LinuC

リナックスアカデミー

LinuC レベル1
技術解説無料セミナー

～LinuC レベル1 受験に向けての準備とポイント解説～

2019.09.28

Linux Academy 専任講師

森川　恵

LPI-JAPAN

講師プロフィール

リナックスアカデミー

リナックスアカデミー Linux講座専任講師 森川 恵 (もりかわ めぐみ) です。

本日はどうぞよろしくお願いいたします。

★Linuxとの出会いは1996年頃

★Webプログラマー、システムエンジニア、サーバエンジニア、ITコンサルタントとしての職務と並行しながら、プログラマーやエンジニアを養成するための講師として20年近くリナックスアカデミーに在籍しています。

（系列のITスクール リカレント在籍期間を含む）

LinuC

本セミナーについて

◆本セミナーは試験対策講座ではございません。
時間の都合上、試験のご説明及びポイント解説のみとなり、進行状況によっては予定内容の全てをご紹介しきれない場合もございます。

なお、このセミナー資料はRedhat系ディストリビューションCentOS7環境を前提とした内容となっています。

◆受講者の想定スキルレベル

- PCの基本操作はできるがLinuxの学習は初めての方
- LinuCレベル1の受験を検討している方

◆本セミナーのゴール

- Linuxの基本コマンドやファイル管理の重要性が理解できる
- システム運用に必要な基本的な手法を知る事ができる

1. LinuC Level 1 試験について

- 試験概要
- 学習の仕方

2. 101試験出題範囲よりポイント解説（抜粋）

- 基本的なファイル管理の実行
- ファイルのパーミッションと所有者の管理

3. 102試験出題範囲よりポイント解説（抜粋）

- シェル環境のカスタマイズと簡単なシェルスクリプト
- ジョブスケジューリングによるシステム管理業務の自動化
- セキュリティ（ホストのセキュリティ設定/暗号化によるデータの保護）

※適宜、休憩をはさみます

1.　LinuC Level 1 試験について

■試験概要

- ✓ **試験内容：Linuxシステムの構築・運用・管理**

  実務で必要な**Linuxの基本操作とシステム管理が行えるエンジニア**であることを証明できます

  Linuxサーバー環境の構築・運用・保守、インフラエンジニア、ネットワークエンジニア、セキュリティエンジニア、データセンター構築、Linuxや組込み系のソフトウェア開発業務、SE営業職、IT研修インストラクター、などの職種

- ✓ **該当試験：101試験 / 102試験**

  LinuCレベル1に認定されるためには**101試験と102試験の両方に合格**する必要があります

- ✓ **受験準備：勉強期間目安1か月～3か月程度**（※初めての方の場合の目安）

- ✓ **その他詳細　→　https://linuc.org/linuc1/**

■学習の仕方

1. 出題範囲を確認する　→　効率的な学習を目指す！

   101試験　https://linuc.org/linuc1/range/101.html

   102試験　https://linuc.org/linuc1/range/102.html

2. 自分に合った学習方法、教材を選ぶ　→　多角的なアプローチがおススメ！
3. 繰り返し問題を解く　→　自身の弱点を重点的に反復！
4. Linux環境を準備し、コマンド操作の実習をする　→　頭だけでなく体で覚える！
5. コマンドは語源から意味を知る　→　コマンドの意味を理解しやすく！

■学習の仕方

～学習環境の準備について～

Linuxを学習するためには、仮想化環境のためのソフトウェアをインストールし、その環境下にLinuxをインストールするのがおすすめです。

複数のディストリビューションをインストールしたり、ネットワーク設定の検証をしたりなど、実機を用いた学習は、より理解度が向上します。

★主な仮想化環境ソフト

<ホスト型：テスト環境、開発環境用>

- VirtualBox（Windows,MacOS,Linux,Oracle Solaris）
- VMware Workstation Player（Windows,Linux）
- Parallels Desktop（MacOS）

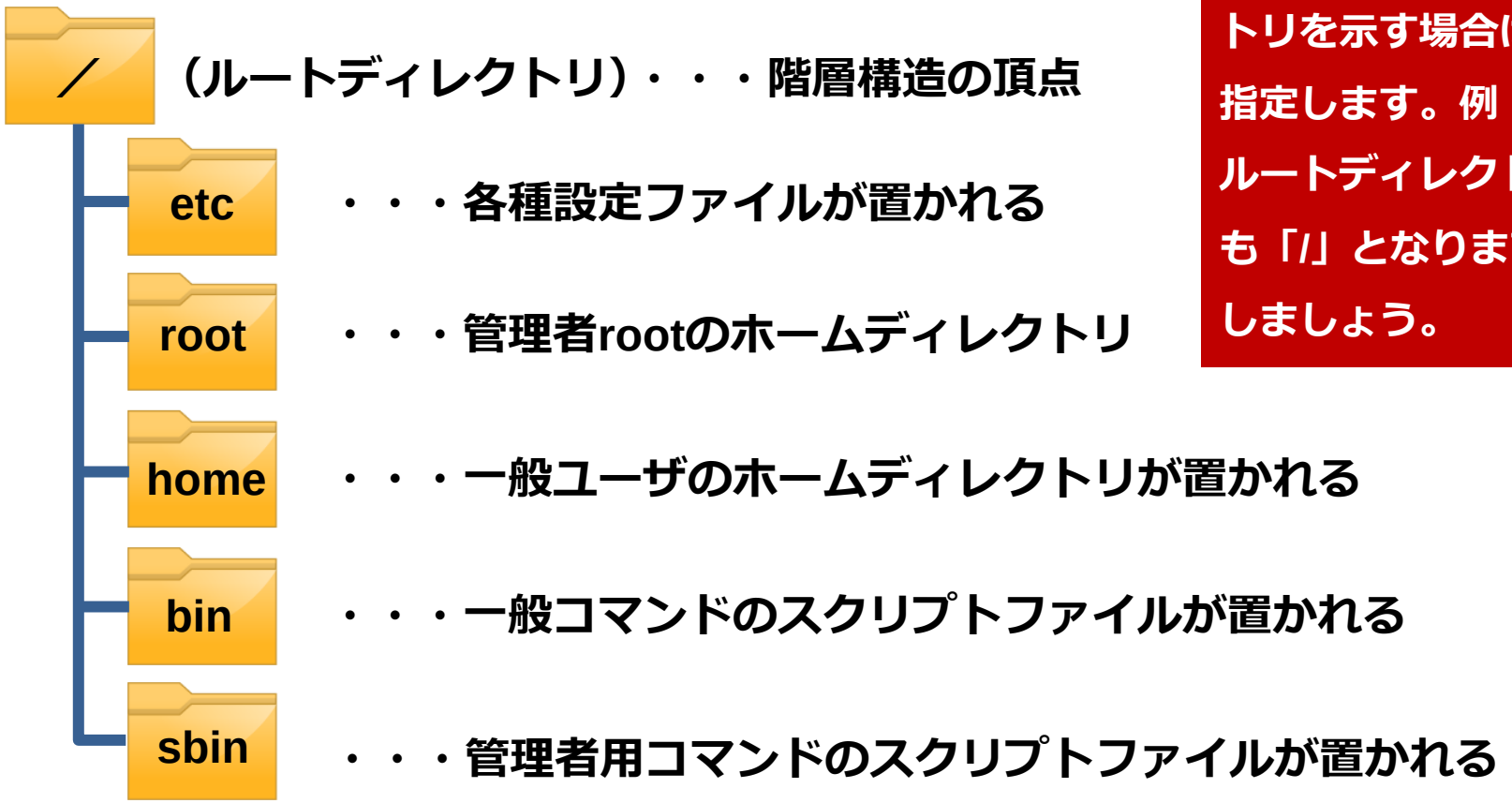
<ハイパーバイザー型：実際の仮想化用>

- Hyper-V
- VMware vSphere
- KVM
- Xen

■学習の仕方

～Linux学習のコツ その１～

★ディレクトリの階層構造を理解しましょう！



ルートディレクトリ「/」を基準としたファイルの位置指定を絶対パスといい、「/etc」などと表記します。さらに下位のディレクトリを示す場合は「/」を区切り文字として指定します。例：/home/user1
ルートディレクトリと区切り文字がいずれも「/」となりますので、混乱しないようにしましょう。

■学習の仕方

～Linux学習のコツ その２～

★ユーザ権限を確認しましょう！

Linuxのコマンドは、一般ユーザが実行できるものと管理者ユーザrootのみが実行できるものがあります。管理者権限で実行する場面では、管理者に切り替えるか sudoコマンドを利用してコマンド実行する必要があります。（sudoコマンドの利用可否はディストリビューションや設定に依存します。）

今回は、管理者権限の場合のプロンプト表示と一般ユーザ権限の場合のプロンプト表示を確認しながら進めていきます。

＜管理者権限＞　[root@localhost ~] #

＜一般ユーザ権限＞　[user1@localhost ~] $
ユーザ名

■学習の仕方

～Linux学習のコツ その３～

★分からないコマンドは調べましょう！

あいまいな知識でコマンドを実行すると思わぬ事態となる恐れがあります。コマンドの使用方法はマニュアルなどを参照してから実行するように心がけましょう。また、試験対策としては各種オプションや引数の指定方法をできる限りマスターしましょう。

＜マニュアルの参照：manコマンド＞

man コマンド名　→　例：$ man passwd

＜インターネット検索の利用＞

検索キーワードに「manpage コマンド名」と指定して検索

※ディストリビューションやカーネルバージョンによってコマンドが異なる場合がありますので、注意しましょう。

2.　101試験出題範囲よりポイント解説

- **基本的なファイル管理の実行**
- ファイルのパーミッションと所有者の管理

■基本的なファイル管理の実行

<主要な知識範囲>

- 個々のファイルおよびディレクトリをコピー、移動、削除する
- 複数のファイルおよびディレクトリを再帰的にコピーする
- ファイルおよびディレクトリを再帰的に削除する
- 基本的なものから高度なものまで、ワイルドカード規則をコマンドで使用する
- **findを使用して、種類、サイズ、または時刻を基にファイルを見つけて操作する**
- **tar、cpioおよびddの使用方法**

■基本的なファイル管理の実行

★findコマンド - ディレクトリ階層をたどって、条件を満たすファイルを検索する

書式： find [オプション] [対象ディレクトリ] [検索オプション] [アクション]

◆主なオプション

オプション	意味
-P	シンボリックリンクをたどらない（デフォルト）
-L	すべてのシンボリックリンクをたどる
-H	コマンドラインで指定したシンボリックをたどる

■基本的なファイル管理の実行

★findコマンド - ディレクトリ階層をたどって、条件を満たすファイルを検索する

書式： find [オプション] [対象ディレクトリ] [検索オプション] [アクション]

◆主な検索オプション

オプション	意味
-name ファイル名	ファイル名/ディレクトリ名で検索（ワイルドカード使用可）
-size サイズ	ファイルサイズで検索
-mtime 日数	更新日時で検索
-user ユーザ名	所有ユーザ名で検索
-perm パーミッション	パーミッションの値で検索
-type ファイルの種類	ファイルの種類で検索 （「f」レギュラーファイル 「d」ディレクトリ 「l」シンボリックリンク）

■基本的なファイル管理の実行

★findコマンド - ディレクトリ階層をたどって、条件を満たすファイルを検索する

書式： find [オプション] [対象ディレクトリ] [検索オプション] [アクション]

◆主なアクション

オプション	意味
-print	検索結果を表示（デフォルト）
-ls	検索結果を詳細表示
-exec コマンド {} ¥;	検索結果に対し指定したコマンドを実行 （検索結果が-execで指定したコマンドの引数となる。 「;」までをコマンドとして認識。）

■基本的なファイル管理の実行

★findコマンドの使用例

＜指定した文字列を含むファイルを検索＞

```
find . -name “*test*’ -print
```

← 文字列「test」が含まれるファイルを検索し表示

＜指定した最終アクセス情報に該当するファイルを削除（rmコマンドを実行）＞

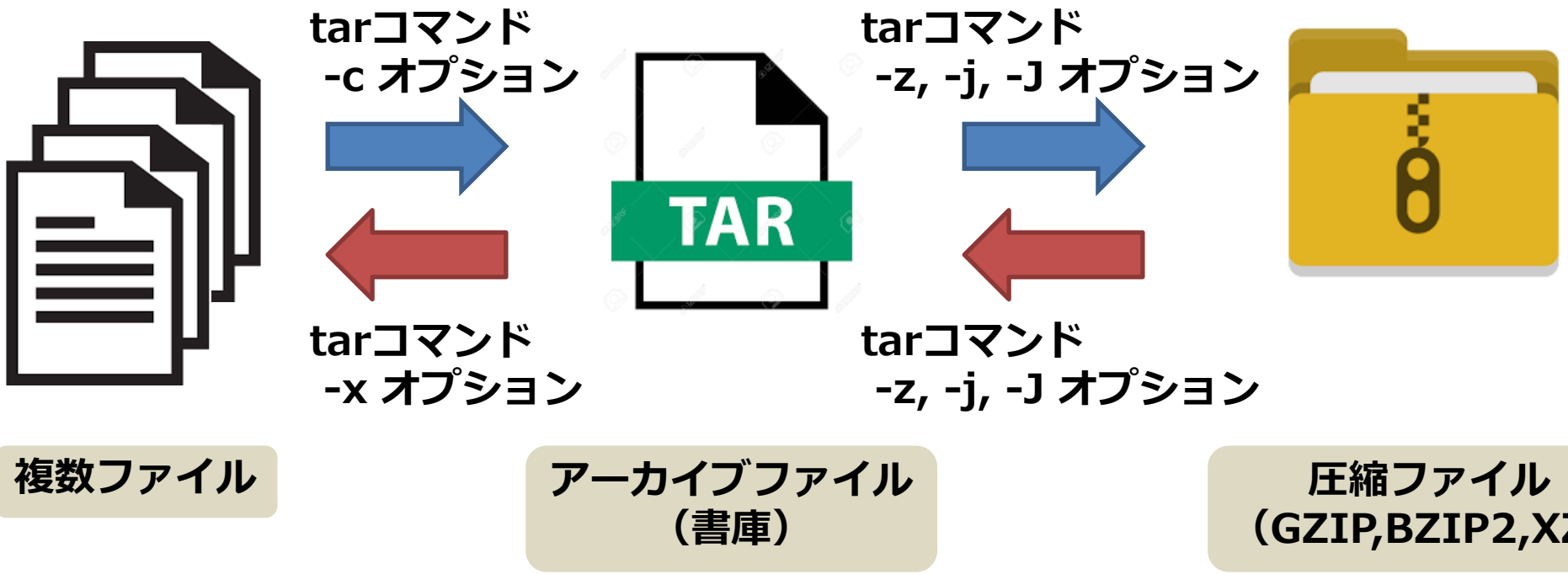
```
find . -atime +30 -exec rm {} ¥;
```

← 最終更新日時から30日以上経過したファイルを削除

findコマンドの実行結果がrmコマンドの引数になる

■基本的なファイル管理の実行

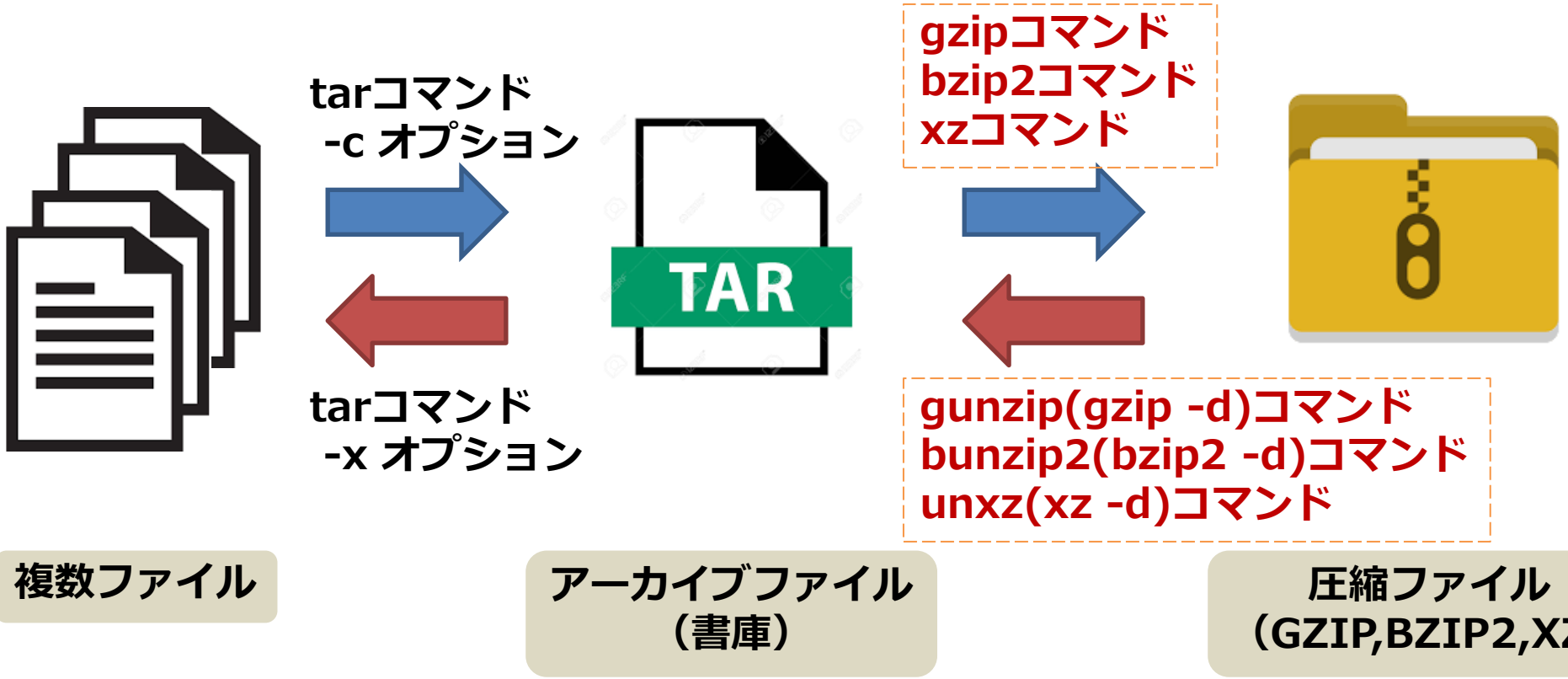
★ファイルのアーカイブと圧縮について（その１）



※tarball形式のファイルと呼ばれる

■基本的なファイル管理の実行

★ファイルのアーカイブと圧縮について（その２）



※tarball形式のファイルと呼ばれる

■基本的なファイル管理の実行

★tarコマンド - GNU 版 tar アーカイブ・ユーティリティー
("tape archives" の略) ※元々は磁気テープ用にアーカイブするためのコマンド

書式： tar [オプション] アーカイブファイル名

tar [オプション] アーカイブファイル名 ファイル名/ディレクトリ名

<参考>

～オプションにハイフン「-」を付ける／付けない～

古い記法ではハイフンが不要でしたが、その後、オプションにハイフンを付けるルールに対応しました。現在ではどちらの記法も利用可能です。ただし、ハイフン付きで使用できないオプションもあります。

■基本的なファイル管理の実行

★tarコマンドの主なオプション一覧

短いオプション	長いオプション	意味
-c	--create	新しいアーカイブファイルを作成
-r	--append	アーカイブファイルの最後にファイルを追加
-x	--extract, --get	アーカイブファイルの展開
-t	--list	アーカイブファイルの内容を一覧表示
-f ファイル名	--file=ファイル名	アーカイブファイル名の指定
-v	--verbose	処理内容の詳細表示
-z	--gzip	GZIP形式に圧縮または解凍(拡張子は tar.gz, tgz)
-j	--bzip2	BZIP2形式に圧縮または解凍(拡張子は tar.bz2)
-J	--xz	XZ形式に圧縮または解凍(拡張子は tar.xz)

■基本的なファイル管理の実行

★tarコマンドの使用例

<アーカイブファイルの作成>

```
tar -cvf archive.tar test{1,2,3}*
```

<GZIP圧縮形式のtarballファイルの作成>

```
tar -czvf archive.tar.gz test{1,2,3}*
```

<アーカイブファイルの展開>

```
tar -xvf archive.tar /home/user1
```

<GZIP圧縮形式のtarballファイルの展開>

```
tar -xzvf archive.tar.gz /home/user1
```

■基本的なファイル管理の実行

★cpioコマンド - アーカイブファイルへのファイルのコピーや、
アーカイブファイルからファイルへのコピーを行う
（“copy in, copy out” の略）

書式： cpio オプション

◆オプション

オプション	意味
-o	コピーアウトモード（アーカイブの作成）
-i	コピーインモード（アーカイブからファイルを取り出す）
-d	-iオプション同時指定の場合に、必要に応じてディレクトリ作成
-t	-iオプション同時指定の場合に、ファイルの一覧を表示

■基本的なファイル管理の実行

★cpioコマンドの使用例

<アーカイブファイルの作成>

```
ls test{1,2,3}* | cpio -o > cpio.file
find . -name ‘test[1-3]*’ -print  | cpio –o > cpio.file
```

<アーカイブファイルからファイルの取り出し>

```
cpio -i < cpio.file
cat cpio.file | cpio -i
```

結果は同じ

<参考>

～cpioコマンドには引数を指定できない～

cpioコマンドはアーカイブ処理をおこなうコマンドのため、ファイルの指定は出力リダイレクト、入力リダイレクト、パイプラインを利用します。

■基本的なファイル管理の実行

★tarコマンドとcpioコマンドの違い（参考）

<tarコマンド>

・アーカイブの先頭に管理情報をまとめて記録するので、
先頭の情報が破損するとすべてのファイルが復元不可能となる。

・取得するファイルやサブディレクトリを指定できない。

<cpioコマンド>

・アーカイブの中でファイルごとに管理情報を記録するので、
アーカイブの一部が破損しても部分的な被害で済む可能性がある。

・パイプを使用して取得するファイルしてディレクトリを指定できる。

■基本的なファイル管理の実行

★ddコマンド - ファイルの変換とコピーを行う
（“dataset definition” の略。※他にも諸説あり）

書式：

dd [オプション] if=出力元ファイルのパス of=出力先ファイルのパス [bs=バイト数]

※パスにデバイス名を指定することもできる

■基本的なファイル管理の実行

★ddコマンドの使用例

</dev/sda1にマウントされている/bootディレクトリをアーカイブとしてバックアップ>

```
dd if=/dev/sda1 of=boot.backup
```

</dev/sda2の先頭セクタ512バイトを/dev/sdbにコピー>

```
dd if=/dev/sda2 of=/dev/sdb bs=512
```

<CD-ROM(/dev/cdrom)からISOイメージファイルを作成>

```
dd if=/dev/cdrom of=file.iso
```

2.　101試験出題範囲よりポイント解説

- 基本的なファイル管理の実行
- ファイルのパーミッションと所有者の管理

■ファイルのパーミッションと所有者の管理

<主要な知識範囲>

- 通常ファイル、スペシャルファイル、およびディレクトリに対するアクセスパーミッションを管理する
- SUID、SGID、スティッキービットなどのアクセスモードを使用して、セキュリティを維持する
- ファイル作成マスクの変更方法を把握する
- グループフィールドを使用して、グループメンバーがファイルにアクセスできるようにする

■ファイルのパーミッションと所有者の管理

★chmodコマンド - ファイルのアクセス権を変更する (“change mode” の略)

書式： chmod [オプション] モード ファイル名またはディレクトリ名

◆オプション

オプション	意味
-R	指定したディレクトリ以下のファイルやサブディレクトリも対象とする

◆モード

モード	内容
シンボルモード	パーミッションを対象、操作、権限それぞれの記号で指定
オクタルモード	パーミッションを数値で指定

■ファイルのパーミッションと所有者の管理

★chmodコマンド

◆シンボルモードで使用できる記号

対象		操作		権限	
u	所有ユーザ	+	追加	r	読み
g	所有グループ	-	削除	w	書き
o	その他	=	設定	x	実行
a	全て			s	SUID/SGID
				t	sticky bit

■ファイルのパーミッションと所有者の管理

★chmodコマンド

◆オクタルモードによるパーミッション例

所有者	グループ	その他
rwx	rwx	rwx
7	7	7

所有者	グループ	その他
rwx	r-x	r-x
7	5	5

所有者	グループ	その他
rw-	rw-	rw-
6	6	6

所有者	グループ	その他
rw-	r--	r--
6	4	4

所有者	グループ	その他
rw-	---	--x
6	0	1

所有者	グループ	その他
r--	---	---
4	0	0

	rwx
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

r:4
w:2
x:1

■ファイルのパーミッションと所有者の管理

★chmodコマンドの使用例

```
chmod ug+w /home/user1/test.txt
chmod a-x /home/user1/test.txt
chmod u+rw,g+r /home/user1/test.txt
chmod u=r /home/user1/test.txt

chmod 664 /home/user1/test.txt
chmod 666 /home/user1/test.txt
chmod 640 /home/user1/test.txt
chmod 400 /home/user1/test.txt

chmod -R go-rwx /home/user1/testdir
chmod -R u=rw /home/user1/testdir

chmod -R 600 /home/user1/testdir
```

■ファイルのパーミッションと所有者の管理

★パーミッションとマスク値

ユーザがファイルやディレクトリを作成すると、マスク値を参照した上でデフォルトのパーミッションが設定されます。umaskコマンドを使用して、現在のマスク値を確認したり、他のマスク値に変更することができます。

ただし、マスク値を変更した場合は、現在起動中のシェルでのみ有効となります。

■ファイルのパーミッションと所有者の管理

★マスク値について

＜既定のマスク値＞

レギュラーファイル　　：0666
ディレクトリ　　：0777

＜ファイル作成時のパーミッション＞

マスク値が0022の場合　→　0666 – 0022 = 0644 / 0777 – 0022 = 0755
マスク値が0002の場合　→　0666 – 0002 = 0664 / 0777 – 0002 = 0775

■ファイルのパーミッションと所有者の管理

★umaskコマンド - ファイルのアクセス権を変更する (“user mask” の略)

書式：umask　・・・確認

umask [オプション] マスク値　・・・変更

※オプションには、-p や –S が使用できますが、ここでは説明を割愛します。

★umaskコマンドの使用例

＜umask値を0022に変更＞

```
umask 0022
```

3. 102試験出題範囲よりポイント解説

- シェル環境のカスタマイズと簡単なシェルスクリプト
- ジョブスケジューリングによるシステム管理業務の自動化
- セキュリティ
  （ホストのセキュリティ設定/暗号化によるデータの保護）

■シェル環境のカスタマイズと簡単なシェルスクリプト

＜主要な知識範囲＞

- ログイン時または新しいシェルを生成したときに、環境変数（PATHなど）を設定する
- **よく使用する一連のコマンド用にBashの関数を作成する**
- **新しいユーザアカウント用のスケルトンディレクトリを保守する**
- コマンドサーチパスを適切なディレクトリに設定する

■シェル環境のカスタマイズと簡単なシェルスクリプト

★シェルの種類

Bash

Bash(GNU Bash)は、GNUプロジェクトによるプロダクトで、多くのUNIX/Linux系OSで標準的に利用されているシェルです。基本的機能を備えているのでカスタマイズせずに利用できます。

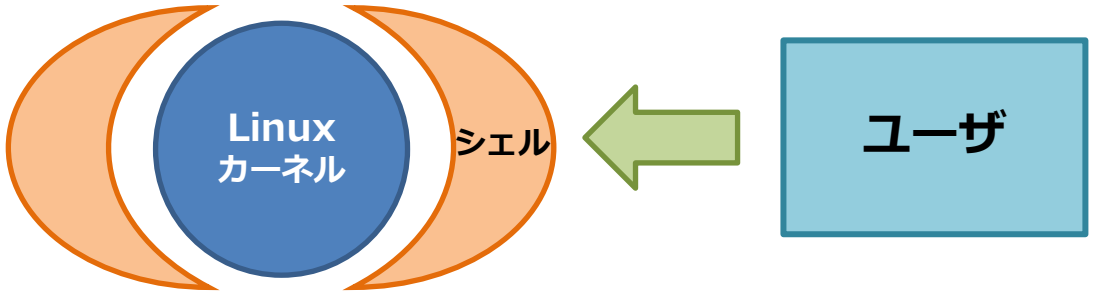
Zsh

UNIXコマンドインタプリタ(シェル)であり、「対話型ログインシェル」および「シェルスクリプトコマンドプロセッサ」として使用できます。
「Bash」「ksh」「tcsh」などのシェル機能を取り込み、多数の改良が加えられている拡張Bourneシェルで、強力なスクリプト言語としても利用できます。
また、さまざまなプラグインでカスタマイズして利用することが可能です。

■シェル環境のカスタマイズと簡単なシェルスクリプト

★シェルと環境

シェルはLinuxカーネル（OSの中核）と、ユーザーの間に位置するレイヤーです。



シェルは、コマンドラインで直接実行するほか、実行するコマンドをあらかじめスクリプトファイルに記述し、そのスクリプトファイルを実行するだけで何度も繰り返す必要のある一連のコマンドを簡単に実行することができます。また、functionコマンドを使用し関数を定義して直接実行したり、関数をシェルスクリプトに記述した上で実行したりすることもできます。

■シェル環境のカスタマイズと簡単なシェルスクリプト

★シェルスクリプトの実行

◆シェルスクリプト実行方法のいろいろ

実行方法	必要な権限	実行例	備考
source コマンド	r(読み)	source スクリプトファイル名	
.	r(読み)	. スクリプトファイル名	
bash コマンド	r(読み)	bash スクリプトファイル名	
./	r(読み)とx(実行)	./スクリプトファイル名	ディレクトリにパスが通っていない場合にカレントディレクトリ「./」を明示する必要がある。絶対パスを指定してもよい。

■シェル環境のカスタマイズと簡単なシェルスクリプト

★関数の定義方法と実行

シェルでは、関数の定義をする際functionコマンドを使用します。
その関数を実行（呼び出し）するには関数名の後に必要な引数を指定します。
関数をスクリプトファイル内に記述した場合には、シェルスクリプトを実行する
いずれかの方法でスクリプトを実行します。

★functionコマンド – 関数を定義する

関数の定義：

```
function 関数名() {
    実行するコマンド
}
```

関数の呼び出し：

```
関数名　引数1 引数2　・　・
```

■シェル環境のカスタマイズと簡単なシェルスクリプト

★関数の定義例

例題：引数に指定した数値を合計する関数
　　関数名 「goukei」

① 合計値を格納する変数kotaeに初期値0をセット
② 引数指定が3つ未満の場合メッセージ表示し終了
③ 引数の値を1つずつ取り出し変数kotaeに加算代入処理を繰り返す
④ 全ての引数を合計した値（変数kotae）を表示
　戻り値0

```
#!/bin/bash
function goukei() {
    kotae=0
    if [ $# -lt 3 ]
            then
            echo “Please specify at least 3 parameters“
            exit
    fi
    for val in $@
    do
        kotae =`expr $kotae + $val`
    done
    echo “Answer: $kotae"
    return 0
}
goukei 100 200 300
```

■シェル環境のカスタマイズと簡単なシェルスクリプト

★関数の定義例

例題：引数に指定した数値を合計する関数
　　関数名 「goukei」

<条件分岐構文>　if 条件 then ～ fi
<繰り返し構文>　for 繰り返し条件 do ～ done
<bashの特殊変数>
　$#　・・・引数の個数
　$@　・・・全ての引数

```
#!/bin/bash
function goukei() {
    kotae=0
    if [ $# -lt 3 ]
            then
            echo “Please specify at least 3 parameters“
            exit
    fi
    for val in $@
    do
        kotae =`expr $kotae + $val`
    done
    echo “Answer: $kotae"
    return 0
}
goukei 100 200 300
```

■シェル環境のカスタマイズと簡単なシェルスクリプト

★関数の定義例

例題：引数に指定した数値を合計する関数
　　関数名 「goukei」

$ source スクリプトファイル名

Answer: 600

```
#!/bin/bash
function goukei() {
    kotae=0
    if [ $# -lt 3 ]
            then
            echo “Please specify at least 3 parameters“
            exit
    fi
    for val in $@
    do
        kotae =`expr $kotae + $val`
    done
    echo “Answer: $kotae"
    return 0
}
goukei 100 200 300
```

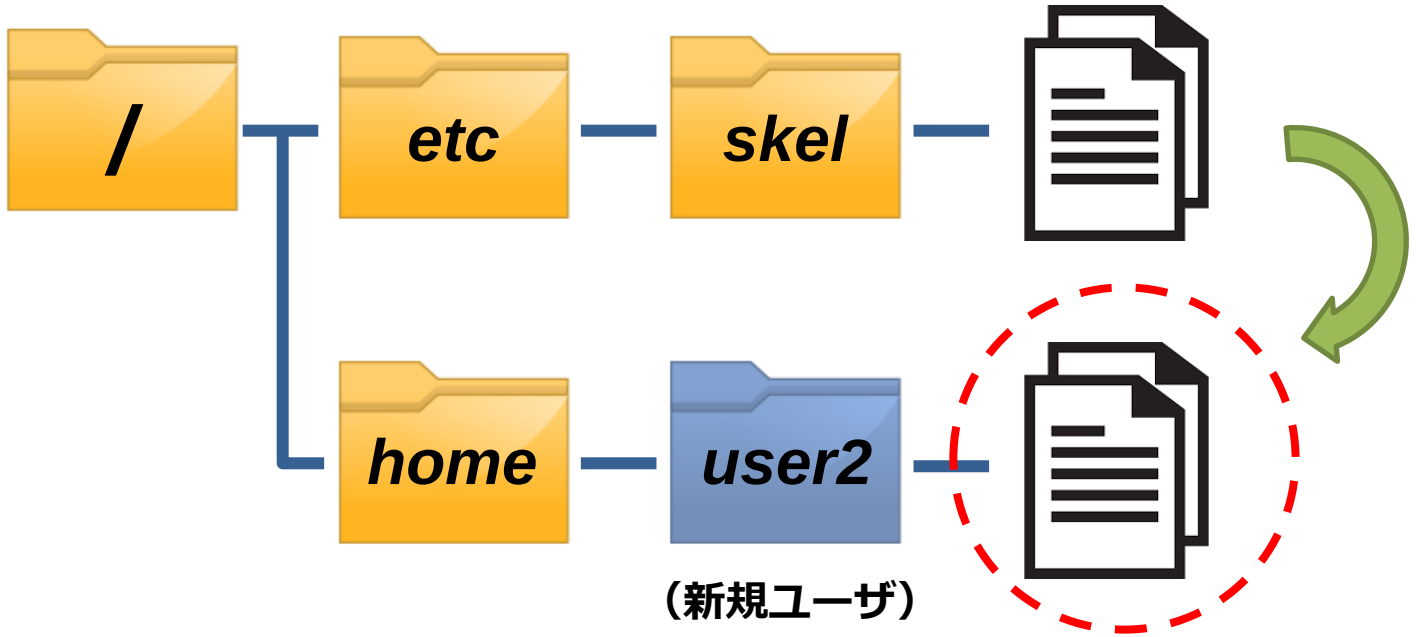
■シェル環境のカスタマイズと簡単なシェルスクリプト

★スケルトンディレクトリについて

useradd コマンドでユーザーアカウントを作成すると、通常/home ディレクトリ配下にそのユーザーのホームディレクトリも同時に作成されますが、その際、/etc/skel ディレクトリ内にあるファイルがホームディレクトリ内にコピーされます。
このスケルトンディレクトリにどのユーザーにも必要と思われる基本的な設定ファイルなどを格納しておくことができます。

■シェル環境のカスタマイズと簡単なシェルスクリプト

★スケルトンディレクトリについて



新規ユーザを追加すると
ファイルが自動的に
コピーされる

■シェル環境のカスタマイズと簡単なシェルスクリプト

★スケルトンディレクトリについて

デフォルトでは/etc/skel ですが、useraddの設定ファイル/etc/default/useradd ファイルで設定変更が可能です。

```
# useradd defaults file
.
.
SKEL=/etc/skel
```

← この部分を変更する

3. 102試験出題範囲よりポイント解説

- シェル環境のカスタマイズと簡単なシェルスクリプト
- **ジョブスケジューリングによるシステム管理業務の自動化**
- セキュリティ
  （ホストのセキュリティ設定/暗号化によるデータの保護）

■ジョブスケジューリングによるシステム管理業務の自動化

＜主要な知識範囲＞

- cronおよびatでジョブを管理する
- ユーザがcronおよびatサービスにアクセスできるよう設定する
- anacronの設定

■ジョブスケジューリングによるシステム管理業務の自動化

★タスクの自動化

自動化タスクユーティリティには、cron、anacron、at、 batch があり、cron と anacron は繰り返されるジョブのスケジュール管理（定期実行）をするのに対し、at と batch は1回のみのジョブのスケジュール管理（指定時実行）をします。

★cron と anacron の違い

cron と anacron はいずれも、時刻、月のうちの日付、月、曜日、週で定義されたある時点で定期的なタスクの実行をスケジュールすることができるデーモンです。
cron は、最高で1分おきの頻度でジョブの実行が可能です。ジョブがスケジュールされている時にシステムが稼働していない場合は、ジョブは実行されません。
一方でanacron は、ジョブがスケジュールされている時点でシステムが稼働していなくても、ジョブを記憶しており、次回システム起動した際にジョブを実行します。ただし、anacron は1日に1回しかジョブを実行できません。

■ジョブスケジューリングによるシステム管理業務の自動化

★cronによるジョブの定期実行

cron は crond というデーモンによって管理されています。cronの設定は一般ユーザが行う場合は crontab コマンドを使用し編集などを行います。管理者 root がシステム用 cron を設定する場合は、各設定ファイルをviエディタで編集します。

◆cronの設定ファイル

ユーザ cron	/var/spool/cron/	コマンド「crontab -e」を実行しviコマンドで編集を行うとこのディレクトリに実行ユーザごとのファイルが作成される
システム cron	/etc/cron.hourly/ /etc/cron.daily/ /etc/cron.weekly/ /etc/cron.monthly/	各ディレクトリ内にスクリプトを準備しておき、 /etc/crontab ファイルの設定内容に従って実行される
	/etc/cron.d/	このディレクトリにジョブの実行日時などを記述したファイルを格納する
	/etc/crontabファイル	システム全体に関する内容を記述する

■ジョブスケジューリングによるシステム管理業務の自動化

★cronのアクセス制御 その1

cron のアクセス制御ファイルは、 /etc/cron.allow ファイル(許可)と /etc/cron.deny ファイル(拒否)です。1 行につき 1 ユーザ名の書式で記述します。このファイルの編集後に crond のサービス再起動は不要です。ユーザがジョブの追加又は削除する度にアクセス制御のチェックがされます。

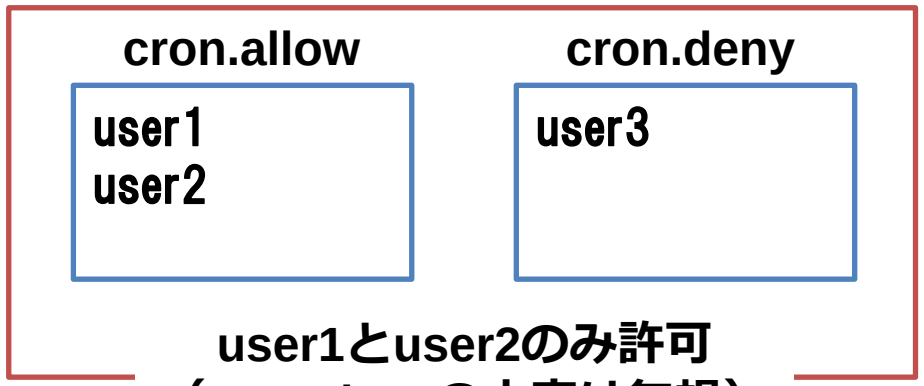
なお、アクセス制御ファイルに記載されているユーザ名に関わらず、管理者 root は常に cron を使用することができます。

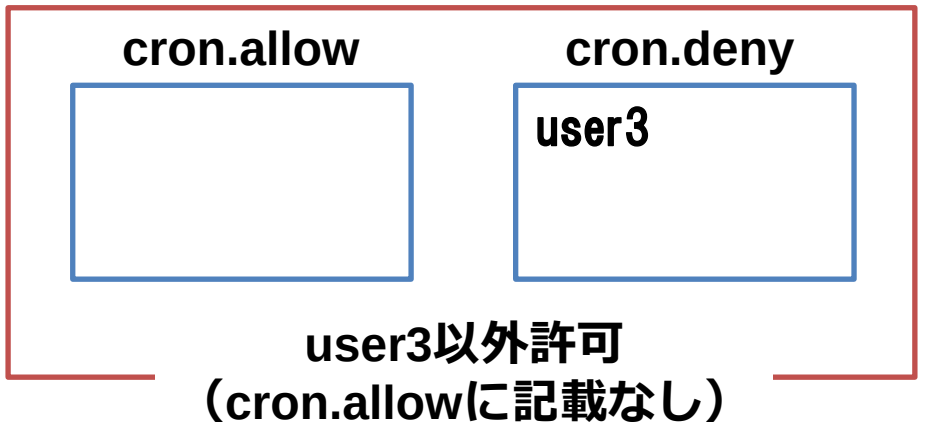
■ジョブスケジューリングによるシステム管理業務の自動化

★cronのアクセス制御 その2

cron.allow ファイルが存在する場合は、そのファイルに記載されているユーザのみ cron の使用を許可され、cron.deny ファイルは無視されます。

cron.allow が存在しない場合、cron.deny に表示されているユーザは cron を使用することができません。





■ジョブスケジューリングによるシステム管理業務の自動化

★cronのアクセス制御 その3

アクセスは制御はPAM(Pluggable Authentication Modules)を利用することもできます。
設定値は /etc/security/access.conf に格納されています。

＜管理者 root 以外のユーザに crontab の作成を禁止する場合＞

```
-:ALL EXCEPT root :cron
```

3.　102試験出題範囲よりポイント解説

- シェル環境のカスタマイズと簡単なシェルスクリプト
- ジョブスケジューリングによるシステム管理業務の自動化
- **セキュリティ**
  （ホストのセキュリティ設定/暗号化によるデータの保護）

■セキュリティ（ホストのセキュリティ設定）

＜主要な知識範囲＞

- シャドウパスワードおよびその機能について知っている
- 使用していないネットワークサービスをオフにする
- TCPラッパーの役割について理解している

■セキュリティ（ホストのセキュリティ設定）

★シャドウパスワードとは

安全にパスワードを管理するための仕組みのこと。
ユーザ情報ファイル/etc/passwordファイルやグループ情報ファイル/etc/group ファイルに管理せず、よりアクセス権の厳しい/etc/shadowファイル、/etc/gshadow ファイルに暗号化パスワードを管理します。

シャドウパスワード管理方式と旧来のパスワード管理方式の切り替えのためのコマンドには、pwconvコマンド、pwunconvコマンド、grpconvコマンド、grpunconvコマンド が

あります。

■セキュリティ（ホストのセキュリティ設定）

★ユーザパスワードのシャドウパスワード管理

<シャドウパスワード管理をしない場合>

/etc/password ファイル

ユーザ名:暗号化パスワード:ユーザID:グループID:～

pwconv →

← pwunconv

<シャドウパスワード管理をする場合>

/etc/passwd ファイル

ユーザ名:x:ユーザID:グループID:～

/etc/shadow ファイル

ユーザ名:暗号化パスワード:～

※ユーザパスワードとは、ログイン時に使用するパスワードのことです。

■セキュリティ（ホストのセキュリティ設定）

★グループパスワードのシャドウパスワード管理

<シャドウパスワード管理をしない場合>

/etc/group ファイル

グループ名:暗号化パスワード:グループID:～

grpconv

<シャドウパスワード管理をする場合>

/etc/group ファイル

グループ名:x:グループID:～

grpunconv

/etc/gshadow ファイル

グループ名:暗号化パスワード:～

※グループパスワードを設定（gpasswdコマンドを実行）すると、
グループにユーザを追加する時などにパスワードが求められます。

■セキュリティ（ホストのセキュリティ設定）

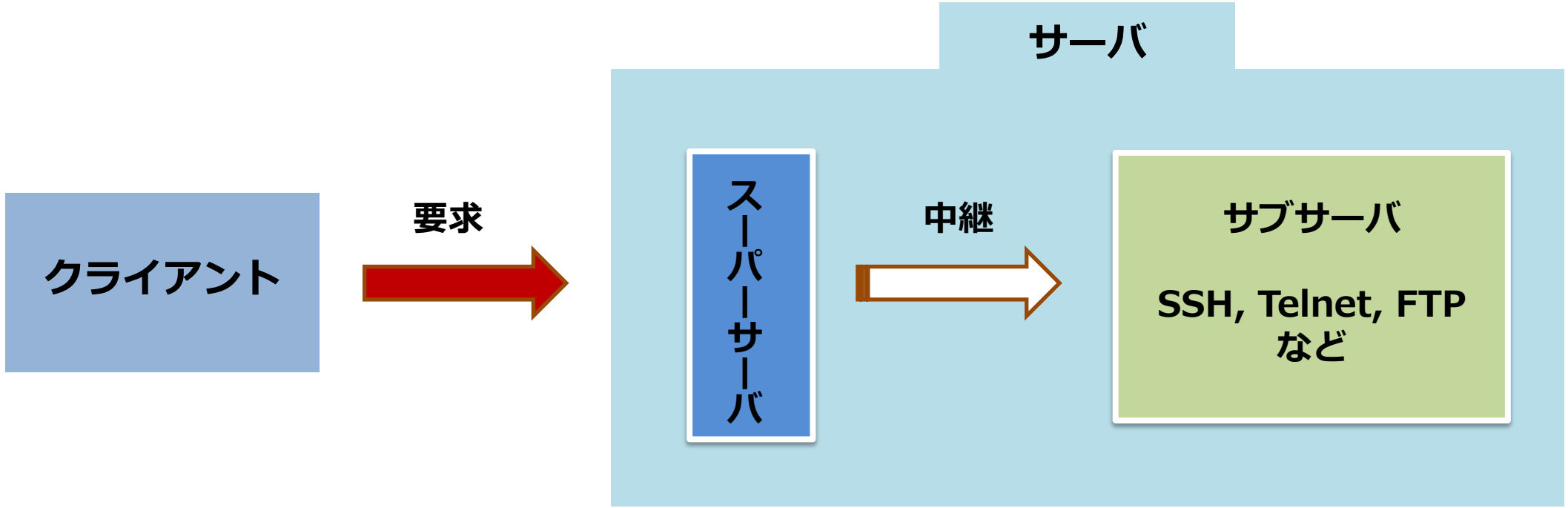
★inetd・xinetd（スーパーサーバ型デーモン）について

inetdのセキュリティ機能を強化した拡張版がxinetdです。
このデーモンを起動するとシステム起動するたびにスーパーサーバが起動し、ネットワークのインターネット・サービス管理を提供します。

このサービスは必要な場合にのみ他のサービス（サブサーバー）を呼び出すことによって、インターネット サービスを内部的に提供しシステムの負荷を減らします。

■セキュリティ（ホストのセキュリティ設定）

★inetd・xinetd（スーパーサーバ型デーモン）について



■セキュリティ（ホストのセキュリティ設定）

★inetd・xinetd（スーパーサーバ型デーモン）について

◆スーパーサーバベースの主なサブサーバー一覧（参考）

サービス名	機 能	サービス名	機 能
comsat	ユーザーに着信メールを通知	talkd	talk コマンド用のサーバー機能を提供
ftpd	FTP プロトコル用のサーバー機能を提供	telnetd	TELNET プロトコル用のサーバー機能を提供
fingerd	finger コマンド用のサーバー機能を提供	tftpd	トリビアル・ファイル転送プロトコル用のサーバー機能を提供
rlogind	rlogin コマンド用のサーバー機能を提供	uucpd	BNU と TCP/IP 間の通信を処理
rexecd	rexec コマンド用のサーバー機能を提供		
rshd	リモート・コマンドを実行するためのサーバー機能を提供		

■セキュリティ（ホストのセキュリティ設定）

★inetd・xinetd（スーパーサーバ型デーモン）について

サービスには単独で動作するタイプ（スタンドアロン）のものと、スーパーサーバベースのものがあり、前者は起動スクリプトや systemctl コマンドを使って制御しますが、後者は構成ファイルで制御します。

・単独で動作するサービス → 起動スクリプト(/etc/init.d/サービス名)
systemctlコマンド

・スーパーサーバベースのサービス → /etc/inetd.conf ファイル
/etc/xinetd.d 配下の各ファイル

■セキュリティ（ホストのセキュリティ設定）

★inetd・xinetd（スーパーサーバ型デーモン）について

セキュリティ上好ましくないサービスや不要なサービスは、始めから起動しないよう設定が必要です。

設定変更にはいくつか方法がありますが、ここでは xinetd の環境下で サブサーバのサービスを無効化する方法をご紹介します。

※xinetd のパッケージがインストールされていない場合はあらかじめインストールしてから xinetd サービスを起動しておきます。

```
# yum -y install xinetd
# systemctl start xinetd
```

■セキュリティ（ホストのセキュリティ設定）

★例：tcpmuxサービスの無効化 → /etc/xinetd.d/tcpmux-server ファイルを編集

```
service tcpmux
{
        disable = yes
        id = tcpmux-server
        wait = no
        socket_type = stream
        user = root
        server =
}
```

disable = yes
とすることで無効化となる

ファイル編集後はサービス再起動

```
# systemctl restart xinetd
```

■セキュリティ（暗号化によるデータの保護）

＜主要な知識範囲＞

- 基本的なOpenSSH2クライアントの設定および利用方法
- OpenSSH2サーバのホストキーの役割について理解している
- **基本的なGnuPGの設定および利用方法**
- SSHポートトンネル（X11トンネルを含む）について理解している

■セキュリティ（暗号化によるデータの保護）

★GnuPG(GNU Privacy Guard＝GPG)とは

GnuPGとは、通信またはストレージを暗号化するためのGNUプロジェクトによるツールで
OpenPGP（RFC 4880）に準拠し、データの暗号化と署名に用いられます。
ファイルの暗号化に利用する場合は、コマンド「gpg --gen-key」を実行し、鍵の種類、
有効期限、登録者名、メールアドレス、パスフレーズ、コメントを対話形式で入力し、
公開鍵と秘密鍵のペアキーを作成する必要があります。

■セキュリティ（暗号化によるデータの保護）

★gpgコマンド – GPGを利用したペアキー作成など

gpgコマンドは各種オプションによりいろいろな機能があります。

◆主なgpgコマンドオプション一覧

コマンド	機 能	備考
gpg --gen-key	公開鍵と秘密鍵のペアキー作成	対話形式により登録者名やパスフレーズなどを入力 公開鍵　~/.gnupg/pubring.gpg 秘密鍵　~/.gnupg/secring.gpg
gpg --list-keys	鍵の一覧表示	
gpg -o pubkey -a --export 登録者名	公開鍵pubkey のファイル出力	バイナリファイルからテキストファイルへ変換される
gpg --import pubkey	受け取った公開鍵pubkey をキーリングにインポート	
gpg -e -a -r 登録者名 ファイル名	ファイルの暗号化	ファイル名の末尾に「.asc」が付加されたファイルが作成される
gpg 暗号化ファイル名	ファイルの復号化	

ご清聴ありがとうございました