企業等の運用現場ではバックアップ専用の商用ソフトウェアが多く使われていますが、Linux で標準で利用可能な様々なツールも使用されています。それらの中でも代表的な以下のツールについて解説します。

- dd コマンド
- dump コマンド
- tar コマンド
- rsync コマンド

2.9.3 dd コマンド

ディスク全体のイメージを出力するツールです。dd コマンドを用いてバックアップを行うことにより、ディスクの中身を完全にコピーすることができます。

■dd コマンドの長所

- ディスクの中身を丸ごとコピーできるため、MBR(Master Boot Record) もバックアップできます。
- i ノード番号、atime、ctime などの属性もバックアップできます。
- ディスクからディスクに直接バックアップを行う場合には、ファイルシステムを介さずに直接コピーを行うので高速です。

■dd コマンドの短所

- ディスクを丸ごとコピーするため、リストア時にファイルシステムのサイズやパラメータを変更できません。また、ファイルシステムにフラグメント（断片化）がある場合にも、そのままコピーされます。
- バックアップの際には、ファイル変更を避けるためアンマウントする必要があります。

2.9.4 dump コマンド

古くからあるバックアップ専用コマンド。ファイルシステム単位でのバックアップを行えます。

■dump コマンドの長所

- ファイルシステム単位でバックアップするため、効率よくバックアップができます。
- 差分／増分バックアップが簡単にできます。
- テープへのバックアップができます。
- i ノード番号、atime、ctime などの属性もバックアップできます。
- リストア時にファイルシステムのサイズやパラメータを変更できます。
- 新しいファイルシステムにリストアすれば、フラグメントを解消できます。

■dump コマンドの短所

- ディレクトリ単位やファイル単位でのバックアップはできません。
- バックアップファイルが壊れると、一部のファイルだけを救済できません。
- 速度はあまり速くありません。
- ext2/3/4 でしか使用できません。それ以外のファイルシステムの場合、たとえば XFS には xfsdump など専用のコマンドを使用する必要があります。

2.9.5 tar コマンド

「Tape Archiver」の名前の通り、元々はテープへのアーカイブを作成するためのコマンドですが、ファイルとしてアーカイブを作成することもでき、柔軟性があります。

■tar コマンドの長所

- ファイル単位のバックアップ、リストアができます。
- 増分バックアップができます。
- テープへのバックアップができます。
- バックアップデータがこわれていても、部分的に復旧できます。

■tar コマンドの短所

- 速度はあまり速くありません。

- リストア時に i ノード番号が変わるため、i ノードを直接参照しているプログラムではリストアしたファイルが元のファイルと同じだと認識できません。

2.9.6 rsync コマンド

「remote sync」の名前の通り、リモートでファイルやディレクトリを同期するために作られたコマンドですが、バックアップにも使用できます。バックアップ先としてローカルホストを指定することもできるため、ローカルにマウントされた外部ストレージにバックアップすることもできます。

■rsync コマンドの長所

- ファイル単位のバックアップ、リストアができます。
- tar コマンドよりも効率よくバックアップできます。
- 増分／差分バックアップができます。

■rsync コマンドの短所

- ディスクをまるごとバックアップする場合には、dd や dump と比べて遅いです。
- リストア時に i ノード番号が変わるため、i ノードを直接参照しているプログラムではリストアしたファイルが元のファイルと同じだと認識できません。

2.9.7 バックアップとリストアの準備

各コマンドを使用したバックアップとリストアの方法を、実際にコマンドを動かしながら解説します。

バックアップ対象を/mnt/backup_test（/dev/sdb1）、リストア先を/mnt/restore_test（/dev/sdc1）とします。

仮想マシンに仮想ハードディスクを 2 つ、同じサイズで追加します。追加した仮想ハードディスクを/dev/sdb と/dev/sdc として OS から認識できるように、仮想マシンを再起動します。

物理マシンを使用している場合には、物理ハードディスクを 2 台追加するか、1 台の追加したハードディスクに 2 つのパーティション（/dev/sdb1 と/dev/sdb2）を作成してください。

もし、LVM の実習で使用したハードディスク/dev/sdb をそのまま利用する場合には、アンマウントした後 fdisk コマンド等でパーティションを一度削除して実習を行います。

fdisk コマンドなどで/dev/sdb にパーティション/dev/sdb1 を作成し、mkfs.ext4 コマンドで ext4 ファイルシステムで初期化した後、/mnt/backup_test にマウントします。具体的なパーティション作成手順は、LVM の実習を参照してください。ただし、この実習では LVM は使用しないので、パーティションタイプの変更は不要です。

```
# fdisk /dev/sdb
※パーティションを作成
# mkfs.ext4 /dev/sdb1
# mkdir /mnt/backup_test
# mount -t ext4 /dev/sdb1 /mnt/backup_test/
```

/mnt/backup_test ディレクトリに、テスト用のディレクトリとファイルを作成しておきます。

```
# mkdir /mnt/backup_test/test_dir
# touch /mnt/backup_test/test_dir/test_file
```

2.9.8 dd コマンドを使ったバックアップ

dd コマンドではファイル単位でバックアップができないため、/dev/sdb デバイス自体をバックアップします。

/dev/sdc は初期化していない状態で、dd コマンドを実行します。/dev/sdb が丸ごと/dev/sdc にバックアップされます。

```
#  dd if=/dev/sdb of=/dev/sdc
208896+0 records in
208896+0 records out
106954752 bytes (107 MB) copied, 1.29132 s, 82.8 MB/s
```

fdisk コマンドを使って、/dev/sdc1 が作成されていることをパーティション情報で確認します。/dev/sdc に書き込まれたパーティション情報を OS に認識させるために OS の再起動を行ってから、確認します。

```
# reboot
※システム再起動後に確認
# fdisk /dev/sdc
（略）
コマンド (m でヘルプ): ※p ←パーティション情報表示のpを入力

ディスク /dev/sdc: 106 MB, 106954752 バイト
ヘッド 255, セクタ 63, シリンダ 13
Units = シリンダ数 of 16065 * 512 = 8225280 バイト
セクタサイズ (論理 / 物理): 512 バイト / 4096 バイト
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
ディスク識別子: 0x43b56949

デバイス ブート      始点        終点     ブロック   Id  システム
/dev/sdc1               1          13      104391   83  Linux
Partition 1 does not start on physical sector boundary.

コマンド (m でヘルプ): ※q ←終了のqを入力
```

/dev/sdc1 を/mnt/restore_test としてマウントします。先ほど/mnt/backup_test ディレクトリ以下に作成したテスト用のディレクトリおよびファイルがリストアされているのが確認できます。

```
# mount /dev/sdc1 /mnt/restore_test
# cd /mnt/restore_test
# ls -l
合計 14
drwx------. 2 root root 12288 12月 22 13:16 2014 lost+found
drwxr-xr-x. 3 root root  1024 12月 22 13:16 2014 test_dir
[root@server restore_test]# ls -l test_dir/
合計 0
-rw-r--r--. 1 root root 0  12月 22 13:16 2014 test_file
```

2.9.9 dump コマンドによるバックアップ

dump コマンドによるバックアップはファイルシステム単位で行います。バックアップ対象の選択は、設定ファイル/etc/fstab で行います。

ここでは例として/boot ディレクトリ全体をファイルとしてバックアップします。/boot ディレクトリは、/（ルート）ディレクトリとは別のパーティションにファイルシステムが作られて/boot ディレクトリにマウントされているので、/boot ディレクトリ以下をすべて dump コマンドでバックアップできます。

CentOS 6 では dump コマンドが標準でインストールされていないため、dump パッケージをインストールします。

```
# yum install dump
```

dump コマンドによるバックアップ対象を/etc/fstab に設定します。/etc/fstab の 5 番目のフィールド（後から 2 番目）が「1」に設定されているファイルシステムが dump コマンドの対象となります。/boot ディレクトリにマウントされるファイルシステムが dump コマンドの対象になっていることを確認します。/proc や/sys などの擬似ファイルシステムは対象外となります。

```
# vi /etc/fstab

/dev/mapper/vg_cent65-lv_root /                       ext4    defaults
1 1
UUID=fe4d3f56-a570-44b4-a863-418b789b42bc /boot                   ext4
defaults        ※1※ 2
/dev/mapper/vg_cent65-lv_swap swap                    swap    defaults
0 0
tmpfs                   /dev/shm                tmpfs   defaults        0 0
devpts                  /dev/pts                devpts  gid=5,mode=620  0 0
sysfs                   /sys                    sysfs   defaults        0 0
proc                    /proc                   proc    defaults        0 0
```

dump コマンドを実行して、/boot ディレクトリをバックアップします。通常はテープなどのバックアップメディアに対してバックアップを行いますが、dump コマンドの出力をパイプで dd コマンドに渡すことでファイルとしてバックアップできます。付与しているオプションの意味は以下の通りです。例では、出力先に標準出力である「-」（ハイフン）を指定している点に注意してください。

オプション	意味
-0	レベル 0 のバックアップを取得する。レベル 0 はフルバックアップ
-u	バックアップ完了後、/etc/dumpdates を更新。差分バックアップに必要
-a	自動サイズ。バックアップメディアから終了通知があるまで書き込む
-n	operator グループに対して通知を行う
-f	出力先を指定

```
# dump -0uan -f - /boot | dd of=/tmp/boot.dump
  DUMP: No group entry for operator.
  DUMP: Date of this level 0 dump: Thu Jan 15 00:07:19 2015
  DUMP: Dumping /dev/sda1 (/boot) to standard output
 （略）
  DUMP: Date this dump completed:  Thu Jan 15 00:07:20 2015
  DUMP: Average transfer rate: 26570 kB/s
  DUMP: DUMP IS DONE
53140+0 records in
53140+0 records out
27207680 bytes (27 MB) copied, 0.202273 s, 135 MB/s

# ls -l /tmp/boot.dump
-rw-r--r--. 1 root root 27207680  1月 15 00:07 2015 /tmp/boot.dump
```

restore コマンドを実行して、/tmp/restore_test ディレクトリにリストアします。-r オプションは、ファイルシステムをすべてリストアすることを指定しています。-f オプションは、入力として標準入力である「-」（ハイフン）を指定しています。dump コマンドで作成したバックアップファイルを cat コマンドで読み込み、標準出力をパイプで restore コマンドの標準入力に渡しています。

```
# mkdir /tmp/restore_test
# cd /tmp/restore_test
# cat /tmp/boot.dump | restore -rf -
# ls
```

```
System.map-2.6.32-504.el6.x86_64  initramfs-2.6.32-504.el6.x86_64.img
config-2.6.32-504.el6.x86_64      lost+found
efi                               symvers-2.6.32-504.el6.x86_64.gz
grub                              vmlinuz-2.6.32-504.el6.x86_64
```

/tmp/restore_test ディレクトリ内のファイルをすべて削除します。

```
# rm -rf /tmp/restore_test/*
```

2.9.10 tar コマンドによるバックアップ

tar コマンドはファイル、ディレクトリをアーカイブとしてひとまとめにしてバックアップが行えます。バックアップを目的とした使用のほか、たとえば Linux カーネルのソースコードなどを一つにまとめて配布する目的でも使用されています。

/boot ディレクトリ以下のファイルおよびディレクトリをバックアップします。アーカイブファイルは/tmp/boot_backup.tar とします。アーカイブの作成は、tar コマンドに-c オプションを付与して実行します。

```
# tar -cvf /tmp/boot_backup.tar /boot
tar: メンバ名から先頭の `/' を取り除きます
/boot/
/boot/grub/
 （略）
/boot/System.map-2.6.32-504.el6.x86_64
/boot/.vmlinuz-2.6.32-504.el6.x86_64.hmac

# ls -l /tmp/boot_backup.tar
-rw-r--r--. 1 root root 26982400  1月 15 00:15 2015 /tmp/boot_backup.tar
```

/tmp/restore_test ディレクトリにファイルをリストアします。アーカイブからのファイルの取り出しは、tar コマンドに-x オプションを付与して実行します。ファイルはカレントディレクトリ以下に展開されます。

```
# cd /tmp/restore_test
# tar -xvf /tmp/boot_backup.tar
boot/
boot/grub/
 （略）
boot/System.map-2.6.32-504.el6.x86_64
boot/.vmlinuz-2.6.32-504.el6.x86_64.hmac
# ls -l
合計 4
dr-xr-xr-x. 5 root root 4096  1月  6 06:20 2015 boot
# ls boot/
System.map-2.6.32-504.el6.x86_64  initramfs-2.6.32-504.el6.x86_64.img
config-2.6.32-504.el6.x86_64      lost+found
efi                               symvers-2.6.32-504.el6.x86_64.gz
grub                              vmlinuz-2.6.32-504.el6.x86_64
```

/tmp/restore_test ディレクトリ内のファイルを削除します。

```
# rm -rf /tmp/restore_test/*
```

2.9.11 rsync コマンドによるバックアップ

rsync コマンドは、ファイル、ディレクトリをバックアップすることができます。ネットワーク経由で別のホストへバックアップを行うなどの用途に使用します。特徴として、新たに追加されたファイルのみバックアップすることができます。

以下の例では、同一ホスト内で/boot ディレクトリ内のファイルを別のディレクトリにバックアップしています。

rsync コマンドを実行して、/boot ディレクトリを/tmp/restore_test ディレクトリ以下にバックアップします。

```
# rsync -av /boot /tmp/restore_test
sending incremental file list
boot/
boot/.vmlinuz-2.6.32-504.el6.x86_64.hmac
（略）
boot/grub/xfs_stage1_5
boot/lost+found/

sent 26964672 bytes  received 457 bytes  53930258.00 bytes/sec
total size is 26959690  speedup is 1.00
```

/tmp/restore_test ディレクトリ以下のファイルを確認します。

```
# ls -l /tmp/restore_test
合計 4
dr-xr-xr-x. 5 root root 4096  1月  6 06:20 2015 boot
# ls -l /tmp/restore_test/boot
合計 25848
-rw-r--r--. 1 root root  2544748 10月 15 13:54 2014 System.map-2.6.32-504.el6.x86_64
-rw-r--r--. 1 root root   106308 10月 15 13:54 2014 config-2.6.32-504.el6.x86_64
（略）
-rw-r--r--. 1 root root   200191 10月 15 13:55 2014 symvers-2.6.32-504.el6.x86_64.gz
-rwxr-xr-x. 1 root root  4152336 10月 15 13:54 2014 vmlinuz-2.6.32-504.el6.x86_64
```

/boot/rsync_test ファイルを新規に作成します。

```
# touch /boot/rsync_test
# ls -l /boot/rsync_test
-rw-r--r--. 1 root root 0  1月 15 00:23 2015 /boot/rsync_test
```

再度 rsync コマンドを実行します。新たに追加されたファイルのみバックアップされます。

```
# rsync -av /boot /tmp/restore_test
sending incremental file list
boot/
boot/rsync_test

sent 832 bytes  received 40 bytes  1744.00 bytes/sec
total size is 26959690  speedup is 30917.08
```

新たにバックアップされたファイルを確認します。

```
# ls -l /tmp/restore_test/boot/rsync_test
-rw-r--r--. 1 root root 0  1月 15 00:23 2015 /tmp/restore_test/boot/rsync_test
```

tmp/restore_test ディレクトリ内のファイルを削除します。

```
# rm -rf /tmp/restore_test/*
```

3 システムのメンテナンス

3.1 パッケージ管理

Linux の各ディストリビューションでは、Linux カーネルやアプリケーションなどシステムに必要なソフトウェアをパッケージ形式でインストール、管理する仕組みを持っています。パッケージは、たとえば Linux カーネルであればカーネル本体およびカーネルモジュールを 1 つのパッケージファイルにまとめたものです。

Red Hat Enterprise Linux や CentOS、SUSE Linux などでは、パッケージ管理に RPM(Red Hat Package Manager) が採用されています。また、アップデート時のバージョン管理等を行う為に開発された Yum（Yellowdog Updater Modified）が、システムの更新等で広く使われています。

Debian GNU/Linux や Ubuntu などのディストリビューションでは、パッケージ管理に Debian パッケージ（deb 形式）が採用されています。パッケージ管理ツールとして APT（Advanced Package Tool）が使われています。

本教科書では、CentOS 6 のパッケージ管理ツールである yum の使用方法について解説します。

3.1.1 Yum とは

以前は RPM パッケージの管理には rpm コマンドが使用されていましたが、パッケージ間の依存関係を自動的に解決できなかったため、パッケージのインストールを行う際に管理者が自分で依存関係を確認しながら、手動で必要なパッケージを指定する必要がありました。そのため、依存関係で必要となるパッケージが多数あった場合、インストール作業が大変でした。

Yum では、yum コマンドを使ったパッケージのインストール時に依存関係の解決を自動的に行い、必要となるパッケージも同時にインストールするため、パッケージ管理が簡単になっています。

3.1.2 Yum の設定

Yum では、RPM パッケージをまとめて置いておく場所を「リポジトリ」と呼びます。パッケージのインストールの際には、リポジトリから必要な RPM パッケージを取得します。

リポジトリの設定ファイルは/etc/yum.repos.d ディレクトリにあります。

```
# ls /etc/yum.repos.d
CentOS-Base.repo       CentOS-Media.repo  CentOS-fasttrack.repo
CentOS-Debuginfo.repo  CentOS-Vault.repo
```

それぞれのリポジトリ設定ファイルに、リポジトリの参照先などが記述されています。

デフォルトで利用される CentOS-Base.repo の中身は以下の通りです。

```
# cat /etc/yum.repos.d/CentOS-Base.repo
 （略）
[base]
name=CentOS-$releasever - Base
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=os&infra=$infra
#baseurl=http://mirror.centos.org/centos/$releasever/os/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
 （略）

#additional packages that extend functionality of existing packages
[centosplus]
name=CentOS-$releasever - Plus
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=centosplus&infra=$
#baseurl=http://mirror.centos.org/centos/$releasever/centosplus/$basearch/
gpgcheck=1
```

```
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
（略）
```

設定項目 mirrorlist で指定した mirror.centos.org は、リポジトリのミラーリストから返される、ネットワークで接続しやすいリポジトリのアドレスに置き換えられます。

設定項目 enabled の値が 0 に設定されていると、そのリポジトリは yum コマンドの–enablerepo オプションで指定されなければ参照しません。

yum コマンドは、インターネットに接続し、HTTP でインストールするパッケージをダウンロードすることができる必要があります。もし、PROXY サーバを経由する必要がある場合には、yum コマンドの設定ファイル/etc/yum.conf に PROXY サーバに関する設定を記述する必要があります。設定項目は以下の通りです。

項目	設定する値
proxy	PROXY サーバの URL
proxy_username	PROXY サーバの認証ユーザ名
proxy_password	PROXY サーバの認証パスワード

また、インターネットに接続できない場合には、インストール用の DVD メディアをリポジトリとして参照する方法が利用できます。具体的な手順は後述します。

3.1.3 yum コマンドの基本的な使い方

yum コマンドは、引数として様々なサブコマンドを指定して使用します。主なサブコマンドは以下の通りです。

■パッケージのインストール

指定されたパッケージをインストールします。

```
yum install パッケージ名
```

■パッケージのアンインストール（削除）

指定されたパッケージをアンインストール（削除）します。

```
yum remove パッケージ名
```

■パッケージの更新の確認

インストールされているパッケージの更新があるかを確認します。

```
yum check-update
```

■パッケージの更新

インストールされているパッケージを更新します。パッケージ名を指定しなかった場合には、すべての更新可能なパッケージが対象となります。

```
yum update [パッケージ名]
```

■パッケージグループの一覧表示

利用可能なパッケージグループを表示します。

```
yum grouplist
```

■パッケージグループのインストール

指定されたパッケージグループに含まれるすべてのパッケージをまとめてインストールします。

```
yum groupinstall パッケージグループ名
```

■パッケージグループのアンインストール（削除）

指定されたパッケージグループに含まれるすべてのパッケージをまとめてアンインストール（削除）します。

```
yum groupremove パッケージグループ名
```

3.1.4 パッケージグループを指定したインストール

yum コマンドを使ったパッケージのインストールは dump コマンドのインストールなどで既に行っているので、パッケージグループを指定したインストールを行います。

以下の例では、高機能なエディタである「Emacs」パッケージグループをインストールします。

利用可能なパッケージグループを表示します。

```
# yum grouplist
読み込んだプラグイン:fastestmirror, refresh-packagekit, security
グループ処理の設定をしています
Loading mirror speeds from cached hostfile
 * base: ftp.nara.wide.ad.jp
 * extras: ftp.nara.wide.ad.jp
 * updates: ftp.nara.wide.ad.jp
インストール済みグループ:
   CIFS ファイルサーバー
   Java プラットフォーム
（略）
利用可能なグループ
   Eclipse
   ※Emacs
（略）
```

「Emacs」パッケージグループをインストールします。

```
# yum groupinstall Emacs
読み込んだプラグイン:fastestmirror, refresh-packagekit, security
グループ処理の設定をしています
Loading mirror speeds from cached hostfile
 * base: ftp.riken.jp
 * extras: ftp.riken.jp
 * updates: ftp.riken.jp
依存性の解決をしています
```

```
--> トランザクションの確認を実行しています。
---> Package emacs.x86_64 1:23.1-25.el6 will be インストール
--> 依存性の処理をしています: emacs-common = 1:23.1-25.el6 のパッケージ:
    1:emacs-23.1-25.el6.x86_64
（略）

依存性を解決しました

================================================================================
 パッケージ              アーキテクチャ
                                    バージョン            リポジトリー
                                                                    容量
================================================================================
インストールしています:
 emacs                  x86_64      1:23.1-25.el6         base        2.2 M
依存性関連でのインストールをします。:
 emacs-common           x86_64      1:23.1-25.el6         base         18 M
 libXaw                 x86_64      1.0.11-2.el6          base        178 k
 libXpm                 x86_64      3.5.10-2.el6          base         51 k
 libotf                 x86_64      0.9.9-3.1.el6         base         80 k
 m17n-db-datafiles      noarch      1.5.5-1.1.el6         base        717 k

トランザクションの要約
================================================================================
インストール         6 パッケージ

総ダウンロード容量: 21 M
インストール済み容量: 73 M
これでいいですか? [y/N]※y ←yを入力
パッケージをダウンロードしています:
(1/6): emacs-23.1-25.el6.x86_64.rpm                       | 2.2 MB     00:00
（略）

インストール:
  emacs.x86_64 1:23.1-25.el6

依存性関連をインストールしました:
  emacs-common.x86_64 1:23.1-25.el6           libXaw.x86_64 0:1.0.11-2.el6
  libXpm.x86_64 0:3.5.10-2.el6                libotf.x86_64 0:0.9.9-3.1.el6
  m17n-db-datafiles.noarch 0:1.5.5-1.1.el6

完了しました!
```

Emacs を起動します

```
# emacs
```

Emacs を終了します。Ctrl+X キーを押した後、Ctrl+C キーを押します。

3.1.5 パッケージグループ名を英語表記で表示する

yum コマンドはロケール（Locale）に対応しているため、環境変数 LANG の値が日本語に設定されているとパッケージグループ名が日本語で表示されます。このため、yum groupinstall コマンドを実行する際に日本語でパッケージグループ名を指定しなければなりません。

日本語がうまく表示できない、あるいは日本語が入力できない環境の場合、yum コマンドの前に「LANG=C」と入力することでパッケージグループ名を英語表記で表示できます。この方法は環境変数 LANG の値を一時的に変更した状態で、yum コマンドを実行したことになります。

```
# LANG=C yum grouplist
 (略)
Installed Groups:
   Additional Development
   Base
   CIFS file server
 (略)
```

インストール時には、ローケルを指定しないでも英語表記のままパッケージグループ名を指定できます。パッケージグループ名に空白が含まれている場合には「"」(ダブルクォート)でパッケージグループ名を括って下さい。

以下の例では、開発ツール(Development tools)パッケージグループを指定して、コンパイラなどをインストールしています。

```
# yum groupinstall "Development tools"
```

3.1.6 インストール DVD メディアをリポジトリにする方法

インターネットに接続できない環境で yum コマンドを利用する方法として、インストール DVD メディアをリポジトリとして参照させる方法があります。

設定ファイル/etc/yum.repos.d/CentOS-Media.repo が用意されており、以下のように設定が記述されています。

```
# cat /etc/yum.repos.d/CentOS-Media.repo
# CentOS-Media.repo
#
#  This repo can be used with mounted DVD media, verify the mount point for
#  CentOS-6.  You can use this repo and yum to install items directly off the
#  DVD ISO that we release.
#
# To use this repo, put in your DVD and use it with the other repos too:
#  yum --enablerepo=c6-media [command]
#
# or for ONLY the media repo, do this:
#
#  yum --disablerepo=\* --enablerepo=c6-media [command]

[c6-media]
name=CentOS-$releasever - Media
baseurl=file:///media/CentOS/
        file:///media/cdrom/
        file:///media/cdrecorder/
gpgcheck=1
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
```

この設定を利用するには、インストール DVD メディアを/media/CentOS ディレクトリにマウントし、yum コマンドにリポジトリ指定のオプションをつけて実行する必要があります。

以下の手順で、インストール DVD メディアをリポジトリとして参照できるようにします。

1. CentOS にユーザ root としてグラフィカルログインします。

2. インストール DVD メディアを DVD ドライブに挿入します。仮想マシンの場合には、インストール ISO イメージファイルを仮想 DVD ドライブで参照します。
3. 自動マウントされることを確認します。
4. mount コマンドで確認します。インストール DVD メディアは/media/CentOS_6.6_Final にマウントされています。

```
# mount
 (略)
/dev/sr0 on /media/CentOS_6.6_Final type iso9660
    (ro,nosuid,nodev,uhelper=udisks,uid=0,gid=0,iocharset=utf8,mode=0400,dmode=0500)
```

5

1. シンボリックリンク/media/CentOS を作成します。

```
# ln -s /media/CentOS_6.6_Final/ /media/CentOS
# ls -l /media
合計 4
lrwxrwxrwx. 1 root root   24  1月 15 02:47 2015 CentOS -> /media/CentOS_6.6_Final/
dr-xr-xr-x. 7 root root 4096 10月 24 23:17 2014 CentOS_6.6_Final
```

6

1. yum コマンドを実行します。–disablerepo オプションですべてのリポジトリを参照不要とし、–enablerepo オプションで c6-media リポジトリのみ参照するように指定します。以下の例では、グループリストを取得しています。

```
# yum --disablerepo=\* --enablerepo=c6-media grouplist
```

3.2 システム監視

システムを適切に運用していく上で、システム上のリソースが常に有効に使われているか、極端にリソースを占有しているプロセスは無いかを監視することは重要な事です。

システム上のリソースを様々な角度で監視する方法を解説します。

3.2.1 stress コマンドのインストール

システムに負荷をかけるために、stress コマンドを使用します。stress コマンドは、CentOS 6 の標準パッケージでは提供されておらず、RPMforge のリポジトリで提供されています。リポジトリを追加して yum コマンドでインストールします。

RPMforge のリポジトリを追加するには、下記のサイトから、ディストリビューションに対応した最新の rpmforge-release パッケージをダウンロードします。

```
http://pkgs.repoforge.org/rpmforge-release/
```

64 ビット版 CentOS 6 の場合、以下のパッケージとなります。パッケージがバージョンアップするとファイル名が変わるので注意が必要です。

```
http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el6.rf.x86_64.rpm
```

コマンドラインでの作業を行っている場合、wget コマンドを使ってダウンロードします。

```
# wget
    http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el6.rf.x86_64.rpm
 (略)

2014-12-24 11:19:30 (19.2 KB/s) - `rpmforge-release-0.5.3-1.el6.rf.x86_64.rpm'
    へ保存完了 [12640/12640]
```

rpm コマンドで rpmforge-release パッケージをインストールします。

```
# ls -l rpmforge-release-0.5.3-1.el6.rf.x86_64.rpm
-rw-r--r--. 1 root root 12640  3月 21 00:59 2013
    rpmforge-release-0.5.3-1.el6.rf.x86_64.rpm
# rpm -ivh rpmforge-release-0.5.3-1.el6.rf.x86_64.rpm
```

yum コマンドで stress パッケージをインストールします。

```
# yum install stress
```

■RPM パッケージの直接取得

インターネットに接続できない場合には、インターネットに接続できる端末で以下の URL から RPM パッケージをダウンロードしてコピーして下さい。

```
http://pkgs.repoforge.org/stress/
http://pkgs.repoforge.org/stress/stress-1.0.2-1.el6.rf.x86_64.rpm
```

3.2.2 top コマンドによるシステムリソース監視

top コマンドは、システムのどのプロセスがどの程度の CPU やメモリなどのリソースを消費しているかを簡単に示してくれる対話型のコマンドです。

top コマンドを実行すると、デフォルトでは先頭五行（サマリーエリア）にシステム全体の情報が表示されます。次の行が対話的にコマンドを入力するエリアです。その次の行から、プロセス毎の情報が表示されます。

```
top - 03:11:49 up 16:28,  4 users,  load average: 0.08, 0.03, 0.01
Tasks: 188 total,   1 running, 187 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0%us,  0.0%sy,  0.0%ni, 99.8%id,  0.2%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   1016372k total,   811796k used,   204576k free,    24736k buffers
Swap:  2064380k total,    41640k used,  2022740k free,   295652k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
    1 root      20   0 19364 1304 1036 S  0.0  0.1   0:01.24 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.03 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   0:00.03 migration/0
    4 root      20   0     0    0    0 S  0.0  0.0   0:00.09 ksoftirqd/0
    5 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 stopper/0
    6 root      RT   0     0    0    0 S  0.0  0.0   0:00.08 watchdog/0
    7 root      RT   0     0    0    0 S  0.0  0.0   0:00.04 migration/1
    8 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 stopper/1
    9 root      20   0     0    0    0 S  0.0  0.0   0:00.07 ksoftirqd/1
   10 root      RT   0     0    0    0 S  0.0  0.0   0:00.06 watchdog/1
   11 root      20   0     0    0    0 S  0.0  0.0   0:03.16 events/0
   12 root      20   0     0    0    0 S  0.0  0.0   0:02.79 events/1
   13 root      20   0     0    0    0 S  0.0  0.0   0:00.00 cgroup
   14 root      20   0     0    0    0 S  0.0  0.0   0:00.01 khelper
   15 root      20   0     0    0    0 S  0.0  0.0   0:00.00 netns
   16 root      20   0     0    0    0 S  0.0  0.0   0:00.00 async/mgr
   17 root      20   0     0    0    0 S  0.0  0.0   0:00.00 pm
```

先頭の 5 行には、以下の情報が表示されています。

行数	意味
1 行目	起動時間、ログインユーザ数、ロードアベレージ
2 行目	各状態のタスク数
3 行目	CPU の使用状況
4 行目	物理メモリの使用状況
5 行目	スワップ領域の使用状況

stress コマンドを実行して、システムに負荷をかけた状態を top コマンドで確認します。

stress コマンドをバックグラウンドで実行します。バックグラウンド実行を行った関係上、コマンドプロンプトが表示された後に stress コマンドのメッセージが表示されます。Enter キーを押せば、再度コマンドプロンプトが表示されます。

```
# stress --cpu 3 --io 4 --vm 2 --vm-bytes 128M &
[1] 9747
# stress: info: [9747] dispatching hogs: 3 cpu, 4 io, 2 vm, 0 hdd
※Enterキーを入力
#
```

top コマンドでプロセスの状況を確認します。stress コマンドが CPU、メモリを使用している様子が確認できます。

```
# top

top - 03:28:09 up 16:44,  3 users,  load average: 16.85, 14.44, 7.86
Tasks: 208 total,  13 running, 195 sleeping,   0 stopped,   0 zombie
Cpu(s): 55.5%us, 44.5%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   1016372k total,   718440k used,   297932k free,     1528k buffers
Swap:  2064380k total,   116124k used,  1948256k free,    39532k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 9692 sato      20   0  6516  176   92 R 17.0  0.0   2:02.20 stress
 9698 sato      20   0  6516  176   92 R 17.0  0.0   2:03.52 stress
 9748 root      20   0  6516  188  104 R 17.0  0.0   0:04.95 stress
 9750 root      20   0  134m 125m  184 R 17.0 12.6   0:05.11 stress
 9754 root      20   0  6516  188  104 R 17.0  0.0   0:05.11 stress
 9694 sato      20   0  134m  24m  168 R 16.6  2.4   2:00.22 stress
 9695 sato      20   0  6516  176   92 R 16.6  0.0   2:02.48 stress
 9751 root      20   0  6516  188  104 R 16.6  0.0   0:04.88 stress
 9697 sato      20   0  134m  59m  168 R 16.3  6.0   2:00.31 stress
 9753 root      20   0  134m  55m  184 R 16.3  5.6   0:04.87 stress
 9755 root      20   0  6516  184  100 D  4.7  0.0   0:01.50 stress
 9756 root      20   0  6516  184  100 D  4.7  0.0   0:01.49 stress
 9696 sato      20   0  6516  172   88 R  4.0  0.0   0:54.59 stress
 9699 sato      20   0  6516  172   88 D  4.0  0.0   0:59.14 stress
 9693 sato      20   0  6516  172   88 D  2.0  0.0   0:57.48 stress
 9700 sato      20   0  6516  172   88 D  2.0  0.0   0:59.43 stress
 9749 root      20   0  6516  184  100 D  2.0  0.0   0:01.60 stress
```

q キーを押して、top コマンドを終了します。バックグラウンドで動作している stress コマンドも fg コマンドでフォアグラウンド実行に変更して、終了します。

```
# fg
stress --cpu 3 --io 4 --vm 2 --vm-bytes 128M
※^C ←Ctrl+Cキーを入力
```

3.2.3 vmstat コマンドによるシステムリソース監視

vmstat コマンドは、メモリの使用状況や CPU の負荷などを表示するコマンドです。

vmstat コマンドを引数無しで実行すると、コマンドを実行した時点のメモリや CPU、ディスクの使用状況が表示されます。

```
# vmstat
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 8  0 116104 408536  58692  71292    0    1    10    11  251   66  2  2 97  0  0
```

表示される内容は以下の表のとおりです。

項目	意味
r	実行待ちプロセス数
b	割り込み不可のスリープ状態にあるプロセス数
swpd	スワップアウトされたメモリ量
free	空きメモリの容量
buff	バッファに使用されているメモリの容量
cache	キャッシュに使用されているメモリの容量
si	1 秒あたりのディスクからスワップインされているメモリの容量
so	1 秒あたりのディスクにスワップアウトされているメモリの容量
bi	1 秒あたりのブロックデバイスに送られたブロック
bo	1 秒あたりのブロックデバイスから受け取ったブロック
in	1 秒あたりの割り込みの回数
cs	1 秒あたりのコンテキストスイッチの回数
us	CPU 総時間当たりのユーザー時間の割合
sy	CPU 総時間当たりのシステム時間の割合
id	CPU 総時間当たりのアイドル時間の割合

vmstat コマンドに引数として数値を与えると、秒間隔でシステムのリソース情報を出力し続けます。終了するには Ctrl+C キーを入力します。

```
# vmstat 5
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
10  0 116104 261708  65040  79460    0    1    11    11  253   70  2  2 97  0  0
 9  0 116104 358068  65712  80356    0    0   189   242 5411 8564 42 58  0  0  0
 7  0 116104 301924  66184  81372    0    0   202   308 4610 7441 41 59  0  0  0
※^C Ctrl+Cキーを入力する
```

3.2.4 sysstat によるシステムリソース監視

稼働中の Linux のシステム情報を継続して集めるには、sysstat パッケージに含まれている iostat コマンドや sar コマンドなどを使うと便利です。

sysstat パッケージをインストールします。

```
# yum install sysstat
```

sysstat パッケージをインストールすると、デフォルトで 10 分間隔でシステムのリソース情報が取得されるように cron が設定されます。

```
# cat /etc/cron.d/sysstat
```

```
# Run system activity accounting tool every 10 minutes
*/10 * * * * root /usr/lib64/sa/sa1 1 1
# 0 * * * * root /usr/lib64/sa/sa1 600 6 &
# Generate a daily summary of process accounting at 23:53
53 23 * * * root /usr/lib64/sa/sa2 -A
```

10 分間隔で起動される/usr/lib64/sa/sa1 スクリプトは、内部で/usr/lib64/sa/sadc を実行し、システムのリソース情報を取得して/var/log/sa/saDD ファイル（DD は 2 桁の日付）に保存しています。23:53 に実行される/usr/lib64/sa/sa2 スクリプトは、sa1 スクリプトで取得した情報をまとめて/var/log/sa/sarDD（DD は 2 桁の日付）というファイルを生成し、古くなった情報を削除します。デフォルトでは 28 日分を保存しておきます。期間を変更したい場合には設定ファイル/etc/sysconfig/sysstat ファイル内の HISTORY 変数を変更します。

まとめられた情報は、後述する sar コマンドで参照できます。

3.2.5 iostat コマンドによるシステムリソース監視

sysstat パッケージに含まれる iostat コマンドは、CPU の使用率や各種 I/O の利用状況を確認するためのコマンドです。I/O は、ハードディスクやテープドライブ、ネットワークマウントしたファイルシステム、端末入出力等の入出力性能を監視できます。

iostat コマンドを実行すると、システムが起動してから iostat コマンドを実行した時点までの間の CPU および I/O の状況が表示されます。

```
# iostat
Linux 2.6.32-504.el6.x86_64 (server.example.com)    2015年01月15日      _x86_64(2 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           1.72    0.00    1.95    0.03    0.00   96.30

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
sda               1.89        44.06       117.04    2720068    7224884
scd0              0.01         0.18         0.00      11204          0
dm-0              6.51        41.98        42.57    2591466    2627904
dm-1              0.49         0.17        74.44      10552    4595040
dm-2              0.01         0.06         0.03       3522       1856
```

表示される内容は以下の表のとおりです。

項目	意味
%user	ユーザプロセスによる CPU の使用率
%nice	実行優先度（nice 値）を変更したユーザプロセスによる CPU の使用率
%system	システムプロセスによる CPU の使用率
%iowait	I/O 終了待ちとなった CPU の使用率
%steal	ハイパーバイザーによる他の仮想 CPU の実行待ちとなった CPU の使用率
%idle	CPU が何も処理をせずに待機していた時間の割合 (ディスク I/O 以外)
tps	1 秒間の I/O 転送回数
Blk_read/s	1 秒間のディスクの読み込み量 (ブロック数)
Blk_wrtn/s	1 秒間のディスクの書き込み量 (ブロック数)
Blk_read	ディスクの読み込み量 (ブロック数)
Blk_wrtn	ディスクの書き込み量 (ブロック数)

iostat コマンドに-x オプションを付与して実行すると、表示が KB 単位に変わります。

項目	意味
kB_read/s	1 秒間のディスクの読み込み量 (KB 単位)
kB_wrtn/s	1 秒間のディスクの書き込み量 (KB 単位)
kB_read	ディスクの読み込み量 (KB 単位)
kB_wrtn	ディスクの書き込み量 (KB 単位)

iostat コマンドに引数として数値を与えて実行すると、1 回目の表示はシステム起動から iostat コマンド実行時までの間の情報ですが、その後指定された秒間隔で全てのデバイスの I/O の利用状況が出力されます。終了するには Ctrl+C を入力します。。

```
# iostat 5
Linux 2.6.32-504.el6.x86_64 (server.example.com)    2015年01月15日      _x86_64(2 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           1.76    0.00    2.01    0.03    0.00   96.20

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
sda               1.89        44.02       116.93    2720092    7225892
scd0              0.01         0.18         0.00      11204          0
dm-0              6.51        41.94        42.54    2591474    2628888
dm-1              0.49         0.17        74.36      10552    4595040
dm-2              0.01         0.06         0.03       3522       1856

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          44.30    0.00   55.70    0.00    0.00    0.00

Device:            tps   Blk_read/s   Blk_wrtn/s   Blk_read   Blk_wrtn
sda               0.00         0.00         0.00          0          0
scd0              0.00         0.00         0.00          0          0
dm-0              0.00         0.00         0.00          0          0
dm-1              0.00         0.00         0.00          0          0
dm-2              0.00         0.00         0.00          0          0

※^C ←Ctrl+Cキーを入力する
```

iostat コマンドに-x オプションを付与して実行します。結果が拡張フォーマットで表示されます。

```
# iostat -x
Linux 2.6.32-504.el6.x86_64 (server.example.com)    2015年01月15日      _x86_64(2 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           1.78    0.00    2.04    0.03    0.00   96.16

Device:         rrqm/s   wrqm/s     r/s     w/s   rsec/s   wsec/s avgrq-sz avgqu-sz
     await  svctm  %util
sda               0.83     4.90    0.83    1.06    44.00   116.88    85.16     0.00
     0.57   0.30   0.06
scd0              0.04     0.00    0.01    0.00     0.18     0.00    27.00     0.00
     14.24   9.61   0.01
dm-0              0.00     0.00    1.17    5.33    41.92    42.52    12.98     0.02
     3.17   0.10   0.06
dm-1              0.00     0.00    0.02    0.47     0.17    74.33   150.83     0.00
     1.80   0.03   0.00
```

```
dm-2              0.00     0.00    0.01    0.00     0.06     0.03     7.97     0.00
     0.37    0.27    0.00
```

表示される内容は以下の表のとおりです。

項目	意味
rrqm/s	1 秒間デバイスへマージされた読み込みリクエスト数
wrqm/s	1 秒間デバイスへマージされた書き込みリクエスト数
r/s	1 秒間の読み込みリクエスト数
w/s	1 秒間の書き込みリクエスト数
rsec/s	1 秒間の読み込みセクタ数
wsec/s	1 秒間の書き込みセクタ数
rkB/s	1 秒間の読み込みキロバイト（KB）数
wkB/s	1 秒間の読み込みキロバイト（KB）数
avgrq-sz	デバイスへの IO リクエストの平均サイズ
avgqu-sz	デバイスへの IO リクエストのキューの平均サイズ
await	デバイスへの IO リクエストの平均待ち時間
svctm	デバイスへの IO リクエストの平均処理時間
%util	デバイスへの IO リクエスト期間 CPU の使用率

3.2.6 sar（System Admin Reporter）によるシステムリソース監視

sar コマンドは CPU やネットワーク、メモリ、ディスクなどのシステム情報を確認・出力するためのコマンドです。sar コマンドで様々なシステム情報を取得、出力できるので、障害発生時の障害を特定するために利用されます。

また、オプションでファイルを指定することにより、過去に取得している sar や sadc のバイナリ出力ファイルから利用状況を抜き出すことができます。sysstat パッケージをインストールした際に設定された cron による情報収集の結果も、sar コマンドで確認できます。

sar コマンドに引数を与えて実行します。例では 1 秒間隔で 3 回、CPU の情報を監視します。

```
# sar 1 3
Linux 2.6.32-504.el6.x86_64 (server.example.com)    2015年01月23日      _x86_64(2 CPU)

18時25分47秒      CPU      %user      %nice   %system   %iowait    %steal      %idle
18時25分48秒      all      38.00       0.00     62.00       0.00       0.00       0.00
18時25分49秒      all      38.50       0.00     61.50       0.00       0.00       0.00
18時25分50秒      all      39.80       0.00     60.20       0.00       0.00       0.00
平均値:       all      38.77       0.00     61.23       0.00       0.00       0.00
```

sar コマンドを-b オプションを付与して実行します。ディスク I/O の利用状況を監視します。

```
# sar -b 1 3
Linux 2.6.32-504.el6.x86_64 (server.example.com)    2015年01月23日      _x86_64(2 CPU)

18時26分15秒       tps      rtps      wtps   bread/s   bwrtn/s
18時26分16秒      0.00      0.00      0.00      0.00      0.00
18時26分17秒      0.00      0.00      0.00      0.00      0.00
18時26分18秒    352.00    142.00    210.00   5648.00   1904.00
平均値:     117.73     47.49     70.23   1888.96    636.79
```

sar コマンドを-r オプションを付与して実行します。メモリやスワップの利用状況を監視します。

```
# sar -r 1 3
```

```
Linux 2.6.32-504.el6.x86_64 (server.example.com)    2015年01月23日      _x86_64(2 CPU)

18時26分32秒 kbmemfree kbmemused  %memused kbbuffers  kbcached  kbcommit   %commit
18時26分33秒     233684    782688     77.01     81008    152872   1562412     50.72
18時26分34秒     101404    914968     90.02     81008    152872   1562412     50.72
18時26分35秒     112552    903820     88.93     81008    152872   1562412     50.72
平均値:      149213    867159     85.32     81008    152872   1562412     50.72
```

sar コマンドを引数無しで実行すると、その日に実行された sysstat の結果が表示されます。

```
# sar
Linux 2.6.32-504.el6.x86_64 (server.example.com)    2015年01月23日      _x86_64(2 CPU)

11時10分01秒      CPU     %user     %nice   %system   %iowait    %steal     %idle
11時20分01秒      all      0.39      0.00      0.36      0.01      0.00     99.24
11時30分02秒      all      9.34      0.00     12.22      0.04      0.00     78.39
11時40分01秒      all     43.10      0.00     56.90      0.00      0.00      0.00
（略）
```

sar コマンドに-f オプションを付与して実行します。オプションの値として/var/log/sa/saDD ファイルを指定します。

```
# sar -f /var/log/sa/sa22
Linux 2.6.32-504.el6.x86_64 (server.example.com)    2015年01月22日      _x86_64(2 CPU)

12時10分02秒      CPU     %user     %nice   %system   %iowait    %steal     %idle
12時20分01秒      all      0.33      0.00      0.34      0.01      0.00     99.32
12時30分01秒      all      0.39      0.00      0.34      0.02      0.00     99.25
平均値:       all      0.36      0.00      0.34      0.01      0.00     99.29
（略）
```

/var/log/sa/sarDD ファイルは 1 日の監視結果を集計したテキストファイルです。less コマンドなどで参照できます。ただし、毎日 23 時 53 分にシステムが動作していないと sarDD ファイルは作成されません。その場合には、root ユーザで以下のコマンドを実行するとその日の sarDD ファイルが作成されます。

```
# /usr/lib64/sa/sa2 -A
# cat /var/log/sa/sar24
Linux 2.6.32-504.el6.x86_64 (server.example.com)    2015-01-23  _x86_64(2 CPU)

11時10分01秒      CPU      %usr     %nice      %sys   %iowait    %steal      %irq
     %soft    %guest     %idle
11時20分01秒      all      0.39      0.00      0.35      0.01      0.00      0.00
     0.01      0.00     99.24
11時20分01秒        0      0.44      0.00      0.36      0.02      0.00      0.00
     0.02      0.00     99.17
（略）
```

3.2.7 logwatch によるメール通知

サーバに出力されたログには、問題が発生した兆候がログとして出力されるものがあります。また、セキュリティ上、不正なアクセスなどもログに記録されます。

logwatch は、サーバのログを見やすいレポートにまとめて毎日メールで送信したり、特定のパターンが含まれるログが出た際にメールで通知を出すように設定できます。これにより、ログのチェックを簡略化することができます。

logwatch をインストールします。

```
# yum install logwatch
```

logwatch を設定します。logwatch.conf のデフォルト設定は/usr/share/logwatch/default.conf/logwatch.conf に記述されています。このデフォルト設定から値を変更したいものを/etc/logwatch/conf/logwatch.conf に記述します。

設定できる値は以下の表のとおりです。

■LogDir

チェックするログの格納先を指定

■TmpDir

一時的なファイルの保存先

■MailTo

結果レポートのメール送信先を指定

■MailFrom

結果レポートのメール送信元を指定

■Print

結果を標準出力（STDOUT）に出力（Yes)、あるいは MailTo 宛にメール送信（No）

■Save

結果レポートをファイルとして保存標準では無効（コメントアウト）: 保存しない

■Archives

アーカイブされたファイルも調査（Yes）標準では無効（コメントアウト）: 調査しない

■Range

チェック対象となるログファイルの日付範囲を指定すべて（All)、当日（Today)、昨日（Yesterday）

■Detail

結果レポートの詳細レベル Low（0)、Med（5)、High（10）のいずれかを指定

■Service

LogWatch でチェックの対象となるサービスを指定/usr/share/logwatch/scripts/services 以下のファイルが対象

■LogFile

特定のログファイルのみをチェック標準では無効（コメントアウト）: すべてチェック

■mailer

メール送信で用いるメールプログラムを指定

■HostLimit

特定のホスト名（hostname コマンドの結果）に関するログエントリのみチェック標準では無効（コメントアウト）: 制限しない

一般的には、MailTo、Detail などを変更します。デフォルトの設定ファイルをコピーして、必要な設定を変更するとよいでしょう。

```
# cp /usr/share/logwatch/default.conf/logwatch.conf /etc/logwatch/conf/logwatch.conf
cp: `/etc/logwatch/conf/logwatch.conf' を上書きしてもよろしいですか(yes/no)? ※y yを入力
```

以下はデフォルト値の抜粋です。

```
MailTo = root
Range = yesterday
Detail = Low
Service = All
```

デフォルト設定では、ローカルのユーザ root 宛に昨日の結果を最小限でメール送信します。対象となるサービスは/usr/share/logwatch/scripts/services ディレクトリ内に用意されているサービスです。

```
# ls /usr/share/logwatch/scripts/services
afpd            eximstats         pam_unix          sendmail-largeboxes
amavis          extreme-networks  php               shaperd
arpwatch        fail2ban          pix               slon
audit           ftpd-messages     pluto             smartd
automount       ftpd-xferlog      pop3              sonicwall
autorpm         http              portsentry        sshd
bfd             identd            postfix           sshd2
cisco           imapd             pound             stunnel
clam-update     in.qpopper        proftpd-messages  sudo
clamav          init              pureftpd          syslogd
clamav-milter   ipop3d            qmail             tac_acc
courier         iptables          qmail-pop3d       up2date
cron            kernel            qmail-pop3ds      vpopmail
denyhosts       mailscanner       qmail-send        vsftpd
dhcpd           modprobe          qmail-smtpd       windows
dnssec          mountd            raid              xntpd
dovecot         named             resolver          yum
dpkg            netopia           rt314             zz-disk_space
emerge          netscreen         samba             zz-fortune
evtapplication  oidentd           saslauthd         zz-network
evtsecurity     openvpn           scsi              zz-runtime
evtsystem       pam               secure            zz-sys
exim            pam_pwdb          sendmail
```

/etc/logwatch/conf/logwatch.conf の設定を変更して、すべての期間のログファイルをチェックするように記述します。

```
# vi /etc/logwatch/conf/logwatch.conf

※#※Range = yesterday ※←行頭に#を追加
```

```
※Range = All ← 新規に追加
```

logwatch の出力テストを行います。logwatch コマンドに–print オプションを付与して実行すると、結果が標準出力に表示されます。

```
# logwatch --print


 ##################### Logwatch 7.3.6 (05/19/07) ####################
        Processing Initiated: Tue Jan 27 11:53:04 2015
        Date Range Processed: all
       Detail Level of Output: 0
              Type of Output: unformatted
           Logfiles for Host: server.example.com
  ################################################################

 ---------------------- Selinux Audit Begin ------------------------

  Number of audit daemon stops: 1

 ----------------------- Selinux Audit End -------------------------
 (略)
 --------------------- Disk Space Begin ------------------------

 Filesystem            Size  Used Avail Use% Mounted on
 /dev/mapper/vg_server-lv_root
                        50G  3.8G   43G   9% /
 /dev/sda1             477M   28M  424M   7% /boot
 /dev/mapper/vg_server-lv_home
                        12G   31M   11G   1% /home


 ---------------------- Disk Space End -------------------------


 ###################### Logwatch End #########################

再度、/etc/logwatch/conf/logwatch.confを設定します。今日のログファイルをチェックするように記述しま

```shell-session
# vi /etc/logwatch/conf/logwatch.conf

Range = Today
```

再度 logwatch コマンドに–print オプションを付与して実行します。結果が短くなったことを確認します。

4 トラブルシューティング

4.1 ログ管理

システム障害の問題解決をはかるトラブルシューティングを行う場合に、もっとも有益な情報源がログです。

ログには、OS が出力するログ、アプリケーションが出力するログなど多くの種類が存在します。
```

ここでは、代表的なログの種類と確認方法、設定方法などを解説します。

4.1.1 ログの種類

CentOS では、ログファイルは/var/log ディレクトリ以下に格納されています。

以下は代表的なログファイルです。

ファイル名	内容
messages	サービス起動時の出力など一般的なログ
secure	認証、セキュリティ関係のログ
maillog	メール間連のログ
dmesg	カーネルが出力したメッセージのログ

4.1.2 ログの確認

サーバのログにサービス起動時、または動作時のエラーログが記録されていないかを確認します。また、クライアント側にもエラーログが記録されていないかを確認します。

- 一般的なトラブルであれば、まずは/var/log/messages を確認します。
- 認証関係やアクセス制限に関係するトラブルは/var/log/secure を確認します。
- メール関係であれば/var/log/maillog を確認します。
- Web サーバであれば/var/log/httpd/error_log などを確認します。

4.1.3 dmesg に記録されるログ

dmesg コマンドは「display message」の略で、Linux カーネルがメッセージを出力するリングバッファ（循環バッファ）の内容を表示します。このリングバッファは一定のサイズ内で循環するようになっており、古いログは消えていきます。dmesg コマンドを用いることにより、システム起動時に出力されるカーネルメッセージの確認ができます。カーネルが正しくハードウェアを認識しているかどうかを確認する場合などに参照します。

```
# dmesg
Initializing cgroup subsys cpuset
Initializing cgroup subsys cpu
Linux version 2.6.32-504.el6.x86_64 (mockbuild@c6b9.bsys.dev.centos.org) (gcc version
    4.4.7 20120313 (Red Hat 4.4.7-11) (GCC) ) #1 SMP Wed Oct 15 04:27:16 UTC 2014
Command line: ro root=/dev/mapper/vg_server-lv_root rd_LVM_LV=vg_server/lv_swap
    rd_NO_LUKS rd_LVM_LV=vg_server/lv_root rd_NO_MD crashkernel=auto  KEYBOARDTYPE=pc
    KEYTABLE=jp106 LANG=ja_JP.UTF-8 rd_NO_DM rhgb quiet
KERNEL supported cpus:
  Intel GenuineIntel
  AMD AuthenticAMD
  Centaur CentaurHauls
Disabled fast string operations
 （略）
```

4.1.4 syslog について

syslog は、カーネルやプログラムなどから出力されるログをまとめて記録する仕組みです。syslog を使うことで、各プログラムは独自にログを記録する仕組みを開発する必要が無くなります。また、syslog サーバをネットワーク上で動作させることで、複数のホストからのログをまとめて記録することで、ログを一元管理することもできます。CentOS 6 では、syslog サーバとして rsyslog が使用できます。

rsyslog は、従来の syslog デーモン（syslogd）に置き換わる、マルチスレッドの syslog デーモンです。rsyslog（Reliable syslog）という名前からも分かる通り、高い信頼性を実現するように開発されています。そのため、ログの転送に TCP を使用したり、

データベースへのログ保存、暗号化したログの転送なども行うことができます。基本的な設定については、従来の syslogd と互換性があります。

4.1.5 ファシリティとプライオリティ

カーネルやプログラムが出力する syslog メッセージには、「ファシリティ」（facility）と「プライオリティ」（priority）と呼ばれる値が設定されています。

ファシリティは、何がそのログメッセージを生成したのかを指定します。たとえば、カーネルやメールといった値が指定されます。

また、プライオリティはメッセージの重要性を指定します。たとえば、単なる情報、非常に危険な状態などといった値が指定されます。

ファシリティには、以下の種類があります。

ファシリティ	意味
auth	セキュリティ・認証関連（login、su など）
authpriv	セキュリティ・認証関連（プライベート）
cron	cron や at のログ
daemon	一般的なデーモン（サーバプログラム）関連
kern	カーネル関連
lpr	プリンタ関連
mail	メール関連
news	NetNews 関連
security	auth と同じ
syslog	syslogd 自身のログ
user	ユーザアプリケーションのログ
uucp	uucp 転送を行うプログラムのログ
local0 から local7	独自のプログラムで利用可能な facility

プライオリティには、以下の種類があります。

プライオリティ	意味
debug	デバッグ用メッセージ
info	一般的な情報メッセージ
notice	通知メッセージ
warning	警告メッセージ
warn	warning と同じ
err	一般的なエラーメッセージ
error	err と同じ
crit	ハード障害などの危険なエラーメッセージ
alert	システム破損などの緊急事態
emerg	非常に危険な状態
panic	emerg と同じ
none	ファシリティを無効にする

4.1.6 syslog サーバの設定

syslog サーバの設定ファイルである/etc/rsyslog.conf には、受け取ったログメッセージをファシリティとプライオリティの組み合わせでどのファイルに出力するかの設定が記述されています。

記述は以下の形式となります。

```
ファシリティ.プライオリティ　アクション
```

syslog サーバの設定ファイル中で、複数のファシリティを指定したい場合には、「,」（コンマ）で区切ります。たとえば、UUCP 転送とメール関連のファシリティを同時に指定したい場合には、以下のように指定します。

```
uucp,news.crit  /var/log/spooler
```

syslog 設定ファイル中でプライオリティを指定すると、そのプライオリティ以上の重要度のプライオリティがすべて当てはまります。たとえば、以下のように設定したとします。

```
mail.warning
```

mail ファシリティからの warning 以上（err、crit、alert、emerg）のすべてのプライオリティが当てはまります。

特定のプライオリティのみ指定したい場合には、「=プライオリティ」と指定します。

```
mail.=warning
```

この指定は mail ファシリティのプライオリティが warning のメッセージのみが当てはまります。

none ファシリティはやや特殊な動きをするので、後述の例で解説します。

4.1.7　アクションの設定

ファシリティとプライオリティを記述した右側に、該当するログをどうするかを指定するアクションを記述します。

主なアクションは、以下の表のとおりです。

■ファイル名

ログをファイルに書き込む。

■-ファイル名

ログをファイルに書き込む際にバッファリングする。書き込み性能が向上するが、書き込まれていないデータがある時にシステム障害が発生するとログが失われる。

■\プログラム

ログメッセージをプログラムに引き渡す。

■* すべてのユーザのコンソールにメッセージを表示する。

■[@ホスト名]（あるいは IP アドレス）

UDP で syslog サーバにログメッセージを送信する。

■@[@ホスト名]（あるいは IP アドレス）

TCP で syslog サーバにログメッセージを送信する。

4.1.8 syslog サーバのデフォルト設定を確認する

設定ファイル/etc/rsyslog.conf に既に設定されている内容を確認します。

```
authpriv.*  /var/log/secure
```

この設定は、ファシリティが authpriv（認証関係)、プライオリティが*（全てのプライオリティ）のログメッセージは/var/log/secure に出力するように指定しています。

```
*.info;mail.none;authpriv.none;cron.none        /var/log/messages
```

この設定は、すべてのファシリティの info プライオリティ以上のログをすべて/var/log/messages に出力するようにしています。ただし、mail、authpriv、cron の 3 つのファシリティには none プライオリティが指定されているため、対象からは除外されています。

除外された各ファシリティの出力は、以下のように別途指定されています。

mail ファシリティのログは、メモリ上にある程度バッファリングした上でログファイルに書き込むように「-（ハイフン)」を指定しています。メールサーバは一度に大量のログを書き込むことが多いからです。

```
authpriv.*                      /var/log/secure
mail.*                          -/var/log/maillog
cron.*                          /var/log/cron
```

4.1.9 カーネルログの syslog 出力設定

デフォルトの設定ではコメントアウトされて無効になっているカーネルからのログ出力の設定を有効にします。カーネルのログは、たとえば iptables のようなカーネルの機能がログを出力します。

iptables の設定ファイル/etc/sysconfig/iptables を編集し、ポート番号 22 番の許可（ACCEPT）と、その他の全てを拒否（REJECT）するルールの間に、ログを取得するルールを追加します。

```
# Firewall configuration written by system-config-firewall
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
※-A INPUT -j LOG --log-level debug --log-prefix '[iptables_test]:' ←新規に追加
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

iptables サービスを reload して、新しい設定を読み込ませます。

```
# service iptables reload
iptables: Trying to reload firewall rules:                 [  OK  ]
```

/etc/rsyslog.conf を編集し、ファシリティが kern、プライオリティが全てのメッセージを/var/log/kern.log に出力する設定を追加します。

```
# vi /etc/rsyslog.conf

# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                                                 /dev/console
※kern.*                                                 /var/log/kern.log ←新規に追加
```

rsyslog サービスを再起動して、新しい設定を読み込ませます。

```
# service rsyslog restart
システムロガーを停止中:                                    [  OK  ]
システムロガーを起動中:                                    [  OK  ]
```

外部のホストから設定を行ったホストに対して、iptables で許可されていないポート番号 80 番に Web ブラウザ等でアクセスします。

/var/log/kern.log にポート番号 80 番に対する通信を拒否した旨のログが出力されます。

```
# tail /var/log/kern.log
Dec 25 14:54:16 server kernel: imklog 5.8.10, log source = /proc/kmsg started.
Dec 25 14:54:50 server kernel: ※'[iptables_test]:'※IN=eth0 OUT=
     MAC=00:1c:42:65:af:c4:00:1c:42:00:00:08:08:00 SRC=192.168.0.2 DST=192.168.0.10
     LEN=64 TOS=0x00 PREC=0x00 TTL=64 ID=24955 DF PROTO=TCP SPT=57191 ※DPT=80※
     WINDOW=65535 RES=0x00 SYN URGP=0
```

4.1.10 リモートホストのログを UDP で受け取る

syslog サーバとしてリモートホストのログを受け取るための設定を行います。syslog のメッセージの送受信は、通常 UDP で行われます。

設定ファイル/etc/rsyslog.conf 内にある以下の 2 行から、行頭のコメントアウトを削除して設定を有効にします。

*ModLoad*は、*UDP*用のプロトコルモジュールのロードを設定しています。UDPServerRun は、UDP でログメッセージを受け取るポート番号を指定しています。

```
[root@server ~]## vi /etc/rsyslog.conf

 (略)
# Provides UDP syslog reception
$ModLoad imudp ※←行頭の#を削除
$UDPServerRun 514 ※←行頭の#を削除
```

rsyslog サービスを再起動します。rsyslogd が UDP のポート番号 514 番で待ち受けるようになります。

```
[root@server ~]# service rsyslog restart
システムロガーを停止中:                                    [  OK  ]
システムロガーを起動中:                                    [  OK  ]
[root@server ~]# lsof -i:514
COMMAND   PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
rsyslogd 9282 root    3u  IPv4 134339      0t0  UDP *:syslog
rsyslogd 9282 root    4u  IPv6 134340      0t0  UDP *:syslog
```

設定後、iptables の設定を変更し、UDP のポート番号 514 番へのパケットを許可するように設定を変更する必要があります。設定については後述します。

4.1.11 リモートホストのログを TCP で受け取る

ログメッセージの送受信に TCP を使用することにより、UDP で発生していたログの取りこぼしを防ぐことができます。UDP はセッションレスなプロトコルのため、送受信に失敗した時に再送信する仕組みが無いためです。

ただし、TCP はプロトコルの性質上 UDP よりも処理が重くなってしまうため、大量のログが送信されてくる環境では逆にボトルネックになってしまい、syslog サーバ側が高負荷で処理が滞ってしまう可能性があります。

そのため、TCP を使ったログメッセージの送受信は、ログの量がそれほど多くなくログ記録の信頼性が必要な場合に設定します。もし、大量のログが送信されてくる場合には、syslog サーバを複数用意するか、UDP を使う必要があります。

設定ファイル/etc/rsyslog.conf 内にある以下の 2 行から、行頭のコメントアウトを削除して設定を有効にします。

*ModLoad*は、*TCP*用のプロトコルモジュールのロードを設定しています。InputTCPServerRun は、TCP でログメッセージを受け取るポート番号を指定しています。

```
[root@server ~]# vi /etc/rsyslog.conf

（略）
# Provides TCP syslog reception
$ModLoad imtcp ※←行頭の#を削除
$InputTCPServerRun 514 ※←行頭の#を削除
```

rsyslog サービスを再起動します。rsyslogd が TCP のポート番号 514 番で待ち受けるようになります。

```
[root@server ~]# service rsyslog restart
シ ス テ ム ロ ガ ー を 停 止 中 :                                    [  OK  ]
シ ス テ ム ロ ガ ー を 起 動 中 :                                    [  OK  ]
[root@server ~]# lsof -i:514
COMMAND    PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
rsyslogd 24138 root    1u  IPv4 107209      0t0  TCP *:shell (LISTEN)
rsyslogd 24138 root    3u  IPv4 107202      0t0  UDP *:syslog
rsyslogd 24138 root    4u  IPv6 107203      0t0  UDP *:syslog
rsyslogd 24138 root    8u  IPv6 107210      0t0  TCP *:shell (LISTEN)
```

ポートが shell と表示されているのは、ポート番号の設定ファイル/etc/services で定義されているためです。動作に影響はありません。

```
# grep 514 /etc/services
shell           514/tcp         cmd             # no passwords used
syslog          514/udp
（略）
```

設定後、iptables の設定を変更し、TCP のポート番号 514 番へのパケットを許可するように設定を変更する必要があります。

4.1.12 syslog サーバの iptables の設定

syslog サーバの iptables の設定を変更し、TCP および UDP のポート番号 514 番の接続を許可しておきます。あるいは、iptables サービスを停止しておきます。

```
[root@server ~]# service iptables stop
iptables: チェインをポリシー ACCEPT へ設定中filter          [  OK  ]
iptables: ファイアウォールルールを消去中:                    [  OK  ]
iptables: モジュールを取り外し中:                            [  OK  ]
```

/etc/sysconfig/iptables への iptables のルールを追加するには、以下のようになります。パケットを Reject するルールの前に、ルール設定を追加します。ルール設定を追加したら iptables サービスを reload しておきます。

```
[root@server ~]# vi /etc/sysconfig/iptables
（略）
※-A INPUT -m state --state NEW -m udp -p udp --dport 514 -j ACCEPT ←新規に追加
※-A INPUT -m state --state NEW -m tcp -p tcp --dport 514 -j ACCEPT ←新規に追加
-A INPUT -j REJECT --reject-with icmp-host-prohibited
```

4.1.13 syslog クライアントの設定

ネットワークで接続された syslog サーバに対してログメッセージを送信する syslog クライアントを設定します。

syslog クライアント側のホストでも rsyslog を設定し、アクションの設定でネットワーク上の syslog サーバを指定します。

syslog クライアントの設定ファイル/etc/rsyslog.conf を修正します。

authpriv ファシリティに関するすべてのログを syslog サーバに送信するように設定を追加します。[@送信先と指定することで UDP を使用した送信を指定できます]。

また、mail ファシリティに関するすべてのログを syslog サーバに送信するように設定を追加します。@[@送信先と指定することで TCP を使用した送信を指定できます]。

```
# vi /etc/rsyslog.conf

# The authpriv file has restricted access.
authpriv.*                                              /var/log/secure
※authpriv.*                                              @192.168.0.10 ←新規に追加

# Log all the mail messages in one place.
mail.*                                                  -/var/log/maillog
※mail.*                                                  @@192.168.0.10 ←新規に追加```

syslogクライアントのrsyslogサービスを再起動します。

```shell-session
[root@client ~]# service rsyslog restart
システムロガーを停止中:                                    [  OK  ]
システムロガーを起動中:                                    [  OK  ]
```

■UDP でログを送信

syslog クライアントで logger コマンドを実行して、authpriv.debug プライオリティでログを出力します。

```
[root@client ~]# logger -p authpriv.debug "This is auth log over UDP"
```

syslog サーバ上の/var/log/secure にログが出力されることを確認します。

```
[root@server ~]# tail -f /var/log/secure
（略）
Dec 25 17:16:50 client root: This is auth log over UDP
```

■TCP でログを送信

syslog クライアントで logger コマンドを実行して、mail.debug プライオリティでログを出力します。

```
[root@client ~]# logger -p mail.debug "This is mail log over TCP"
```
```

syslog サーバ上の/var/log/maillog にログが出力されることを確認します。

```
[root@server ~]# tail /var/log/secure
（略）
Dec 25 17:18:03 client root: This is mail log over TCP
```

4.1.14 logrotate によるログローテーション

ログファイルは常に追記されていくため、ファイルサイズが次第に肥大化してディスク容量を圧迫し、後でログを確認する際に必要なログを見つけにくくなります。これらの問題を回避するため、ログを一定期間でローテーションする logrotate が使われています。

logrotate は、cron から 1 日 1 回、/etc/cron.daily/logrotate スクリプトによって起動されます。/etc/logrotate.conf が logrotate の設定ファイルとなっており、ログファイルをローテーションするタイミングや、ログファイルを何世代まで残すかなどの設定が記述されています。サービス毎の詳細な設定は、/etc/logrotate.d ディレクトリに格納されています。

logrotate の設定で使用できるディレクティブは以下のとおりです。

■create [モード] [所有ユーザ] [所有グループ]

ローテーションを行った後、代わりに空の新規ログファイルを作成します。属性も指定できます。モードは 0755 のような数値書式。指定しない属性については元のファイルの属性が引き継がれます。

■nocreate

create をグローバルに設定した場合に、個別に create を無効にしたい際に使用します。

■copy/nocopy

元のログファイルはそのままにして、コピーを保存します。

■copytruncate/nocopytruncate

copy の動作を行った後、元のログファイルの内容を消去します。見かけ的には create と同じ結果となります。これはログファイルをリロードする方法が無いプログラムへの対処法のひとつです。たとえば Oracle 10g R1/R2 の alert ログに対しては、この方法を行わないと以前のログファイル（例えば alert_xx.log.1）にログが書き込み続けられます。

■rotate 世代数

世代ローテーションの世代数を指定します。たとえば元のログファイルが a.log の場合、num に 2 を指定すると、a.log → a.log.1 → a.log.2 →廃棄となります。0 の場合、a.log →廃棄となります。

■start 数値

最初のローテーションファイルの末尾に付加する値を指定します。デフォルトは 1 です。たとえば num に 5 を指定すると、a.log → a.log.5 → a.log.6 となります。

■extension 拡張子

ローテーションした旧ログファイルに付ける拡張子を指定します。指定には区切りのドットも必要です。たとえば拡張子に「.bak」と指定すると、some.log の初代ローテーションログは some.log.1.bak となります。圧縮も行う場合、圧縮による拡張子はさらにその後ろに付きます。

■compress/nocompress

ローテーションした後の旧ファイルに圧縮を掛けます。デフォルトは nocompress（非圧縮）です。

■compresscmd コマンド

ログファイルの圧縮に使用するプログラムを指定します。デフォルトは gzip です。

■uncompresscmd コマンド

ログファイルの解凍に使用するプログラムを指定します。デフォルトは gunzip です。

■compressoptions オプション

圧縮プログラムへ渡すオプションを指定します。デフォルトは gzip に渡す「-9」（圧縮率最大）です。「-9 -s」のようにスペース入りで複数のオプションを指定することはできません。

■compressext 拡張子

圧縮後のファイルに付ける拡張子（ドットも必要）を指定します。デフォルトでは、使用する圧縮コマンドに応じたものが付けられます。

■delaycompress/nodelaycompress

圧縮処理を次のローテーションまで遅らせる、あるいは遅らせません。

■olddir ディレクトリ/noolddir

ローテーションした旧ログをディレクトリに移動します。移動先は元と同じデバイス上で指定します。元のログに対する相対指定も有効です。

■mail address/nomail

旧ログファイルを address に送信します。どの段階のログを送るかは maillast などのオプションで決まります。

■maillast

世代が終わって破棄されるログをメールします。

■mailfirst

初代ローテーションログをメールします。

■daily/weekly/monthly

ログローテーションを日毎/週毎/月毎に行います。デフォルトは daily。たとえば weekly なら、毎日実行したとしても、週に 1 回だけローテーションが行われます。

■size サイズ [K/M]

ログのサイズがサイズバイトを超えていればローテーションを行います。この条件は daily,weekly などの条件より優先されます。キロバイト（K）、メガバイト（M）での指定もできます。

■ifempty/notifempty

元のログファイルが空でもローテーションを行う、あるいは行いません。

■missingok/nomissingok

指定のログファイルが存在しなかったとしてもエラーを出さずに処理を続行する、あるいはエラーを出力します。

■firstaction

ローテーションを行う前にスクリプトを実行します。prerotete よりも前に実行される個別定義内でのみ指定可能です。

■prerotate

ローテーションを行う前にスクリプトを実行します。firstaction の後に実行されます。個別定義内でのみ指定できます。

■postrotate

ローテーションが行われた後にスクリプトを実行します。lastaction より前に実行されます。個別定義内でのみ指定できます。

■lastaction

ローテーションが行われた後（よりも後）にスクリプトを実行します。postrotate の後に実行されます。個別定義内でのみ指定できます。

■sharedscripts

ローテーションするログが複数あった場合に、prerotate、postrotate のスクリプトを一度だけ実行します。

■nosharedscripts

ローテーションするログが複数あった場合に、prerotate、postrotate のスクリプトを各ログファイル毎に実行します。

■include ファイル（ディレクトリ）

include の記述のある位置に別の設定ファイルを読み込みます。ディレクトリを指定した場合、そのディレクトリ内から、ディレクトリおよび名前付きパイプ以外の通常ファイルがアルファベット順に読み込まれます。

■tabooext [+] 拡張子 [, 拡張子,...]

include でディレクトリを指定した場合に読み込むファイルから除外するファイルの拡張子を指定します。デフォルトで「.rpmorig」「.rpmsave」「,v」「.swp」「.rpmnew」「~」「.cfsaved」「.rhn-cfg-tmp-*」が指定されています。+ を指定するとデフォルト指定に対して追加で拡張子を指定できます。+ を指定しないとデフォルト指定を破棄して新規に拡張子を指定します。

4.1.15 ログローテート設定ファイルの確認

/etc/logrotate.d/httpd を参考に、ローテートの設定を確認します。

```
# cat /etc/logrotate.d/httpd
/var/log/httpd/*log {
    missingok
    notifempty
    sharedscripts
```

```
    delaycompress
    postrotate
        /sbin/service httpd reload > /dev/null 2>/dev/null || true
    endscript
}
```

この例では、以下の通りログローテーションの処理が行われます。

対象となるログファイルは/var/log/httpd ディレクトリ内の、ファイル名が log で終わるすべてのログファイルです。デフォルトでは access_log、error_log というファイル名のログファイルが作成されています。

- 1 行目の missingok でログファイルが実在しなかったとしてもエラーを出さずに処理を続行します。
- 2 行目の notifempty で元のログファイルが空ならばローテーションしません。
- 3 行目の sharedscripts で prerotate,postrotate のスクリプトを一度だけ実行します。
- 4 行目の delaycompress で圧縮処理を次のローテーションまで遅らせます。
- 5 行目の “postrotate” から “endscript” までが、ローテーションが行われた後に実行されるスクリプトです。service コマンドを実行して httpd サービスを reload することで、新しいログファイルが生成されます。

4.2 ネットワークツールを使ったトラブルシューティング

サーバに接続できないなどネットワークに起因する問題が発生した場合、基本的な原因の調査を行うためのツールとして、以下のようなネットワークツールを使用します。

- ping
- traceroute
- netstat
- tcpdump
- Wireshark

これらのツールを使用した、トラブルシューティングについて解説します。一般的には、外部からサービスへの接続ができなくなった場合には、以下のような手順で原因の調査を行います。

1. ログの確認
2. ping コマンドによる IP 通信の確認
3. telnet コマンドによる TCP 通信の確認
4. netstat コマンドによるポートの状況の確認
5. 通信内容の確認

4.2.1 ping コマンドによる IP 通信の確認

ping コマンドを使って、サーバに対する通信が行えるかどうかを確認します。ping コマンドは ICMP を使った通信で IP 通信が可能か確認できます。サーバに対する ping に応答が無い場合、以下のような問題が考えられます。

■サーバ自身の問題

IP アドレスやデフォルトゲートウェイが適切に設定されていなかったり、iptables などのパケットフィルタリングで ICMP を通さない設定になっていることが考えられます。サーバのネットワーク設定を再度確認してみます。また、サーバ側から他のホストへ ping コマンドを実行して、応答があるか確認してみます。

■ネットワーク経路の問題

ネットワーク通信経路上にあるケーブルやスイッチ、ルーター、ファイアーウォールやロードバランサーなどのネットワーク機器に問題が無いかを確認します。ルーティングに問題があるかを確認するためには traceroute コマンドを使用しますが、traceroute コマンドは ICMP を使用しているため、途中のルーターで ICMP を通さない場合、すべての経路が確認できないことがあります。

4.2.2 telnet コマンドによる TCP 通信の確認

telnet コマンドは 2 番目の引数にポート番号を指定して、サーバのサービスに接続することができるので、TCP 通信が可能か確認できます。

```
telnet 接続先IPアドレス ポート番号
```

ただし、ディストリビューションによっては telnet コマンドがインストールされていないので、インストールする必要があります。

```
# yum install telnet
```

サービスに接続できない場合には、以下のような問題が考えられます。

■ネットワーク経路の問題

iptables やネットワーク経路上のファイアーウォールなどで、指定されたポートへの通信が許可されていない。iptables やファイアーウォールのポート許可設定を確認します。

■サーバ自身の問題

サービスが停止しており、指定されたポートを Listen していない。あるいは、ローカルループバックアドレス（127.0.0.1）のみ Listen しており、接続先に指定した IP アドレスにポートがバインドされていない。netstat コマンドや lsof コマンドなどを使用して、ポートの状態を確認します。

4.2.3 netstat でのポートの状況の確認

netstat コマンドを使って、サービスプロセスとポート番号、さらに IP アドレスとのバインドの状況が確認できます。

netstat コマンドに-p オプションを指定して実行します。

```
# netstat -anp | grep sshd
tcp        0      0 0.0.0.0:22      0.0.0.0:*   LISTEN   1493/sshd
```

この結果から、以下のことが分かります。

- sshd のプロセス ID が 1493 であること
- TCP ポート番号 22 番で LISTEN していること
- ポート番号 22 番がサーバのすべての IP アドレス（0.0.0.0:22）にバインドされていること
- 送信元制限を行っていないこと（0.0.0.0:*）

4.2.4 パケットキャプチャによる通信内容の確認

サーバとの接続が行えており、ログにも手がかりとなるエラーが無いが、サービスが正しく動作しないような場合には、通信パケットをキャプチャして、通信内容を確認します。パケットをキャプチャすることで、サーバとクライアントの間でどのような通信が行われているかを確認できます。パケットキャプチャのツールとしては、シンプルに機能する tcpdump コマンドと、GUI で操作できる Wireshark などがあります。

4.2.5 tcpdump コマンドを使ったパケットキャプチャ

tcpdump コマンドは、送受信しているパケットをキャプチャして、その情報を標準出力に出力するコマンドです。tcpdump コマンドはデフォルトでは全てのパケットの情報を出力するので、オプションで出力結果をフィルタリングして、必要な情報を得られるようにします。

例として、-i オプションでネットワークインターフェースを指定して、eth0 を通じて入ってくる通信のパケットを取得してみます。

サーバ上で tcpdump コマンドを実行します。結果をリダイレクトして、tcpdump.out ファイルに記録します。

```
# tcpdump -i eth0 > tcpdump.out
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
```

クライアントから SSH でサーバにログインし、ログアウトします。

サーバで Ctrl+C キーを入力して、tcpdump コマンドを終了します。

```
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
※^C※216 packets captured ※←Ctrl+Cキーを入力
216 packets received by filter
0 packets dropped by kernel
```

作成された tcpdump.out ファイルの内容を確認します。

```
# grep ssh tcpdump.out
13:17:06.041096 IP client.example.com.43880 > server.example.com.ssh: ※Flags [S]※,
    seq 4050960604, win 14600, options [mss 1460,sackOK,TS val 13231 ecr 0,nop,wscale
    6], length 0
13:17:06.041125 IP server.example.com.ssh > client.example.com.43880: ※Flags [S.]※,
    seq 3335753529, ※ack 4050960605※, win 14480, options [mss 1460,sackOK,TS val
    22019990 ecr 13231,nop,wscale 6], length 0
13:17:06.041240 IP client.example.com.43880 > server.example.com.ssh: ※Flags [.]※,
    ※ack 1※, win 229, options [nop,nop,TS val 13231 ecr 22019990], length 0
```

左から時間（マイクロ秒単位)、送信元 IP アドレス. ポート番号、通信の向きの矢印、宛先ホスト. ポート番号、フラグ（SYN)、シーケンス、ウィンドウ、オプション、最大セグメントサイズとなっています。

■1 行目

クライアントのポート 43880 からサーバのポート 22（ssh）に向けて SYN フラグの TCP パケットと送信して接続の要求

■2 行目

1 行目のパケットに対して、SYN+ACK フラグの TCP パケットを送信

■3 行目

ACK フラグの TCP パケットを送信して、TCP のスリーウェイハンドシェイクが完了

このように、サーバとクライアントの間の通信を確認できます。

4.2.6 Wireshark を使った確認

tcpdump の出力ファイルは少量のパケットを見る場合には充分ですが、大量のパケットを確認するには可読性が低いのが難点です。

GUI を持つパケットキャプチャリングソフトである Wireshark を使えば、パケットキャプチャリングを行ったパケットの中身を見たり、フィルタリング機能で必要なパケットのみに絞り込んでパケットを確認することができます。

Wireshark をインストールします。GUI 版をインストールするため、wireshark-gnome パッケージをインストールします。

```
# yum install wireshark-gnome
```

1. Wireshark を起動します。CentOS に GUI でログインし、端末から wireshark コマンドを実行するか、「アプリケーション」メニュー→「インターネット」→「Wireshark Network Analyzer」を起動します。

```
# wireshark &
```

2

2. キャプチャを行うデバイスを選びます。

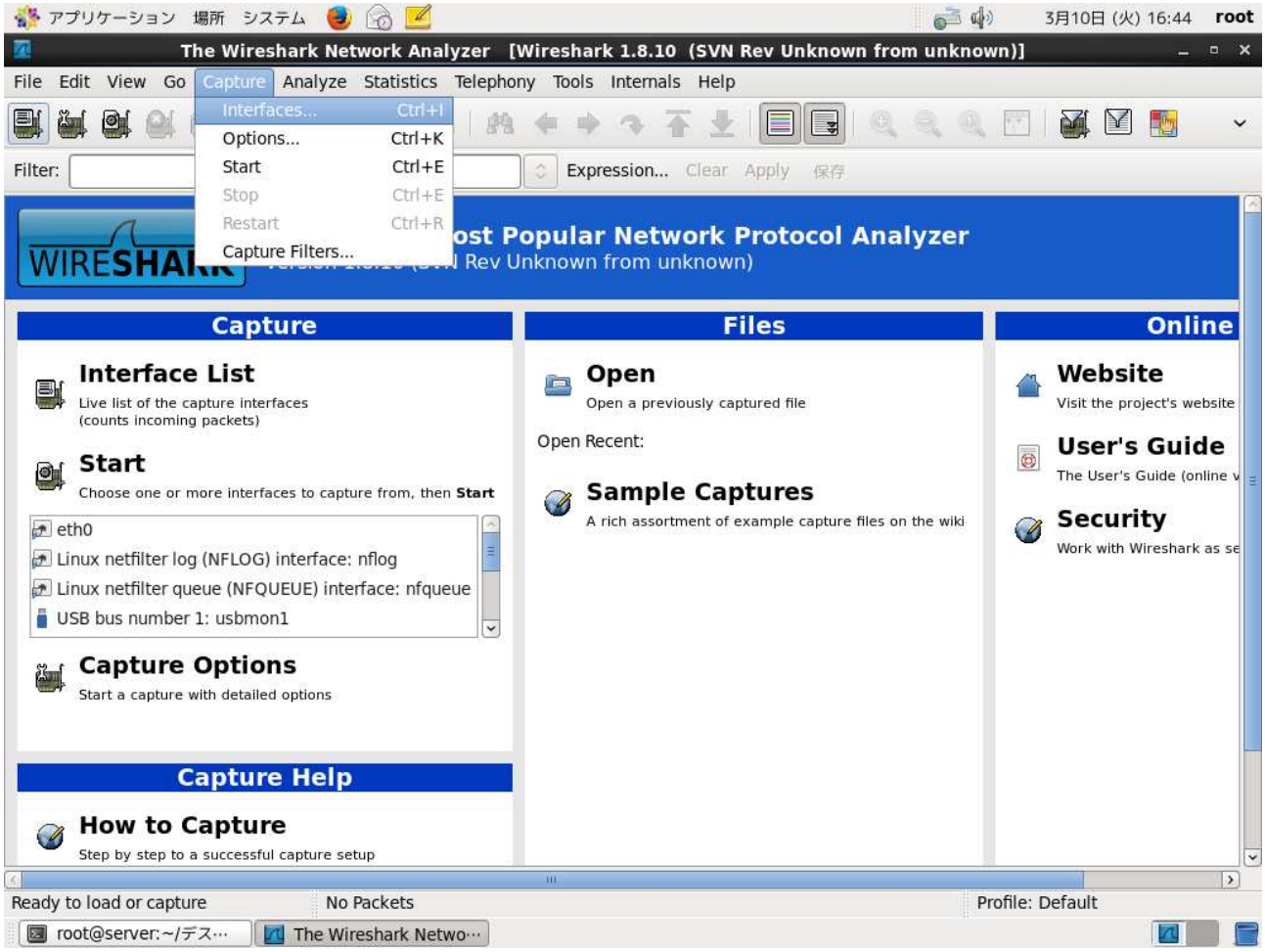


図 29　「Capture」メニュー→「Interfaces」を選択します

「Capture」メニュー→「Interfaces」を選択し、パケットキャプチャを行いたいデバイスを選びます。

3

3. パケットキャプチャを開始します。

外部との通信をパケットキャプチャするために eth0 を選びます。「Start」ボタンをクリックして、パケットキャプチャを開始します。

4

4. Web サーバにアクセスします。サーバと通信を行ってパケットキャプチャを行います。クライアントで Web ブラウザを起動し、サーバの Web サーバにアクセスします。

5

5. パケットキャプチャを停止します。「Capture」メニュー→「Stop」を選択し、パケットキャプチャを停止します。

6

6. 結果の絞り込みを行います。

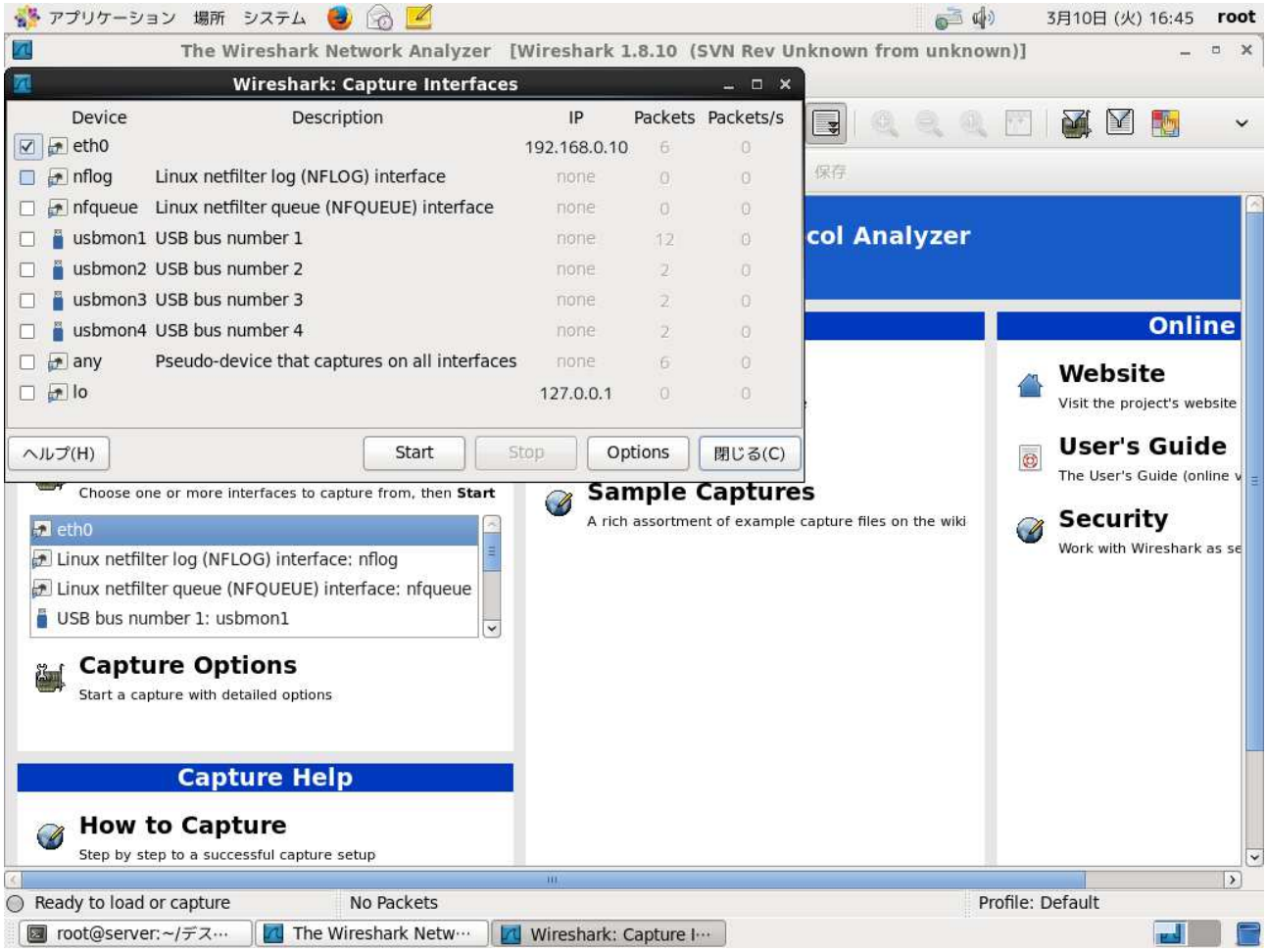


図 30　eth0 を選択します

「Filter:」のテキストボックスに「http」と入力して、Enter キーを押して絞り込みます。参照したいパケットを選択し、ウインドウ真ん中の詳細情報で「Hypertext Transfer Protocol」をダブルクリックして、HTTP 通信の内容を確認します。

4.3　ファイルシステム障害の修復

ファイルシステム障害が発生して OS が正常に起動しなくなった場合、起動ディスクである程度までシステム起動が可能ならばシングルユーザーモードで起動したり、起動ディスクでシステムを起動できない場合にはインストール用のメディアをレスキューモードで起動することで、ファイルシステムを修復できます。

4.3.1　シングルユーザモードでの起動

シングルユーザモードで Linux を起動すると、ランレベル 1 で起動するため各種サービスの起動が行われず、root ユーザだけがシステムにアクセスできる状態で起動します。たとえば、サービスの設定を間違えたためランレベル 3 やランレベル 5 で起動するとシステムに不具合が発生する場合には、シングルユーザーモードで起動して設定を修正します。

起動時に表示できる GRUB メニューで起動パラメーターを編集してシングルモードで起動します。

1. 起動時のデフォルトでは、設定された秒数（デフォルトでは 5 秒）が過ぎると自動的に起動しますが、何かキーを入力すると GRUB メニューが表示されます。
2. キーボードの e キーを押して起動パラメーターの編集モードに入り、kernel 行を選択してさらに e キーを押し、末尾に「single」（あるいは 1）とパラメーターを追記します。
3. Enter キーを押して編集モード画面に戻ります。
4. b キーを押してシングルユーザモード起動します。

5

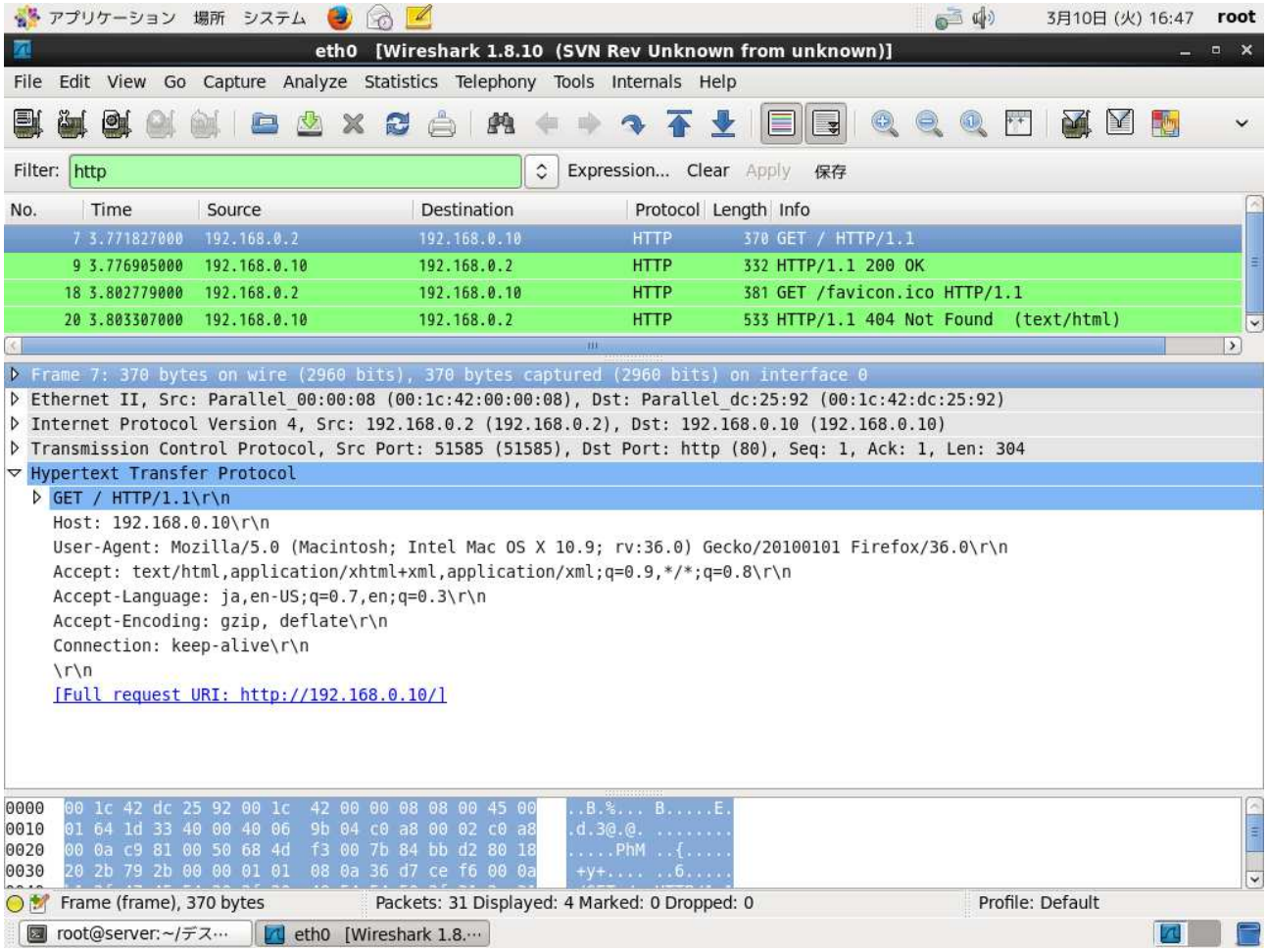


図 31　http で絞り込みを行います

5. シングルユーザモードで起動すると、パスワード無しで root ユーザとしてログインしている状態となります。必要に応じて fsck コマンドでファイルシステムを修復したり、設定ファイルを修正するなどしてトラブルの解決を行います。
6. シェルから exit すると、デフォルトのランレベルに移行します。

4.3.2　インストール DVD メディアからレスキューモードで起動

起動ディスクのファイルシステムに障害が発生して、正常に OS が起動できなくなってしまった場合には、インストール DVD メディアからレスキューモードで起動し、ファイルシステムの修復を行います。

1. CentOS のインストール DVD メディアでシステムを起動します。BIOS で起動デバイスの順番を変更するなどして、DVD ドライブから起動するようにします。
2. 起動メニューから「Rescue installed system」を選択します。

＃ 3

1. Language、キーボードレイアウト、修復作業中にネットワークを使用するかを選択します。

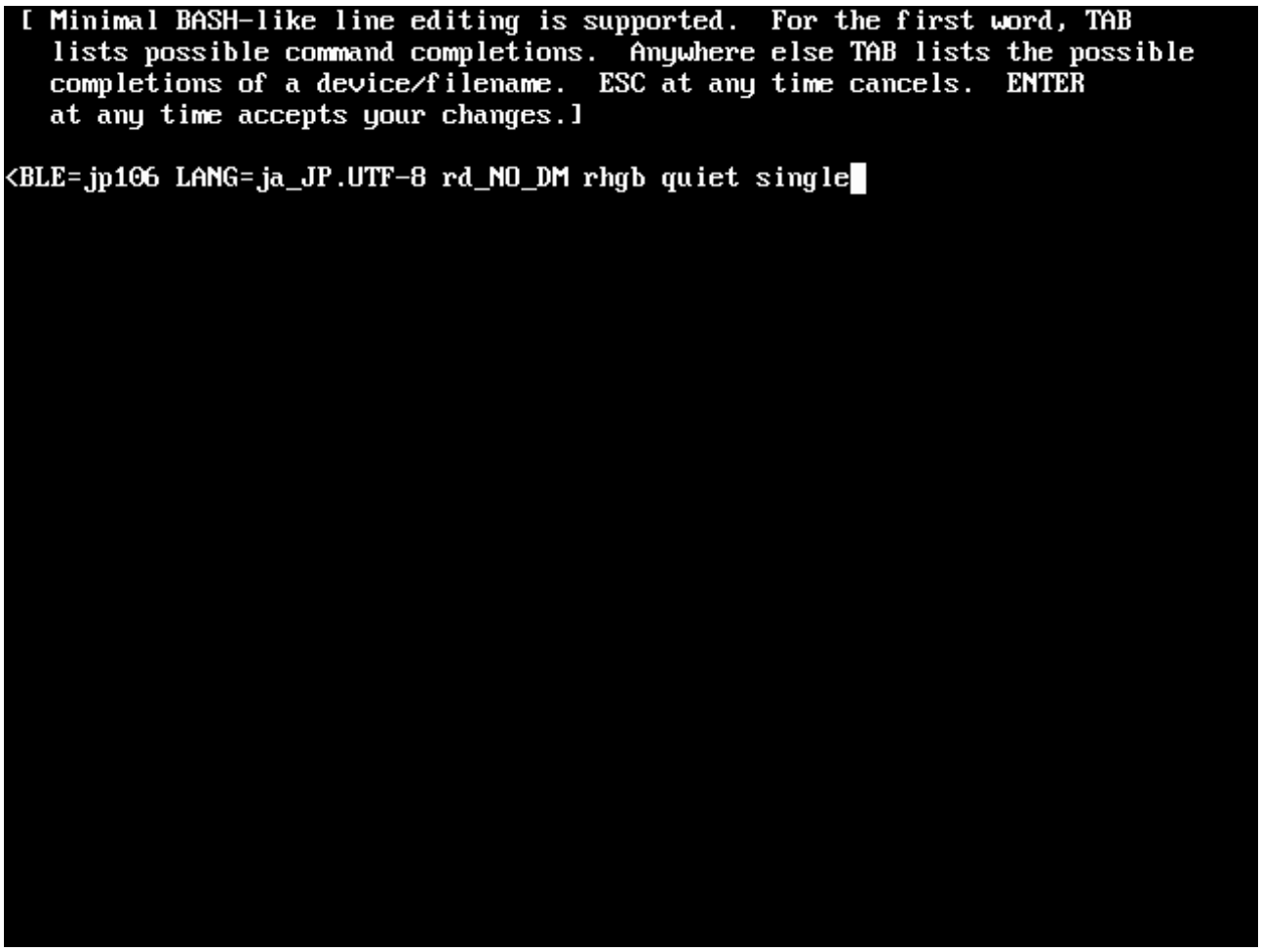
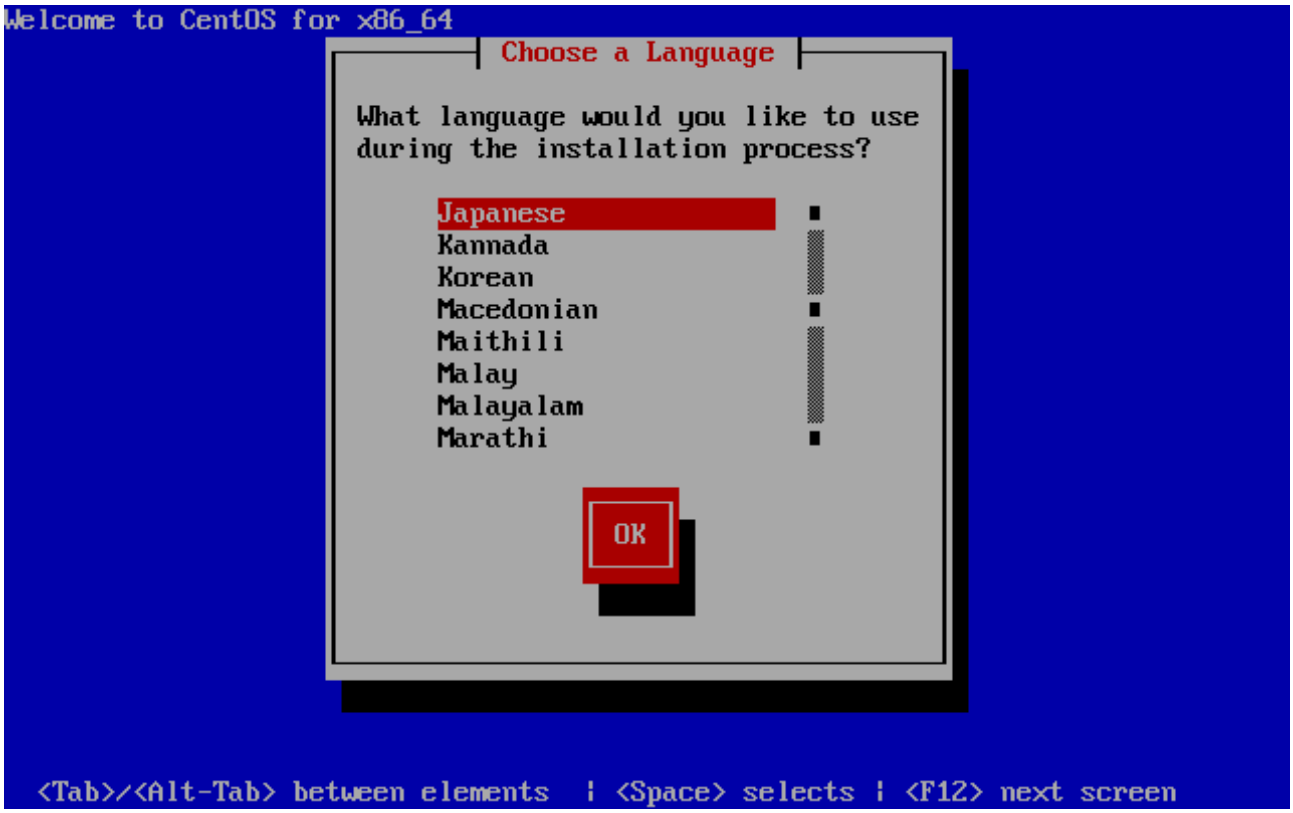


図 32　カーネルパラメータでシングルユーザーモード起動を設定します



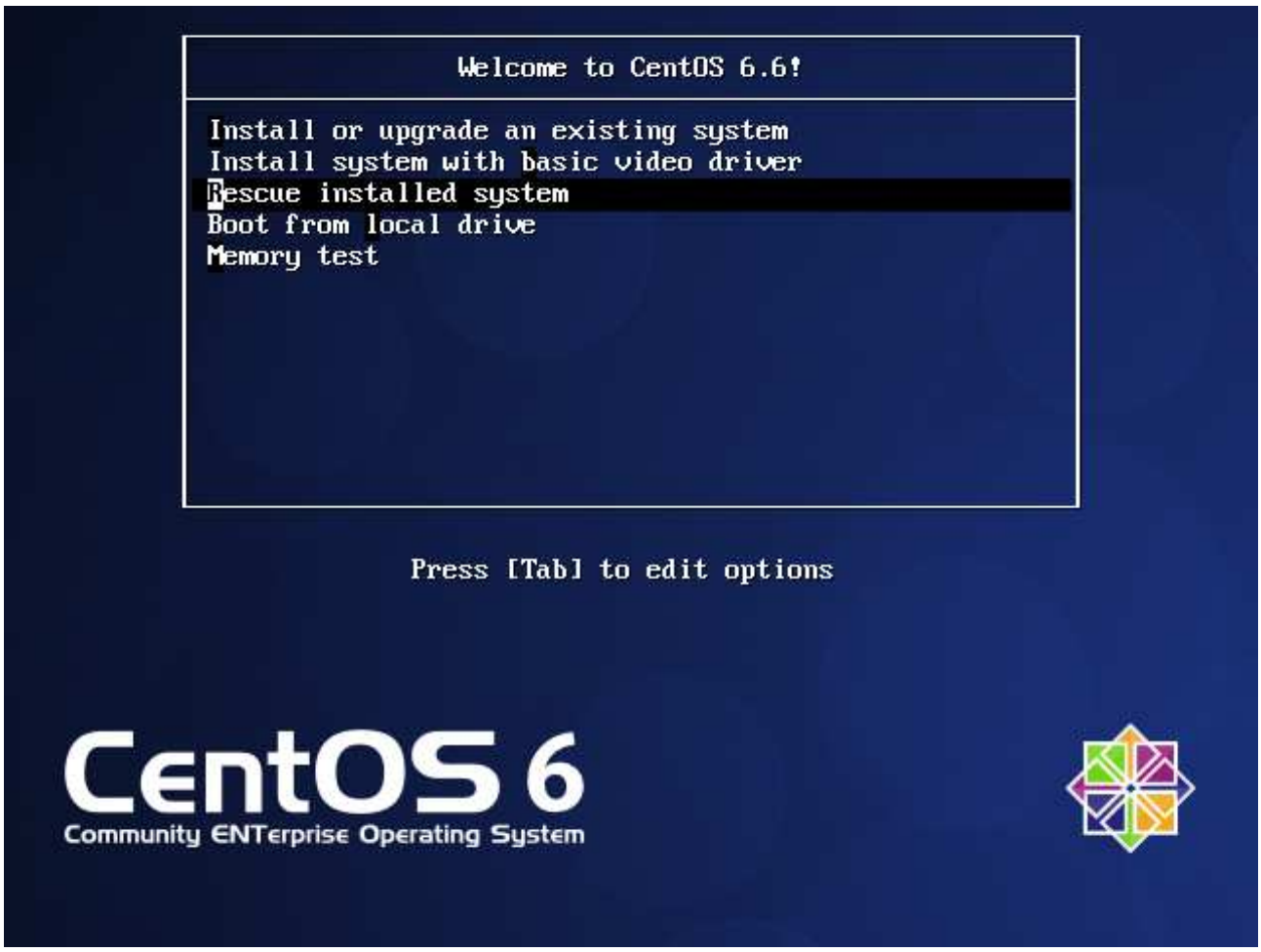
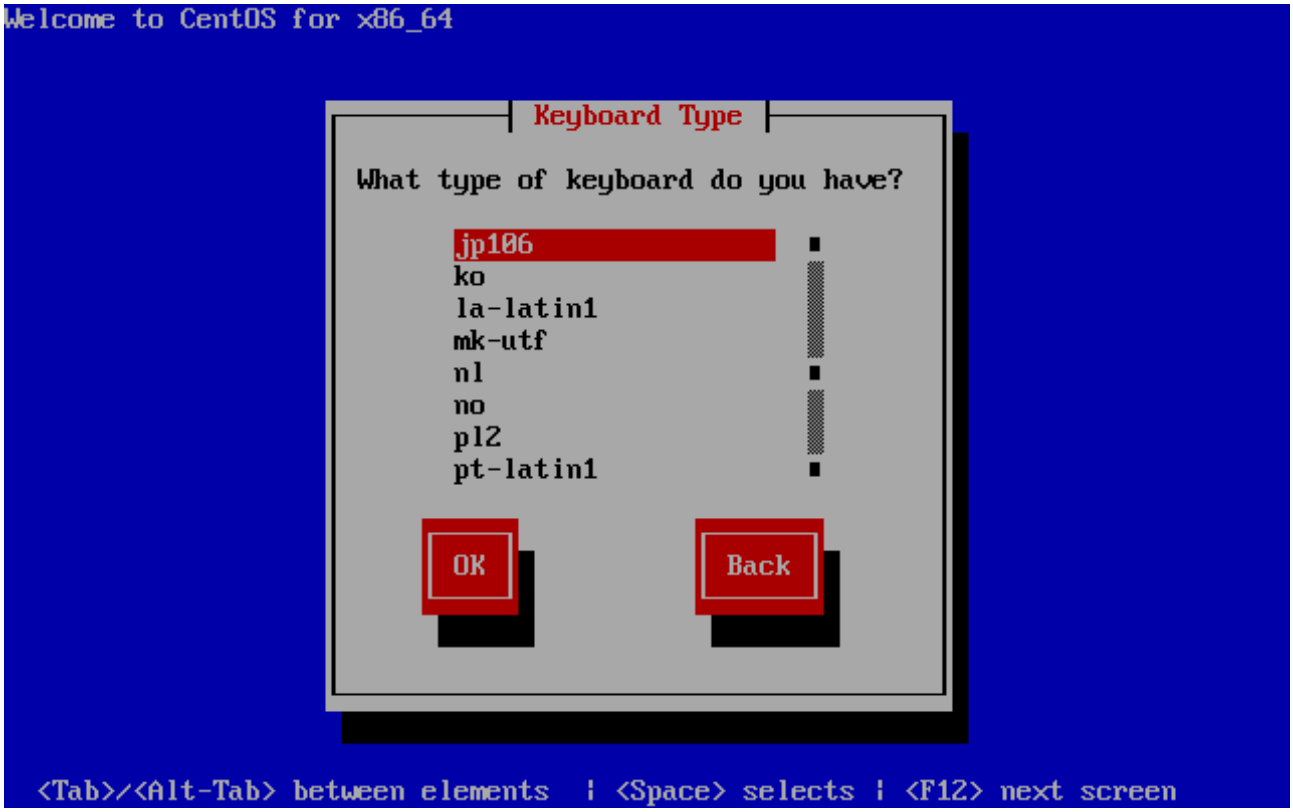
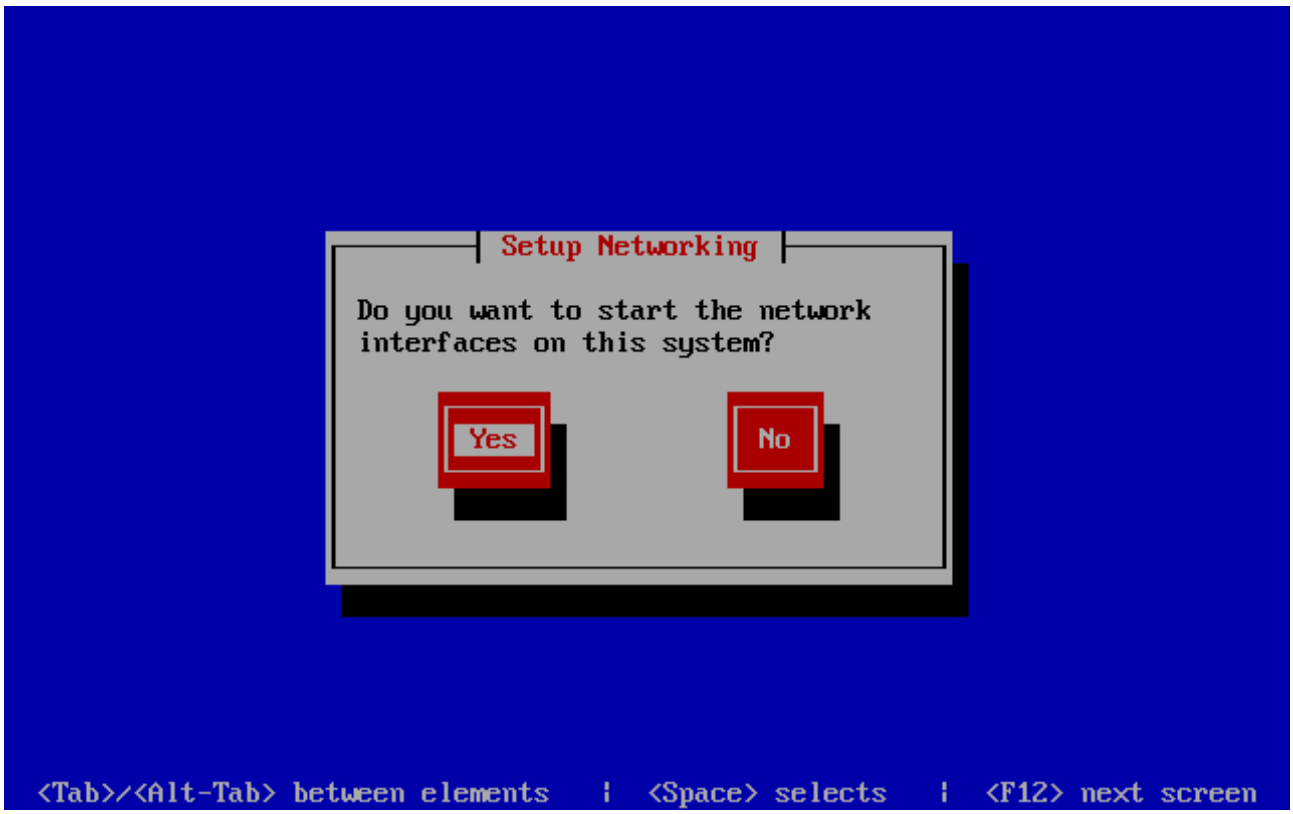

Welcome to CentOS 6.6!
Install or upgrade an existing system
Install system with basic video driver
Rescue installed system
Boot from local drive
Memory test
Press [Tab] to edit options
CentOS 6
Community ENTerprise Operating System

図 33　起動メニュー


Welcome to CentOS for x86_64
Keyboard Type
What type of keyboard do you have?
jp106
ko
la-latin1
mk-utf
nl
no
pl2
pt-latin1
OK
Back
<Tab>/<Alt-Tab> between elements ¦ <Space> selects ¦ <F12> next screen



4

1. ハードディスクを検索し、/mnt/sysimage 以下にマウントする旨の説明が表示されます。「Read-Only」を選ぶと、ハードディスクが読み取り専用でマウントされます。修復を行うため、「Continue」を選択します。

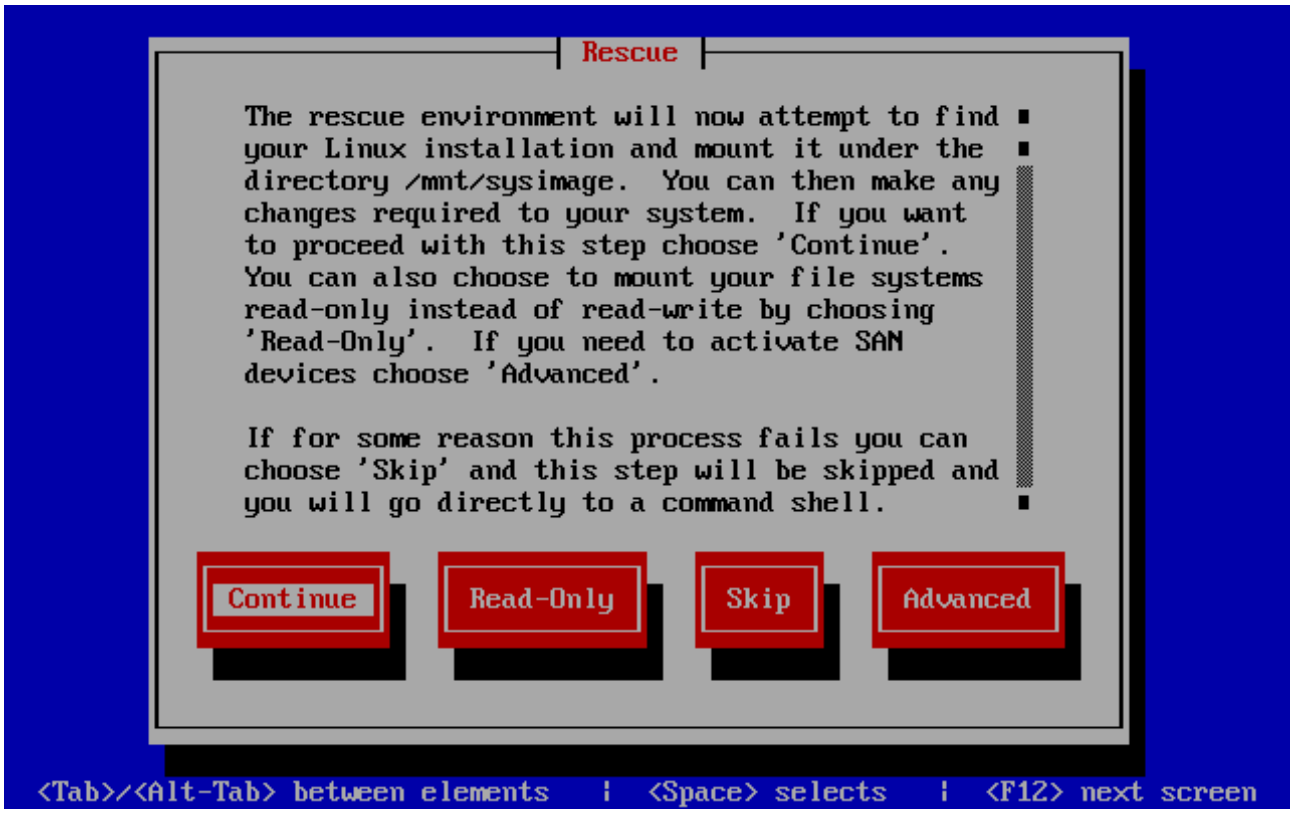


図 34　Continue を選択します

5

1. ハードディスクを検索し、/mnt/sysimage 以下にマウントできた旨が表示されます。

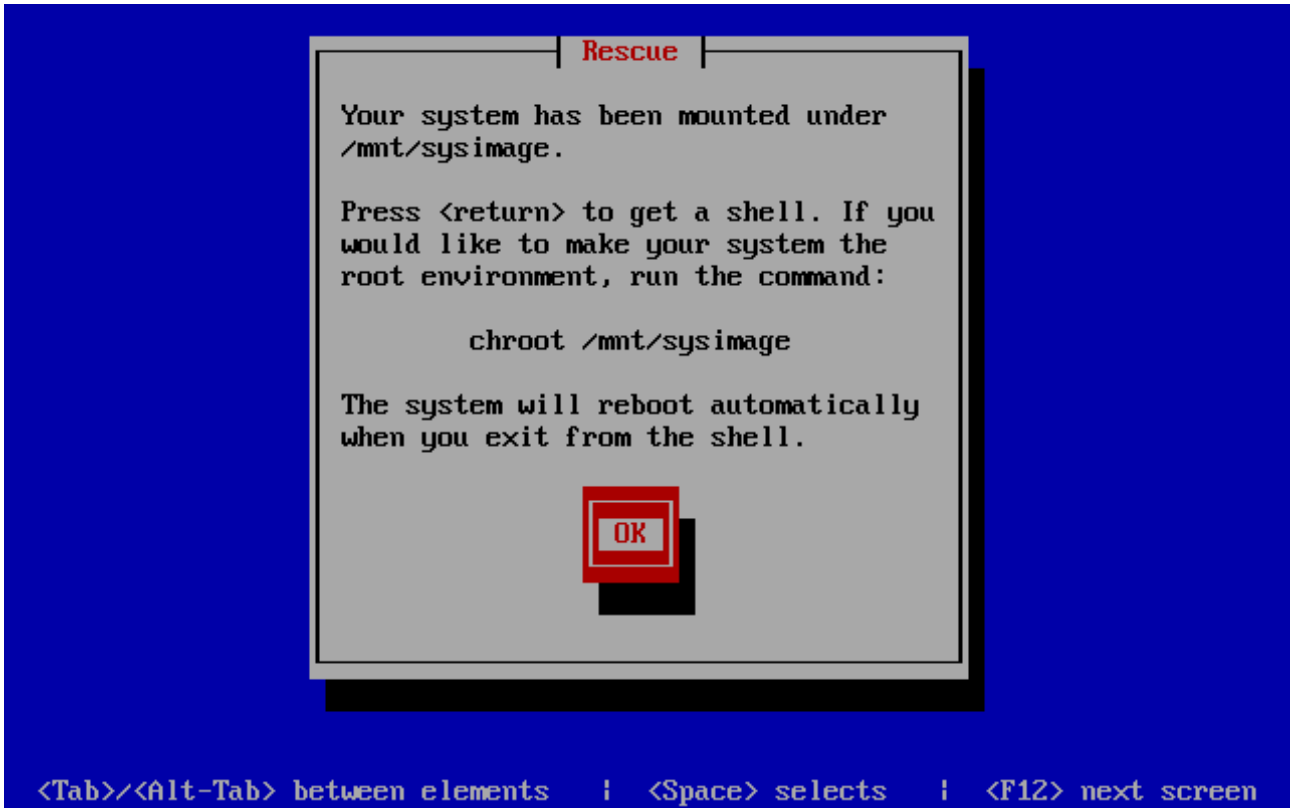


図 35　/mnt/sysimage にハードディスクがマウントされました

＃ 6

1. 実行する作業を選択します。「shell」を選ぶとシェルが起動します。「fakd」を選ぶと First Aid Kit が実行されてシステムの検査が行えます。「reboot」を選ぶとシステムを再起動します。「shell」を選択します。

＃ 7

1. bash が起動します。/mnt/sysimage 以下に、ハードディスクのルートパーティションがマウントされていることを確認します。

＃ 8

1. fsck コマンドなどを利用して、ファイルシステムの修復作業を行います。修復作業が終了したら、exit でシェルを終了します。作業の選択画面に戻ります。
2. 「reboot」を選択して、システムを再起動します。インストール DVD メディアは DVD ドライブから取り出しておきます。

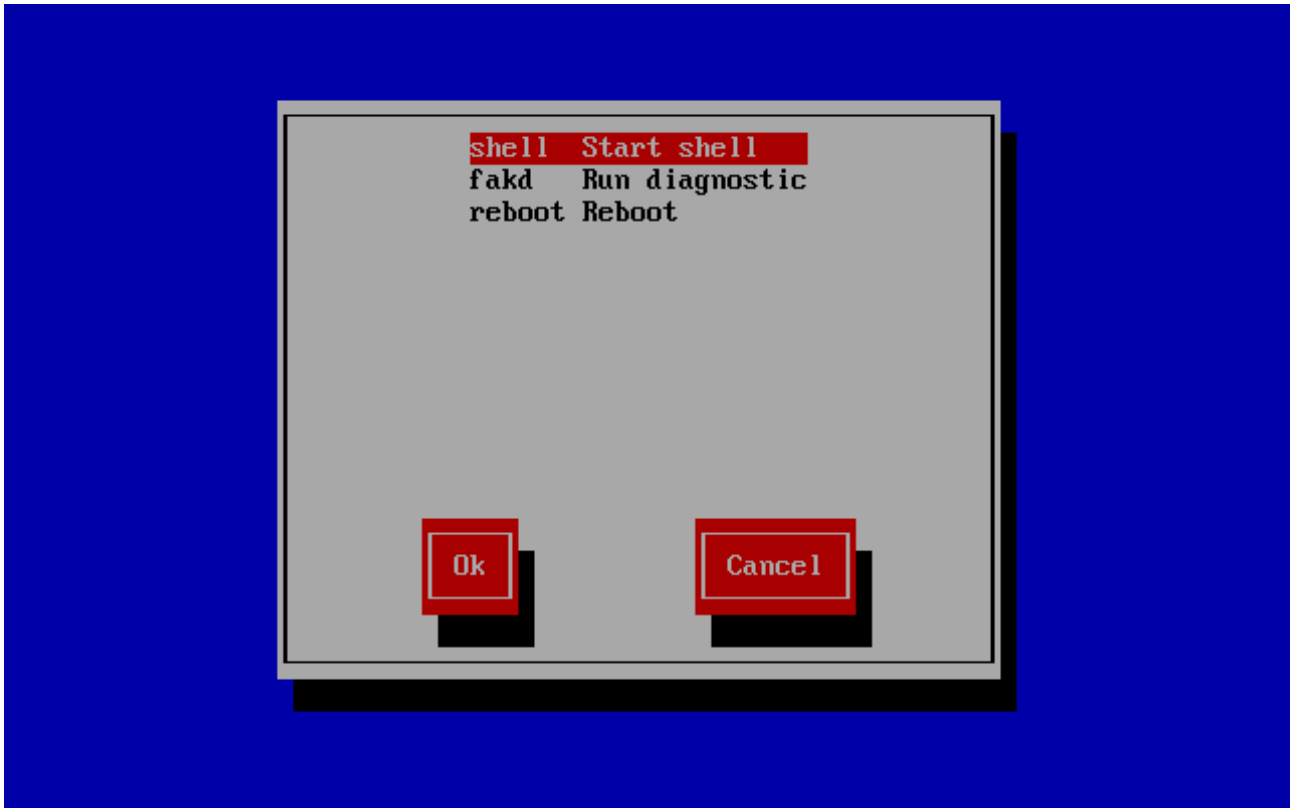
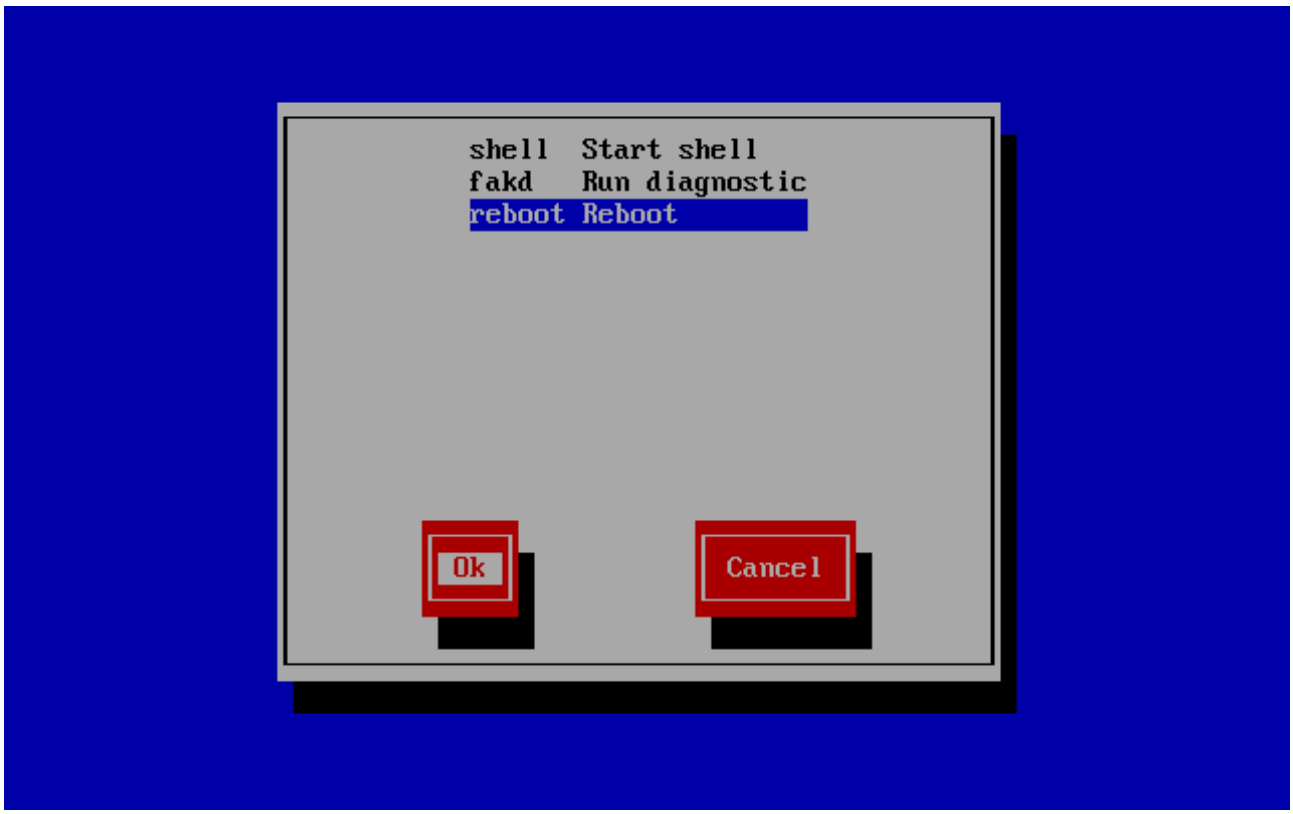


図 36　shell を選択します

```
Starting shell...
bash-4.1# mount
rootfs on / type rootfs (rw,relatime)
/proc on /proc type proc (rw,relatime)
/dev on /dev type tmpfs (rw,seclabel,relatime)
/dev/pts on /dev/pts type devpts (rw,seclabel,relatime,gid=5,mode=620,ptmxmode=0
00)
/sys on /sys type sysfs (rw,seclabel,relatime)
none on /tmp type tmpfs (rw,seclabel,relatime,size=508188k)
/dev/loop0 on /mnt/runtime type squashfs (ro,relatime)
/selinux on /selinux type selinuxfs (rw,relatime)
/dev/mapper/vg_server-lv_root on /mnt/sysimage type ext4 (rw,seclabel,relatime,b
arrier=1,data=ordered)
/dev/sda1 on /mnt/sysimage/boot type ext4 (rw,seclabel,relatime,barrier=1,data=o
rdered)
/dev on /mnt/sysimage/dev type tmpfs (rw,seclabel,relatime)
/dev/devpts on /mnt/sysimage/dev/pts type devpts (rw,seclabel,relatime,gid=5,mod
e=620,ptmxmode=000)
/dev/tmpfs on /mnt/sysimage/dev/shm type tmpfs (rw,seclabel,relatime)
/dev/mapper/vg_server-lv_home on /mnt/sysimage/home type ext4 (rw,seclabel,relat
ime,barrier=1,data=ordered)
/dev/proc on /mnt/sysimage/proc type proc (rw,relatime)
/dev/sysfs on /mnt/sysimage/sys type sysfs (rw,seclabel,relatime)
/selinux on /mnt/sysimage/selinux type selinuxfs (rw,relatime)
bash-4.1# _
```

図 37　シェルが起動します



CentOS 7 への移行 ## CentOS 7 への移行 本教科書では CentOS 6 を使って解説をしてきましたが、現在では新しいバージョンである CentOS 7 もリリースされています。

CentOS 7 へのバージョンアップ後も基本的な運用管理の知識は大きくは変わりませんが、CentOS 7 で大きく変更になった以下の点について解説します。

- SysV init から systemd への移行
- journald によるログ記録
- firewalld によるパケットフィルタリング

また、変更されたわけではありませんが、ネットワークは NetworkManager が CentOS 6 と変わらず標準なので、CUI の NetworkManager 用のツールである nmtui を紹介します。

4.4　SysV init から systemd への移行

CentOS 7 から、これまでのサービス管理である SysV init、または Upstart から Linux 向けの新しいサービス管理マネージャーである「systemd」に置き換えられました。/etc/rc.d ディレクトリ以下のサービス起動スクリプトを使う方式から改められました。

簡単に systemd でのサービス管理について解説します。

4.4.1　ユニットでの管理

従来ランレベルやサービスと呼ばれていた仕組みは、systemd では「ユニット」として再定義されました。ユニットには、「ターゲット」（ランレベルに相当）ユニットや「サービス」ユニットがあり、それぞれのユニットは依存関係の定義ができるようになっています。

依存関係とは、たとえば「このサービスを実行するにはあらかじめこのサービスが実行されていなければならない」という関係です。従来の SysV init ではサービスには依存関係が無かったため、サービスを順番に実行するという方法で依存関係を解消していました。しかし、実行の順番を保証するために 1 つずつ実行する「順列処理」となるため、システム起動が遅くなるという難点がありました。

systemd では依存関係にないサービスを「並列処理」で実行するため、高速にシステムを起動できるという利点があります。

主なユニットの種類は以下の通りです。

ユニット	役割
service	従来のサービスと同様
target	サービスを取りまとめるためのユニット
mount	マウントポイント
swap	スワップ領域
device	デバイス

4.4.2 サービスの操作

systemd では、サービスの起動や停止を行うのに systemctl コマンドを使用します。これは従来の service コマンドに相当します。

Web サービスの起動や停止、再起動、そして状態の確認を行うには、以下の systemctl コマンドを使用します。

■サービスの起動

systemctl start コマンドで、サービスを起動します。

```
# systemctl start httpd
```

■サービスのステータス確認

systemctl status コマンドで、サービスのステータスを確認できます。

systemd では、サービスのプロセスを「コントロールグループ」（cgroup）という Linux カーネルの仕組みで実行するように変わりました。cgroup を使うことで、CPU やメモリなどのリソースを柔軟に割り当てることができる利点があります。

```
# systemctl status httpd
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since 水 2015-01-28 15:23:50 JST; 33s ago
 Main PID: 2926 (httpd)
   Status: "Total requests: 0; Current requests/sec: 0; Current traffic:   0 B/sec"
   CGroup: /system.slice/httpd.service
           ├─2926 /usr/sbin/httpd -DFOREGROUND
           ├─2927 /usr/sbin/httpd -DFOREGROUND
           ├─2928 /usr/sbin/httpd -DFOREGROUND
           ├─2929 /usr/sbin/httpd -DFOREGROUND
           ├─2930 /usr/sbin/httpd -DFOREGROUND
           └─2931 /usr/sbin/httpd -DFOREGROUND

 1月 28 15:23:50 centos7.example.com httpd[2926]: AH00557: httpd: apr_sockad...
 1月 28 15:23:50 centos7.example.com httpd[2926]: AH00558: httpd: Could not ...
 1月 28 15:23:50 centos7.example.com systemd[1]: Started The Apache HTTP Ser...
Hint: Some lines were ellipsized, use -l to show in full.
```

■サービスの再起動

systemctl restart コマンドで、サービスを再起動します。

```
# systemctl restart httpd
# systemctl status httpd
httpd.service - The Apache HTTP Server
```

```
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since 水 2015-01-28 15:24:40 JST; 2s ago
  Process: 2945 ExecStop=/bin/kill -WINCH ${MAINPID} (code=exited, status=0/SUCCESS)
 Main PID: 2950 (httpd)
 (略)
```

■サービスの停止

systemctl stop コマンドで、サービスを停止します。

```
# systemctl stop httpd
# systemctl status httpd
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: inactive (dead)
```

4.4.3 ユニット一覧の取得

systemd で管理されているユニットの一覧を取得するには、systemctl list-unit-files コマンドを実行します。

```
# systemctl list-unit-files
```

すべての種類のユニットが表示されてしまうので、ユニットの種類を絞り込むには-t オプションを付与して実行します。

たとえば、service ユニットだけを表示するには以下の systemctl コマンドを実行します。これは従来の chkconfig –list コマンドに相当します。

```
# systemctl list-unit-files -t service
```

表示されるステータス（STATE）の意味は以下の通りです。

ステータス	意味
enabled	システム起動時に実行される
disabled	システム起動時に実行されない
static	システム起動時の実行の有無は設定できない

4.4.4 現在のユニットの状況を確認

現在のユニットの状況を確認するには、systemctl list-units コマンドを実行します。systemctl コマンドのデフォルトはこのサブコマンドの指定になっています。

以下の例は同じ結果を返します。

```
# systemctl list-units
# systemctl
```

-t オプションを使って、service ユニットだけに絞り込むこともできます。

```
# systemctl -t service
UNIT                        LOAD   ACTIVE SUB     DESCRIPTION
abrt-ccpp.service           loaded active exited  Install ABRT coredump hook
abrt-oops.service           loaded active running ABRT kernel log watcher
abrt-xorg.service           loaded active running ABRT Xorg log watcher
abrtd.service               loaded active running ABRT Automated Bug Reporting
```

```
alsa-state.service           loaded active running Manage Sound Card State (rest
atd.service                  loaded active running Job spooling tools
 (略)
kdump.service                loaded failed failed  Crash recovery kernel arming
 (略)
```

表示の意味は以下の通りです。

項目	意味
UNIT	ユニット名
LOAD	systemd への設定の読み込み状況
ACTIVE	実行状態の概要。active か inactive で表される
SUB	実行状態の詳細。running（実行中）や exited（実行したが終了した）などで表される。
DESCRIPTION	ユニットの説明

デフォルトでは、項目 ACTIVE の実行状態が active になっているもののみが表示されています。inactive のユニットも表示するには–all オプションを付与して実行します。

項目 LOAD は、systemctl mask コマンドで無効化されると masked に変わります。詳細は後述します。

項目 ACTIVE が failed になっていると、何らかの原因で起動失敗しているということになります。上記の例では、kdump（カーネルダンプ）サービスの起動に失敗しています。

4.4.5 デバイス一覧の確認

-t device オプションを付与して、デバイス一覧を表示します。

```
# systemctl list-units -t device
UNIT

     LOAD   ACTIVE SUB     DESCRIPTION
sys-devices-pci0000:00-0000:00:05.0-virtio0-net-eth0.device
                                 loaded active plugged Virtio network device
sys-devices-pci0000:00-0000:00:1f.2-ata3-host2-target2:0:0-2:0:0:0-block-sda-sda1.device
     loaded active plugged CentOS_7-0_SSD
 (略)
```

4.4.6 マウント状況の確認

-t mount オプションを付与して、マウントの状況一覧を表示します。

```
# systemctl list-units -t mount
UNIT                         LOAD   ACTIVE SUB     DESCRIPTION
-.mount                      loaded active mounted /
boot.mount                   loaded active mounted /boot
dev-hugepages.mount          loaded active mounted Huge Pages File System
dev-mqueue.mount             loaded active mounted POSIX Message Queue File Syst
home.mount                   loaded active mounted /home
 (略)
```

4.4.7 スワップ状況の確認

-t swap オプションを付与して、スワップの状況一覧を表示します。

```
# systemctl list-units -t swap
UNIT             LOAD   ACTIVE SUB    DESCRIPTION
dev-dm\x2d0.swap loaded active active /dev/dm-0
 （略）
```

4.4.8 サービスの自動起動の設定

システム起動時にサービスを自動起動するには、systemctl enable コマンドを実行します。これは従来の chkconfig コマンドに相当します。

例として、Web サービスをシステム起動時に自動起動するように設定します。/usr/lib/systemd/system/httpd.service が Web サービスの起動スクリプトです。systemctl enable コマンドを実行すると、/etc/systemd/system/multi-user.target.wants ディレクトリにシンボリックリンクが作成されます。

この動作は、multi-user.target ターゲットユニットが呼び出された時に、シンボリックリンクの起動スクリプトが実行されるように設定しています。SysV init における/etc/init.d ディレクトリ内のサービス起動スクリプトと/etc/rc.d ディレクトリ内にランレベル毎に作成されるシンボリックリンクの関係に相当します。

```
# systemctl enable httpd
ln -s '/usr/lib/systemd/system/httpd.service'
     '/etc/systemd/system/multi-user.target.wants/httpd.service'
```

システム起動時の自動起動を行わないようにするには、systemctl disable コマンドを実行します。作成されたシンボリックリンクが削除され、起動スクリプトは呼び出されなくなります。

```
# systemctl disable httpd
rm '/etc/systemd/system/multi-user.target.wants/httpd.service'
```

4.4.9 サービスの systemd からの除外

systemctl mask コマンドを実行すると、指定したサービスが systemd の管理から除外され、手動での起動も行えなくなります。

動作としては、/etc/systemd/system/httpd.service が/dev/null へのシンボリックリンクとして作成され、この起動スクリプトが呼び出されても何も行われなくなります。

Web サービスを systemd から除外します。

```
# systemctl mask httpd
ln -s '/dev/null' '/etc/systemd/system/httpd.service'
# systemctl start httpd
Failed to issue method call: Unit httpd.service is masked.
```

systemctl is-enabled コマンドで、サービスの状態が確認できます。httpd サービスの状態は masked となっています。

```
# systemctl is-enabled httpd
masked
```

systemctl unmask コマンドを実行すると、シンボリックリンクが削除されて、指定したサービスが systemd で管理されるようになります。httpd サービスの状態は disabled になります。

```
# systemctl unmask httpd
rm '/etc/systemd/system/httpd.service'
# systemctl is-enabled httpd
disabled
```

4.4.10 systemd のサービスに関連するディレクトリとシステム起動の仕組み

systemd が内部的にどのような仕組みになっているのか、関連するディレクトリを解説します。

systemctl enable コマンドの動作を見ても分かる通り、systemd の仕組みにおいて、関連するディレクトリは以下の 2 つです。

■/usr/lib/systemd/system ディレクトリ

サービス起動スクリプトが格納されています。/etc/rc.d/init.d ディレクトリに相当します。

■/etc/systemd/system ディレクトリ

サービス起動スクリプトに対するシンボリックリンクが配置されます。/etc/rc.d ディレクトリに相当します。

システム起動時の systemd の動作は、/etc/systemd/system ディレクトリ以下のサブディレクトリ内に作成されたサービス起動スクリプトへのシンボリックリンクが順次実行されてサービスが起動されます。シンボリックリンクの作成される場所は、役割別のターゲットユニット毎にディレクトリが分けられています。

ターゲット毎のディレクトリとその役割、実行の順番は以下の通りです。

■1. /etc/systemd/system/sysinit.target.wants/

システムの初期に実行されるスクリプトです。rc.sysinit スクリプトに相当します。

■2. /etc/systemd/system/basic.target.wants/

システム共通に実行されるスクリプトです。

■3. /etc/systemd/system/multi-user.target.wants/

従来のランレベル 3（CUI）に相当します。

■4. /etc/systemd/system/graphical.target.wants/

従来のランレベル 5（GUI）に相当します。

SysV init ではランレベル 3 とランレベル 5 は別々の扱いでしたが、systemd では multi-user.target を実行後に graphical.target が実行されるようになっています。

どこまで実行するかは、次に説明するデフォルトターゲットの設定によって決められています。

4.4.11 デフォルトターゲットの変更

systemd ではランレベルではなく、サービス起動スクリプトを順番に実行していき、デフォルトターゲットで指定されたターゲットまで実行します。デフォルトターゲットを変更することで、CUI 起動をするか、GUI 起動にするかを選択できます。

デフォルトターゲットの変更は、systemctl set-default コマンドを実行します。これは、SysV init の設定ファイル/etc/inittab で指定している起動時ランレベル（initdefault）の変更に相当します。

■デフォルトターゲットの確認

systemctl get-default コマンドで、現在のデフォルトターゲットを確認します。

```
# systemctl get-default
graphical.target
```

■デフォルトターゲットを CUI に変更

デフォルトターゲットを multi-user.target に変更し、再起動します。CUI で起動してくることを確認します。

```
# systemctl set-default multi-user.target
# reboot
```

■デフォルトターゲットを GUI に変更

GUI での起動に戻すには、以下の systemctl set-default コマンドを実行します。

```
# systemctl set-default graphical.target
# reboot
```

4.4.12 現在のターゲットの一時的な変更

systemd での現在のターゲットを一時的に変更するには、systemctl isolate コマンドを実行します。これは、SysV init のランレベル変更（telinit コマンド）に相当します。

GUI から CUI に変更します。GUI ログインしている場合、ログアウトします。

```
# systemctl isolate multi-user.target
```

CUI から GUI に変更します。

```
# systemctl isolate graphical.target
```

4.5 journald によるログ記録

systemd にはサービスなどからのログを記録する journald が用意されており、syslog とは別にログが記録されています。

4.5.1 journald のログの確認

journald のログを確認するには、journalctl コマンドを実行します。オプションを付与しないで実行すると、すべてのログが表示されます。

以下の例では、dmesg コマンドで確認できる Linux カーネル起動時のログが記録されているのが分かります。

```
# journalctl
-- Logs begin at 水 2015-01-28 17:29:04 JST, end at 水 2015-01-28 17:29:38 JST.
 1月 28 17:29:04 centos7.example.com systemd-journal[149]: Runtime journal is us
 1月 28 17:29:04 centos7.example.com systemd-journal[149]: Runtime journal is us
 (略)
```

特定のサービスのログに絞るには、-u オプションを付与して実行します。

以下の例では、httpd サービス起動時のログが確認できます。

```
# journalctl -u httpd
-- Logs begin at 水 2015-01-28 17:29:04 JST, end at 水 2015-01-28 17:31:34 JST.
 1月 28 17:31:28 centos7.example.com systemd[1]: Starting The Apache HTTP Server
 1月 28 17:31:34 centos7.example.com httpd[2232]: AH00557: httpd: apr_sockaddr_i
 1月 28 17:31:34 centos7.example.com httpd[2232]: AH00558: httpd: Could not reli
 1月 28 17:31:34 centos7.example.com systemd[1]: Started The Apache HTTP Server.
```

4.5.2 journald のログの保存

journald のログは、再起動すると消えてしまう設定がデフォルトとなっています。journald の設定ファイル/etc/systemd/journald.conf の Storage 設定の値がデフォルトでは auto に設定されています。この設定は、以下のように動作します。

1. /var/log/journal ディレクトリが存在すれば書き込む
2. /var/log/journal ディレクトリが存在しないか、書き込めない場合には、/run/log/journal ディレクトリに書き込む

デフォルトでは/var/log/journal ディレクトリは存在しないため、/run/log/journal ディレクトリにログが書き込まれます。/run/log/journal ディレクトリは tmpfs でメモリ上に作られた一時領域なので、システム再起動時にログのファイルは消えてしまいます。

journald のログをシステム再起動時に消えないようにするには、以下のように/var/log/journal ディレクトリを作成して、システムを再起動します。

```
# mkdir /var/log/journal
# chmod 700 /var/log/journal
# reboot
```

ログファイルが作成されたことを確認します

```
# ls -l /var/log/journal/
合計 0
drwxr-sr-x. 2 root systemd-journal 49  1月 28 14:53 3b71b9857a284561a3450996bf78a306
# ls -l /var/log/journal/3b71b9857a284561a3450996bf78a306/
合計 16392
-rw-r-----. 1 root root            8388608  1月 28 14:56 system.journal
-rw-r-----+ 1 root systemd-journal 8388608  1月 28 14:55 user-42.journal
```

4.6 firewalld によるパケットフィルタリング

CentOS 7 では、Linux カーネルのパケットフィルタリングの仕組みである iptables を firewalld サービスが管理しています。firewalld を利用すると複雑なパケットフィルタリングを実現できますが、ここでは基本的な設定を解説します。

また、複雑な設定が不要な場合には従来の iptables サービスによる管理に戻すこともできます。

4.6.1 firewalld の設定確認

firewalld では、パケットフィルタリングの設定を「ゾーン」という仕組みで管理しています。

ゾーンを指定しなかった場合に利用されるデフォルトゾーンを確認するには、firewall-cmd コマンドに–get-default-zone オプションを付与して実行します。デフォルトでは public というゾーンが利用されるのが分かります。

```
# firewall-cmd --get-default-zone
public
```

デフォルトゾーン public の設定を確認します。DHCP クライアントと SSH が許可されています。

```
# firewall-cmd --list-all
public (default, active)
  interfaces: eth0
  sources:
  services: dhcpv6-client ssh
  ports:
  masquerade: no
  forward-ports:
```

```
  icmp-blocks:
  rich rules:
```

デフォルトゾーンで許可されているサービスだけを確認するには、–list-services オプションを付与します。

```
# firewall-cmd --list-services
dhcpv6-client ssh
```

定義されているサービスを確認します。ここで確認できるサービスをゾーンに適用することで、受信パケットを許可できます。

```
# firewall-cmd --get-services
amanda-client bacula bacula-client dhcp dhcpv6 dhcpv6-client dns ftp high-availability
     http https imaps ipp ipp-client ipsec kerberos kpasswd ldap ldaps libvirt
     libvirt-tls mdns mountd ms-wbt mysql nfs ntp openvpn pmcd pmproxy pmwebapi
     pmwebapis pop3s postgresql proxy-dhcp radius rpc-bind samba samba-client smtp ssh
     telnet tftp tftp-client transmission-client vnc-server wbem-https
```

4.6.2 firewalld で HTTP を許可する

firewalld の設定を変更して、HTTP の受信を許可します。

–add-service オプションで定義されているサービスをゾーンに適用します。

–permanent オプションを付与すると、ゾーンの設定ファイルである/etc/firewalld/zones/public.xml が修正されて、システム再起動後でも HTTP の受信が許可されるようになります。

```
# firewall-cmd --add-service=http  --permanent
success
# firewall-cmd --list-services
dhcpv6-client http ssh
# cat /etc/firewalld/zones/public.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Public</short>
  <description>For use in public areas. You do not trust the other computers on
       networks to not harm your computer. Only selected incoming connections are
       accepted.</description>
  <service name="dhcpv6-client"/>
  <service name="http"/>
  <service name="ssh"/>
</zone>
```

Web サービスを起動し、外部から Web サーバに接続できることを確認します。

```
# systemctl start httpd
```

4.6.3 iptables を有効にする

firewalld サービスを停止し、iptables サービスを有効にするには以下のコマンドを実行します。

```
# systemctl stop firewalld
# systemctl disable firewalld
# systemctl enable iptables
# systemctl start iptables
```

firewalld サービスを有効に戻すには、以下のコマンドを実行します。

```
# systemctl stop iptables
# systemctl disable iptables
# systemctl enable firewalld
# systemctl start firewalld
```

*NetworkManager 用管理ツール nmtui CentOS 7 でも引き続き NetworkManager を使ってネットワークを管理します。

NetworkManager では、設定ファイルを直接編集することは推奨されていません。GUI と CUI 用の管理ツールが提供されているので、ツールを使って設定を変更します。

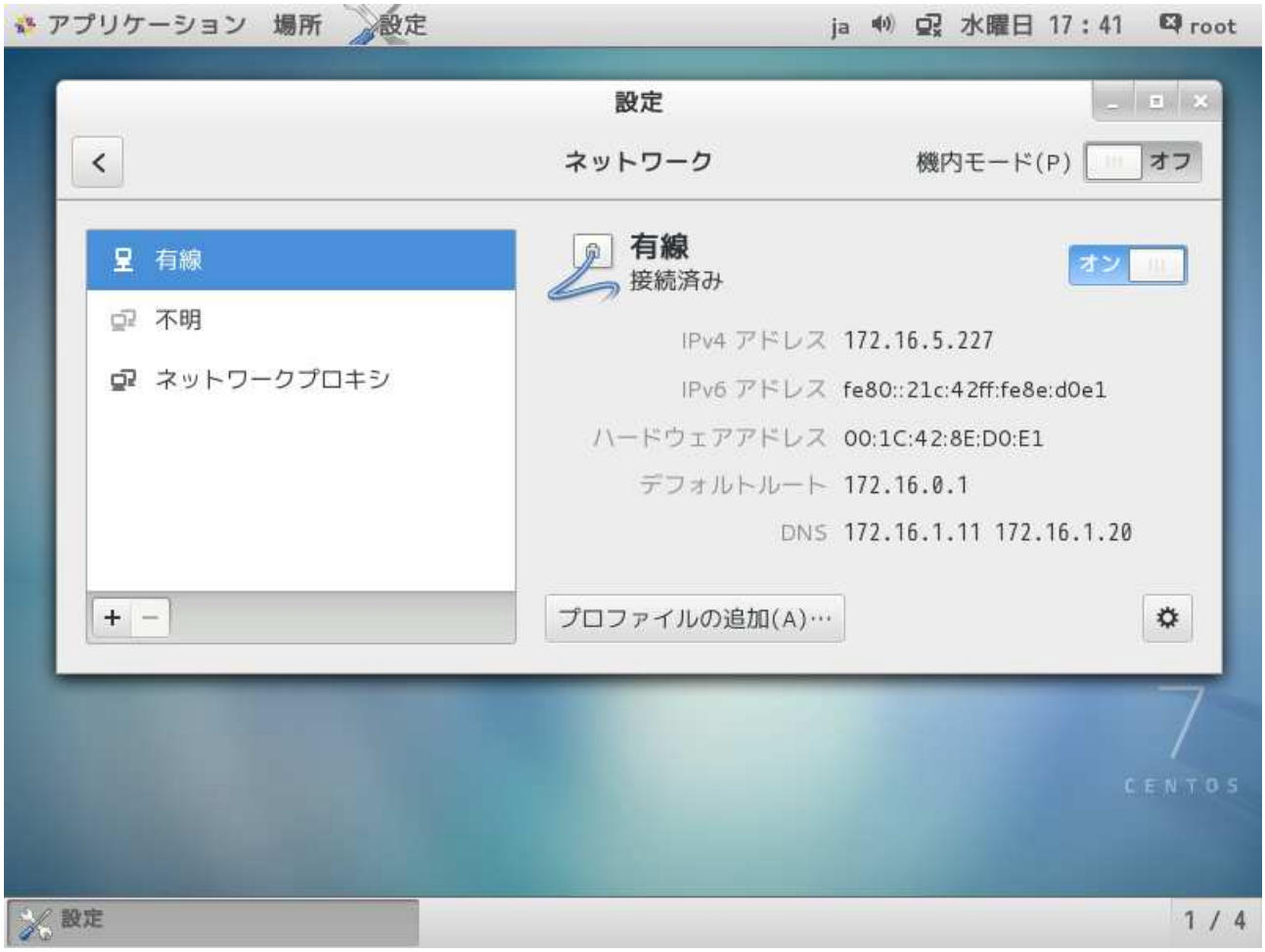


図 38　GUI の NetworkManager 設定画面

GUI では「システムツール」メニューから「設定」を実行することでネットワークの設定も行えます。

CUI では nmtui を実行することで、簡単にネットワークが設定できます。ネットワークインターフェースの IP アドレス等の設定や、有効／無効の切り替えも行えるようになっています。

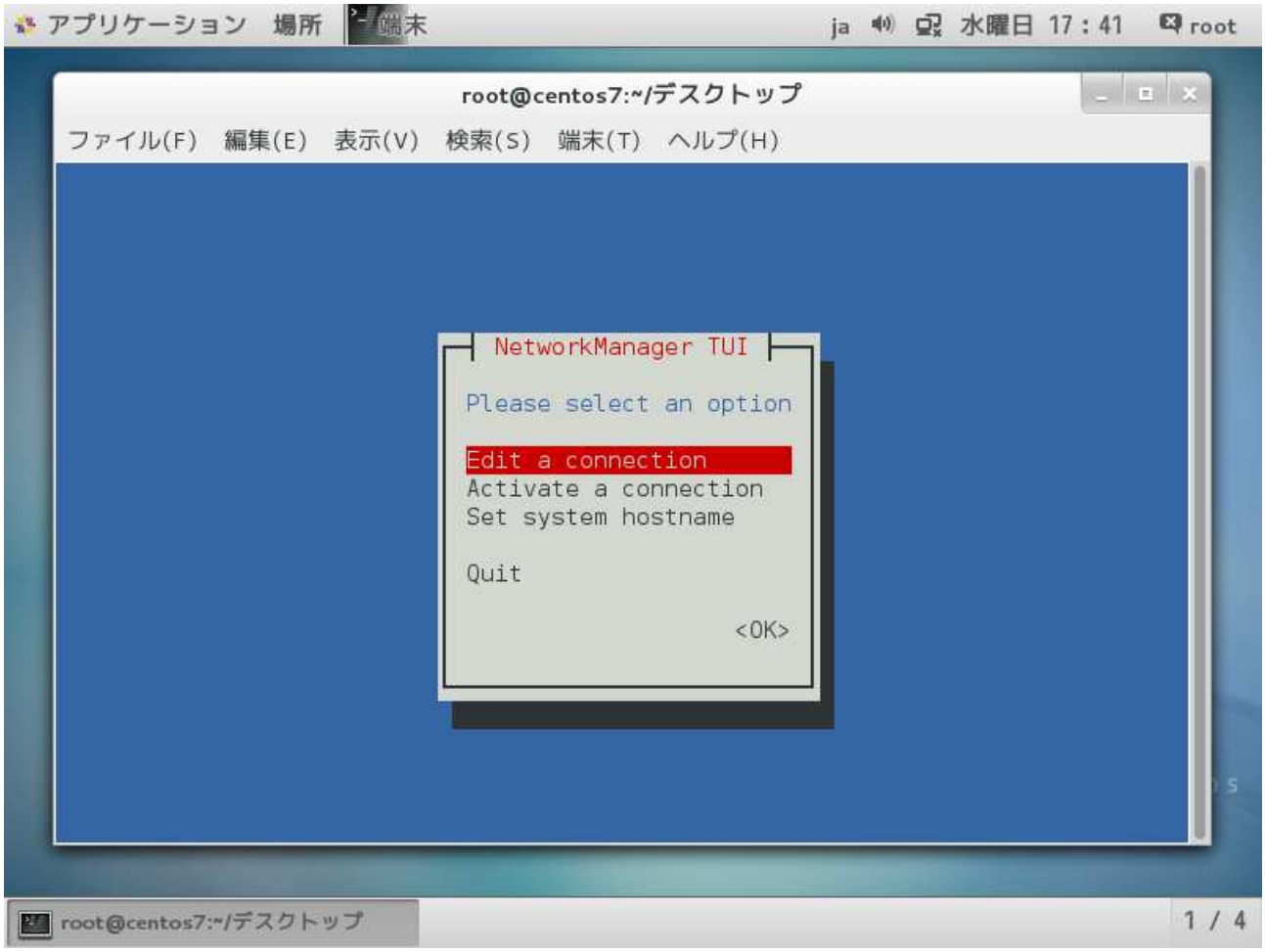


図 39　CUI の NetworkManager 設定画面

2015 年 4 月 16 日 v1.0.0 版発行

2022 年 8 月 22 日 11.0.1 版発行

著 者 LPI-Japan

www.lpi.or.jp