1. ファイル・ディレクトリの操作と管理

ファイルの所有者とパーミッション/基本的なファイル管理の実行/ハードリンクとシンボリックリンク/ファイルの配置と検索

2. GNUとUnixのコマンド

コマンドラインの操作/フィルタを使ったテキストストリームの処理/基本的なファイル管理の実行/ストリーム、パイプ、リダイレクトの使用/正規表現を使用したテキストファイルの検索/エディタを使った基本的なファイル編集の実行

3. Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用/ブートプロセスとsystemd/Linuxのインストール、起動、接続、切断と停止/プロセスの生成、監視、終了/デスクトップ環境の利用

4. リポジトリとパッケージ管理

apt コマンドによるパッケージ管理/Debianパッケージ管理/yumコマンドによるパッケージ管理/RPMパッケージ管理

5. ハードウェア、ディスク、パーティション、ファイルシステム

ハードウェアの基礎知識と設定/ハードディスクのレイアウトとパーティション/ファイルシステムの作成と管理、マウント

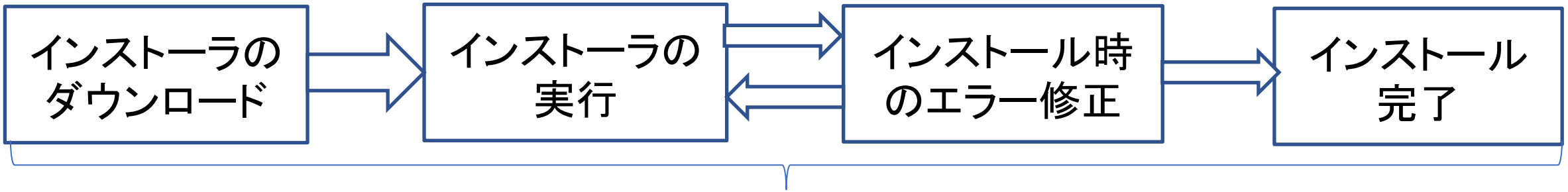
ここではDockerについて学びます。Dockerは、仕組みが複雑でとっつきずらいため、なるべく分かりやすくご説明したいと考えています。

docker-compose などのコンテナを管理するためのより応用的なツールはここではご紹介しませんが、Dockerの基本的な概念はご理解いただけるようにご説明します

なぜDockerを使うのか？

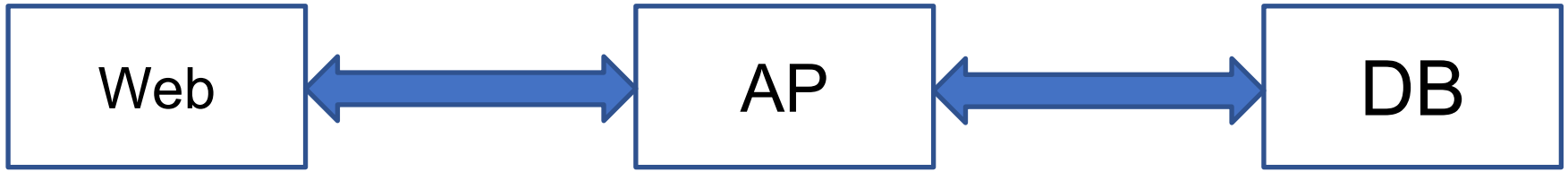
なぜDockerを使うのか？
（Dockerを利用するメリット）

1. Dockerを利用すると、OSを含むソフトウェアをインストールして仮想的な環境を構築し、すぐに利用することができる



このプロセスをパッケージ化してセットで実行して仮想環境構築を簡素化する

2. 複雑なサーバ構成を1つのサーバ上で実現できる



Dockerとは何か？

Dockerとは何か？

Dockerには、Docker Ecosystem(生態系)と呼ばれる複数のソフトウェアがあり、**コンテナ**を作成する。これらソフトウェアをまとめてDockerと呼ぶ

Docker Client (Dockerを利用するためのコマンドを提供する)	Docker Server (コンテナの作成、削除、起動を行うプログラム)	Docker images (コンテナを立ち上げるための元になるファイル)
Docker Hub (Dockerイメージを共有するためのリポジトリ)	Docker Compose (複数のDockerコンテナを管理するプログラム)	Docker Machine (仮想環境上のDockerをインストールするツール)

Dockerコンテナとは何か？
Dockerイメージとの違い

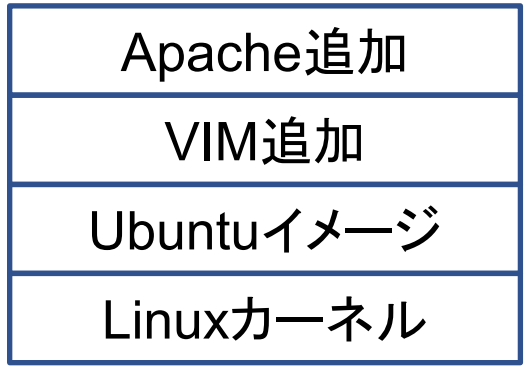
Dockerコンテナとは何か？Dockerイメージとの違い

Dockerイメージ

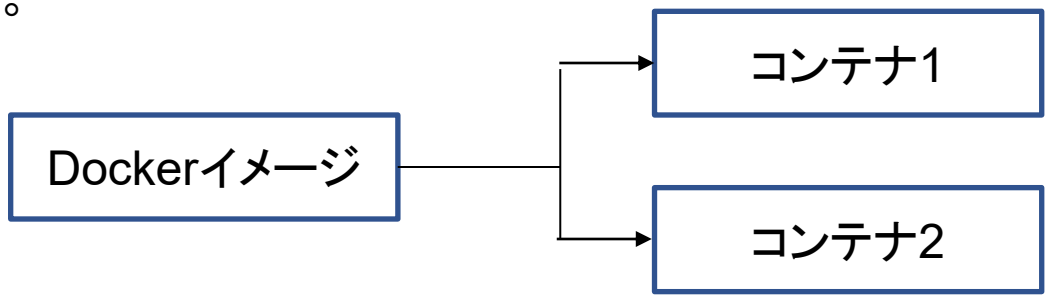
・**読み取り専用**のファイル集で、コンテナを立ち上げるためのソースコード、ライブラリ、ツールなどが含まれる
・イメージ自体を修正して保存することはできない。イメージを修正して新たなイメージとして保存する
・イメージはimage layerという階層構造になっており、イメージの内容を修正したら階層のトップに加える

Dockerコンテナ

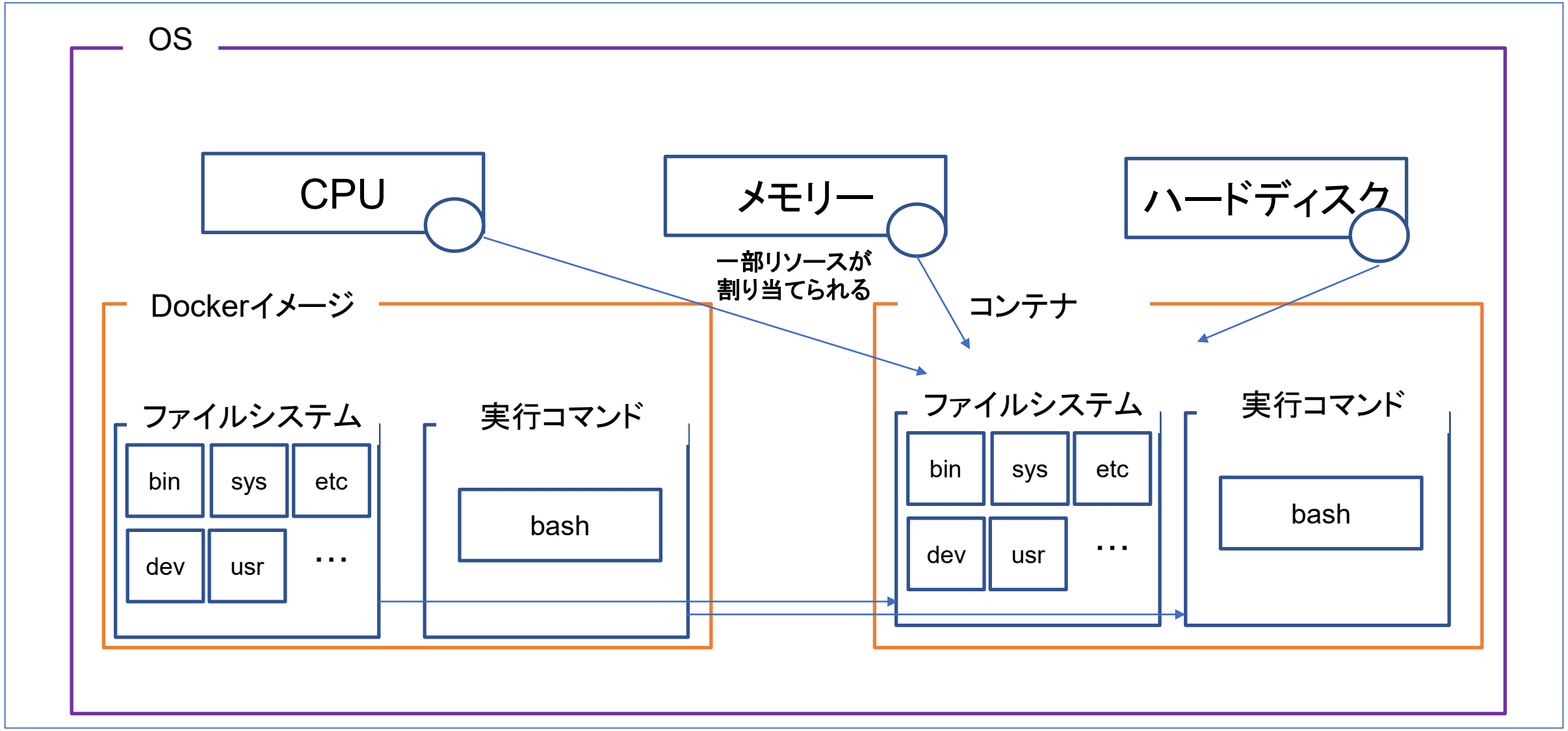
・Dockerイメージはテンプレートであるのに対して、コンテナはイメージを実態化したもの。イメージと異なり書き込んだり編集することができる
・OS全体から独立して立ち上げることのできる仮想環境
・Dockerイメージを元にして作成されるため、Dockerイメージがないと起動できない。



イメージの階層構造



Dockerコンテナとは何か？Dockerイメージとの違い



1.　Linuxのインストールと仮想マシン・コンテナの利用

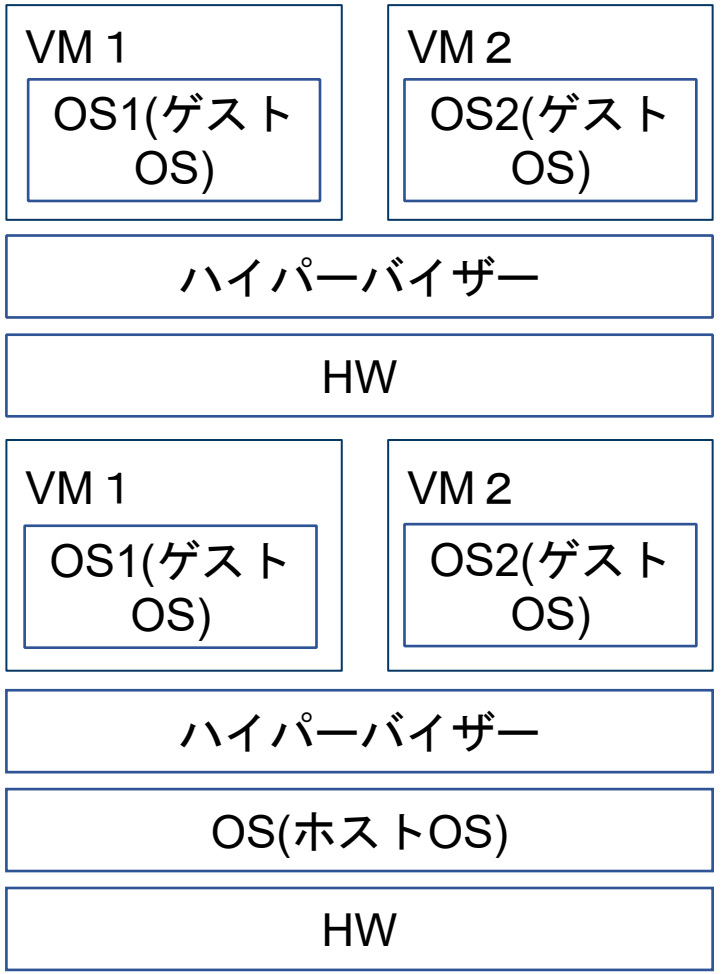
仮想マシン・コンテナの概念と利用

覚えること: カーネルとハイパーバイザー、ホストOSとコンテナ、仮想マシンの起動(virsh)、仮想マシンへのログイン、コンテナの起動と停止(docker)

カーネルとは・・・OSの中核となる部分。OS上で動作しているプログラムを管理したり、CPU、メモリなどのハードウェア資源を有効に管理したり、プログラムを優先度に従って切り替えたりする。

ハイパーバイザーとは・・・コンピュータの仮想化技術を用いてOS上に仮想マシンを作成するためのプログラム

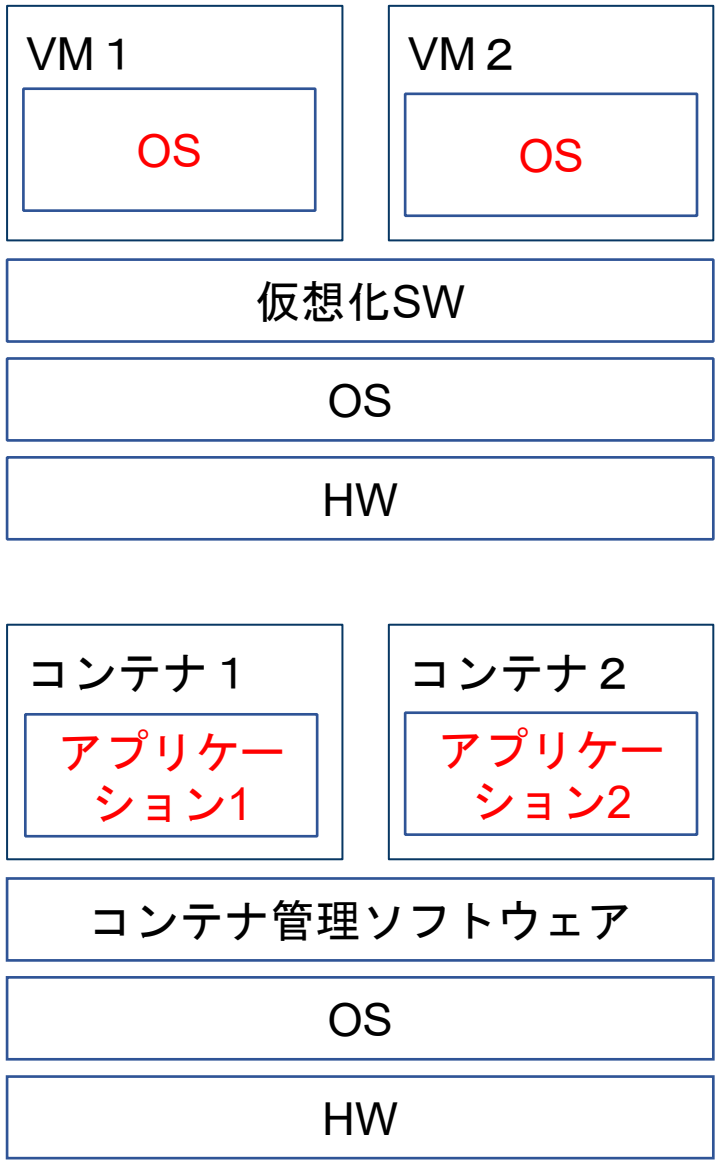
ネイティブハイパーバイザー
→ハードウェア上にハイパーバイザーが直接動作して、様々なOSがハイパーバイザー上に動作する(右図上)
例) vSphere, Hyper-V, KVMなど



1.　Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用

・ホスト型仮想化は、ホストOSの上に新たなOS(ゲストOS)を仮想的に作成する(Vmware, VirtualBox)



コンテナ型仮想化とは何か

・コンテナ型仮想化は、ゲストOSは必要なくホストOSのコア部分(カーネル)を共有して使用する
（原則、LinuxOS上でWindowsコンテナを使用することはできず、WindowsOS上でLinuxコンテナを使用することはできない）

Dockerとは

・コンテナ型の仮想環境を構築できるコンテナ管理ソフトウェア
・様々な環境を再現することができる
・Go言語で記述されている

1. Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用

覚えること: カーネルとハイパーバイザー、ホストOSとコンテナ、仮想マシンの起動(virsh)、仮想マシンへのログイン、コンテナの起動と停止(docker)

ホスト型仮想化とコンテナ型仮想化の違い

	メリット	デメリット
ホスト型仮想化	• 自由にOS指定して環境構成をインストールできる • 仮想環境ごとに環境が明確に分離されている	• OSを仮想環境ごとに作成するためリソースの使用量が大きい • 仮想環境のバージョンなどの変更に柔軟に対応できない
コンテナ型仮想化	• OSを共有するためリソース使用量を節約できる • バージョンの変化などに応じて、仮想環境を容易に変更できる	• ホストOSとは異なるOSの仮想環境を構築できない • OSは共通のため、仮想環境ごとの分離は明確でない

1.　Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用

覚えること: カーネルとハイパーバイザー、ホストOSとコンテナ、仮想マシンの起動(virsh)、仮想マシンへのログイン、コンテナの起動と停止(docker)

CentOSにdockerエンジンをインストール

yum install -y yum-utils　**# yum-config-managerをインストール**

yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo

yum-config-manager --enable docker-ce-nightly # コミュニティエディションのインストールを有効化

 # インストール対象のリポジトリにdockerエンジンを追加

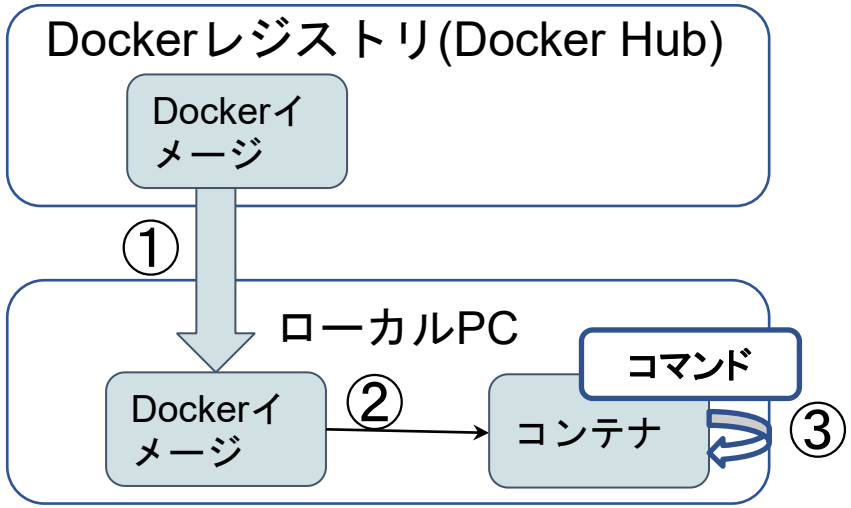
yum install -y docker-ce docker-ce-cli containerd.io　**# dockerエンジンのインストール**

1.　Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用

覚えること: カーネルとハイパーバイザー、ホストOSとコンテナ、仮想マシンの起動(virsh)、仮想マシンへのログイン、コンテナの起動と停止(docker)

docker pull: イメージの取得をする　docker rmi イメージ名: 取得したイメージの削除
docker create: コンテナの作成する(作成する際にコマンドを付ける)
docker start: コンテナの起動する(-iをつけるとコンテナに接続する)
docker run: 上の3つを一度に実行する(docker pull → docker create → docker start)
Dockerイメージ: コンテナの元となるデータが入ったファイル。Docker HUB(https;//hub.docker.com)上からイメージを取得することができる。
*) Dockerイメージを管理するサイトは、**Dockerレジストリ**と呼ばれる



コンテナ実行までの手順
①. ローカル上にイメージがない場合は、Dockerレジストリ(Docker Hub)からイメージを取得する(docker pull)
②. 取得したイメージを用いて、ローカルPC上にコンテナを作成する(コンテナ内で実行するコマンドを指定)(docker create)
③. コンテナを起動する(docker start)

1.　Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用

覚えること: カーネルとハイパーバイザー、ホストOSとコンテナ、仮想マシンの起動(virsh)、仮想マシンへのログイン、コンテナの起動と停止(docker)

docker run: イメージを取得してコンテナを立ち上げる
docker run = docker pull + docker create + docker start
docker run イメージ名 ○○（コンテナ内で呼び出すコマンドを指定する）
docker run –it: -i(ターミナルから入力を行うことができる), -t(実行結果を分かりやすく表示する)

コンテナから抜け出す場合
exit: コンテナを停止して抜け出す
ctrl+P → ctrl+Q: コンテナを停止せずに抜け出す

docker run –d: コンテナをバックグラウンドで実行する
docker tag イメージ名 新しいイメージ名: イメージを別名で新しく作成する
docker rmi (-f) イメージ名(イメージID): ローカル上に存在するDockerイメージを削除する(-を付けた場合は強制削除)
docker commit コンテナID イメージ名(:タグ): コンテナから新しいイメージを作成する

1.　Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用

覚えること: カーネルとハイパーバイザー、ホストOSとコンテナ、仮想マシンの起動(virsh)、仮想マシンへのログイン、コンテナの起動と停止(docker)

docker images: ダウンロードしたDockerイメージを確認する
docker inspect: イメージの詳細情報を表示する
docker ps: 動作中のコンテナを表示する
docker ps –a(--all): 終了したコンテナも含めて、コンテナを全て表示する
docker rm コンテナID: 指定したコンテナIDのコンテナを削除する
docker start コンテナID or コンテナ名: コンテナの開始
docker stop コンテナID or コンテナ名: 動作中のコンテナを終了させる(SIGTERM)
docker kill コンテナID or コンテナ名: 動作中のコンテナを終了させる(SIGKILL)

docker build –t タグ名 . : 現在のフォルダ上の Dockerfileからイメージをビルドする
-f ファイル名: Dockerfile以外のファイルを指定する

docker system prune: コンテナ全削除
docker image prune: イメージ全削除

Dockerfileの記述内容

Dockerfile・・・イメージとコンテナをどのように構築するのかを定義した設定ファイル

FROM: ベースとなるイメージを指定する

RUN: イメージからコンテナを作成してコマンドを実行する
（ソフトウェアのインストールなどを行う）

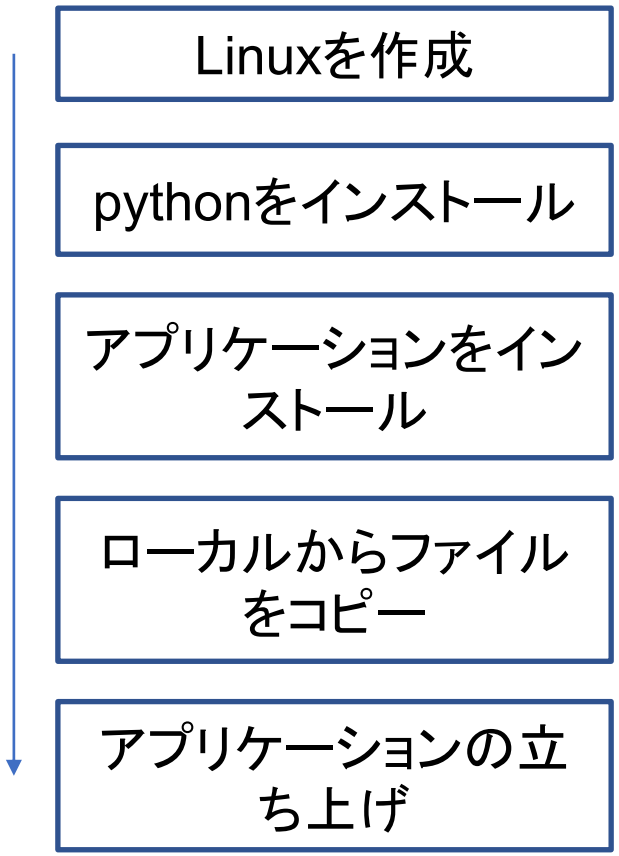
COPY: ローカルからファイルをコンテナ内にコピーする

WORKDIR: コンテナでの作業用のディレクトリを指定

ENTRYPOINT:コンテナ立ち上げ時に実行されるコマンドを指定する
(デフォルトのエントリーポイントは/bin/sh –c(後続のシェルを実行する))

CMD: コンテナ作成時にエントリーポイントに渡す引数、またはコマンドを指定する

Pythonのアプリケーションを立ち上げる場合



1.　Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用

覚えること: カーネルとハイパーバイザー、ホストOSとコンテナ、仮想マシンの起動(virsh)、仮想マシンへのログイン、コンテナの起動と停止(docker)

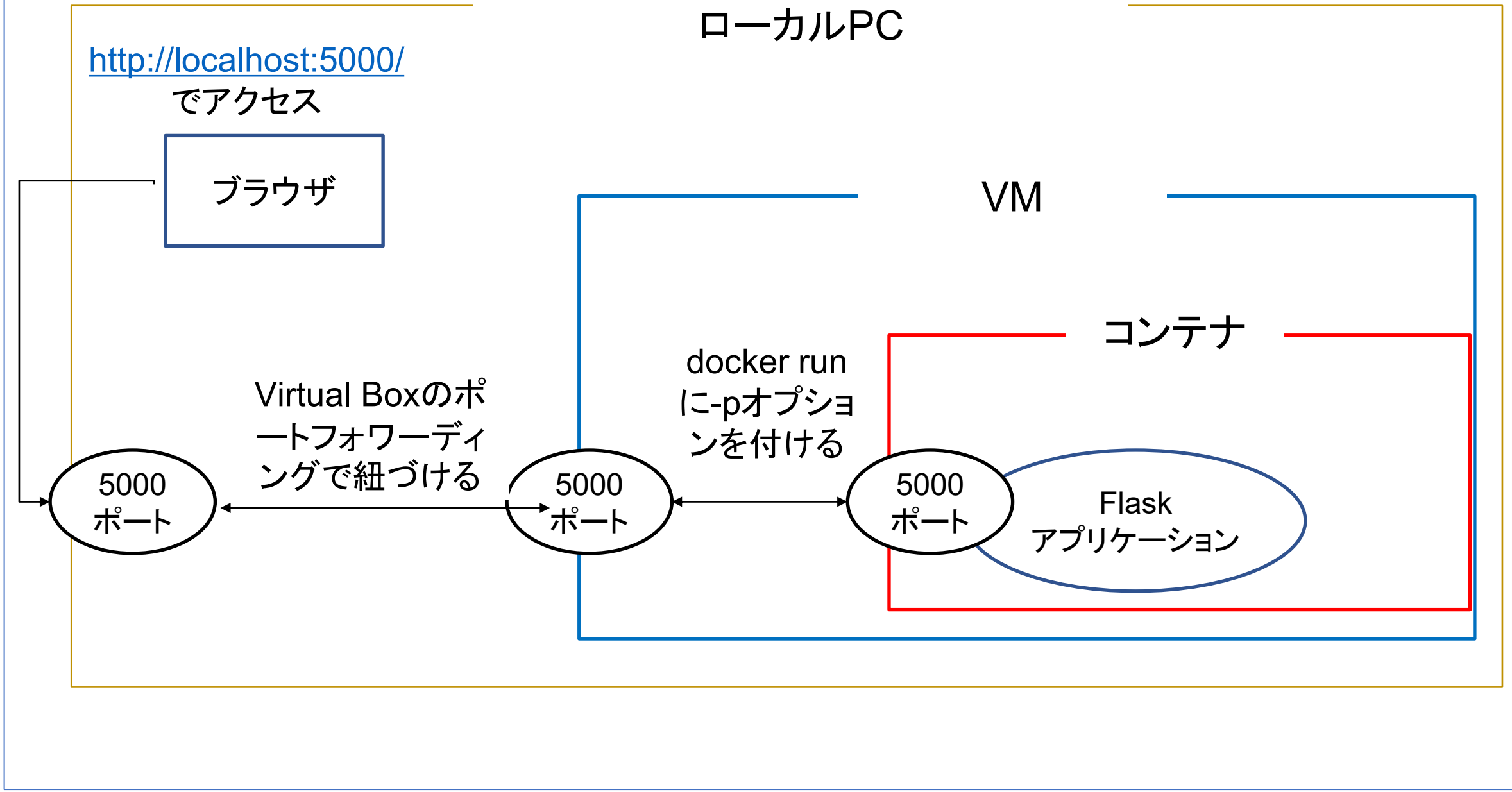
virt-manager	GUIで利用できる仮想マシン管理用のソフト。
virt-install	CUIでの仮想マシン作成用のコマンド。
libvirt	仮想化機構(KVMなど)と連携するライブラリ。

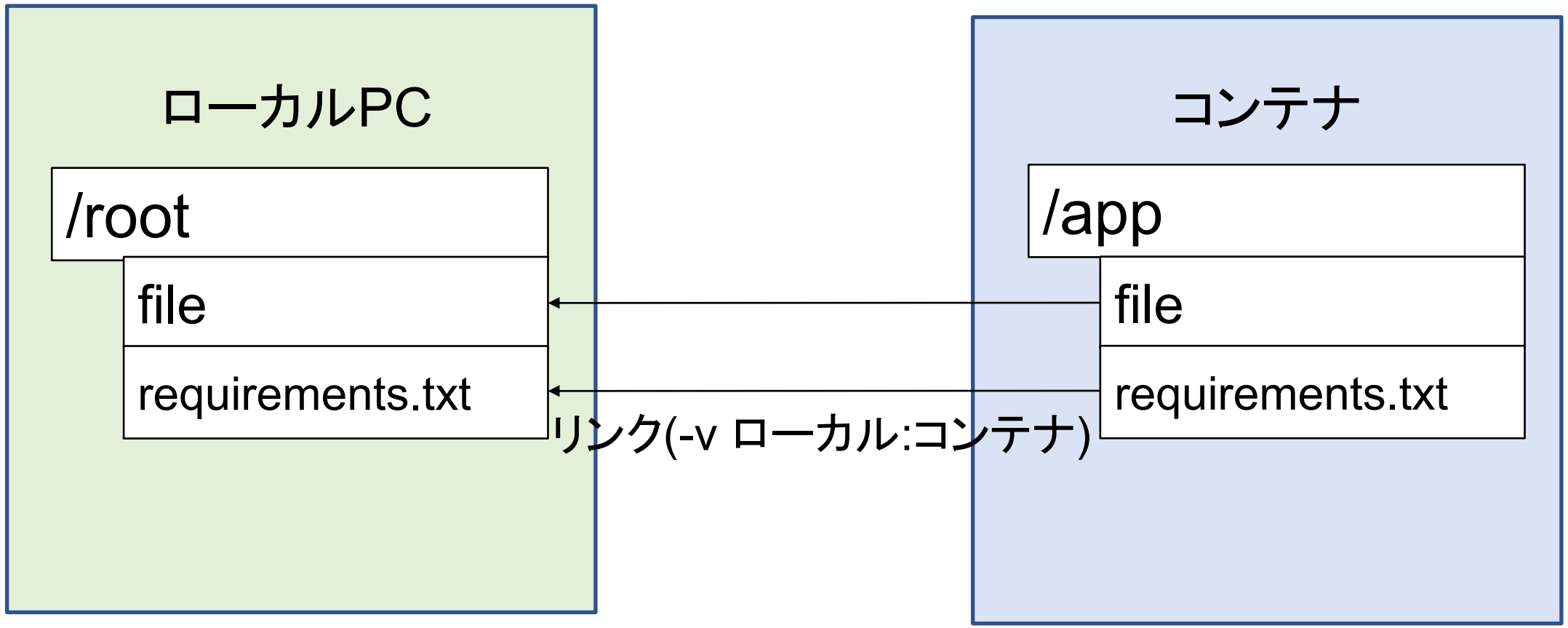
yum install virt-manager
yum install libvirt virt-install

virshコマンドは、仮想マシンを管理するコマンドで、仕様頻度は低く演習では扱うのは難しいため、演習では割愛します。試験向けにこちらのオプションを確認頂けると幸いです

virsh: 仮想マシンとハイパーバイザーのリソースを管理するコマンド
virsh list: ドメインの一覧を表示
virsh start VM: 仮想マシンの起動
virsh console VM: virshコマンドを用いてVMに接続する。
virsh shutdown VM: 仮想マシンを停止する。
virsh destroy VM: 仮想マシンを削除する。
virsh dominfo VM: 仮想マシンのドメインに関する基本的な情報を表示する
virsh domid VM: 仮想マシンのドメインIDを表示する
virsh domstate VM: 仮想マシンの実行状態（シャットオフ、実行中などの状態）

参考資料


ローカルPC
http://localhost:5000/
でアクセス
ブラウザ
VM
コンテナ
docker run
に-pオプショ
ンを付ける
Virtual Boxのポ
ートフォワーディ
ングで紐づける
5000
ポート
5000
ポート
5000
ポート
Flask
アプリケーション


ローカルPC
/root
file
requirements.txt
コンテナ
/app
file
requirements.txt
リンク(-v ローカル:コンテナ)

1. ファイル・ディレクトリの操作と管理

ファイルの所有者とパーミッション/基本的なファイル管理の実行/ハードリンクとシンボリックリンク/ファイルの配置と検索

2. GNUとUnixのコマンド

コマンドラインの操作/フィルタを使ったテキストストリームの処理/基本的なファイル管理の実行/ストリーム、パイプ、リダイレクトの使用/正規表現を使用したテキストファイルの検索/エディタを使った基本的なファイル編集の実行

3. Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用/ブートプロセスとsystemd/ Linuxのインストール、起動、接続、切断と停止/プロセスの生成、監視、終了/デスクトップ環境の利用

4. リポジトリとパッケージ管理

apt コマンドによるパッケージ管理/Debianパッケージ管理/yumコマンドによるパッケージ管理/RPMパッケージ管理

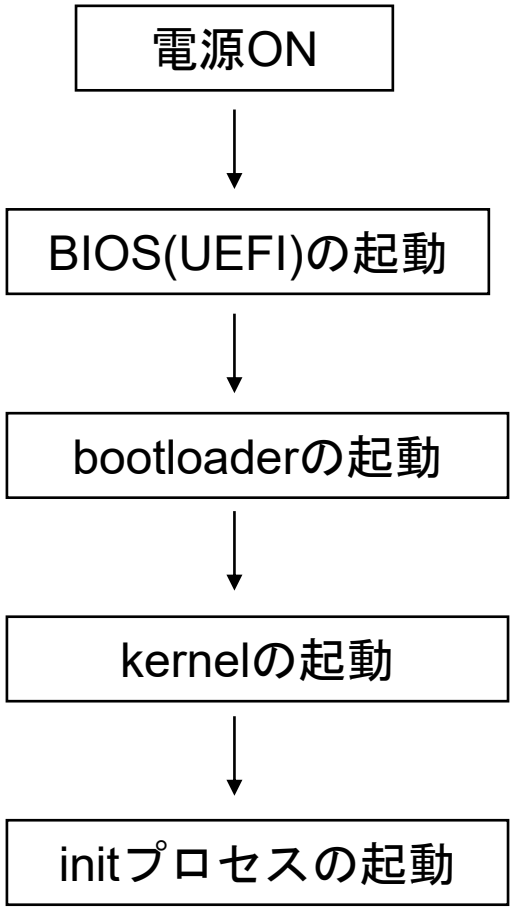
5. ハードウェア、ディスク、パーティション、ファイルシステム

ハードウェアの基礎知識と設定/ハードディスクのレイアウトとパーティション/ファイルシステムの作成と管理、マウント

1.　Linuxのインストールと仮想マシン・コンテナの利用

ブートプロセスとsystemd

覚えること: デフォルトのブートターゲットを設定する(systemd, systemctl), ブートターゲット（シングルユーザモードを含む）を変更する(systemctl)



- ハードウェアの初期化・設定
- メモリのチェック
- 起動するデバイスの設定・チェック(OS, ディスク等)
- 起動デバイスに格納されたbootloaderの実行

- GRAB or GRAB2を使用
- OS(kernel) をメモリにロードする

- OS(カーネル)はメモリを初期化し、一時的なファイル(initramfs)をマウントする。Initramfsには、システムにアクセスするための最低限のファイルが入っている

- ランレベルに応じて、/etc/init.dに格納されているプロセスを起動する

1.　Linuxのインストールと仮想マシン・コンテナの利用

ブートプロセスとsystemd

覚えること: デフォルトのブーターゲットを設定する(systemd, systemctl), ブーターゲット（シングルユーザモードを含む）を変更する(systemctl)

dmesg: システム起動時のカーネルが出力するメッセージの内容を表示する。

【initプロセスの種類】

SysVinit	ランレベルに応じてサービスを起動。並列起動できずに起動するのに時間がかかる
Upstart	サービスの起動準備が整ったらイベントを受け取り、並列にサービスを起動する。
systemd	細かい制御ができ、必要なサービスのみ並列に高速に起動できる。

ランレベル: initプロセスの動作モード

0	システムの停止(poweroff.target)
1	シングルユーザモード(rescue.target)
2, 3	マルチユーザモード(multi-user.target)

4	未使用
5	GUIモード(graphical.target)
6	システムの再起動(reboot.target)

init, telinit: ランレベルを変更できる
runlevel: 前回と今回のランレベルを表示する

1. Linuxのインストールと仮想マシン・コンテナの利用

ブートプロセスとsystemd

覚えること: デフォルトのブーターゲットを設定する(systemd, systemctl), ブーターゲット（シングルユーザモードを含む）を変更する(systemctl)

systemctl: systemdで制御しているサービスを管理するコマンド
systemctl オプション サービス名

start	サービス起動
stop	サービス停止
restart	サービス再起動
reboot	システム再起動

poweroff	シャットダウン
get-default	現在のデフォルトのブーターゲットを表示
set-default	現在のデフォルトのブーターゲットを変更
isolate	現在のブーターゲットを変更

ブーターゲット一覧
0: システムの停止(poweroff.target)、1: シングルユーザモード(rescue.target)、2,3: マルチユーザモード(multi-user.target)、4: 未使用、5: GUIモード(graphical.target)、6システムの再起動(reboot.target)

デフォルトのランレベルを変更するには2つの方法がある
1. systemctl set-default name.targetを実行する
2. default.targetファイルを作成してシンボリックリンクを作成する方法

wall: 全ユーザにメッセージを送る

1. ファイル・ディレクトリの操作と管理

ファイルの所有者とパーミッション/基本的なファイル管理の実行/ハードリンクとシンボリックリンク/ファイルの配置と検索

2. GNUとUnixのコマンド

コマンドラインの操作/フィルタを使ったテキストストリームの処理/基本的なファイル管理の実行/ストリーム、パイプ、リダイレクトの使用/正規表現を使用したテキストファイルの検索/エディタを使った基本的なファイル編集の実行

3. Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用/ブートプロセスとsystemd/**Linuxのインストール、起動、接続、切断と停止**/プロセスの生成、監視、終了/デスクトップ環境の利用

4. リポジトリとパッケージ管理

apt コマンドによるパッケージ管理/Debianパッケージ管理/yumコマンドによるパッケージ管理/RPMパッケージ管理

5. ハードウェア、ディスク、パーティション、ファイルシステム

ハードウェアの基礎知識と設定/ハードディスクのレイアウトとパーティション/ファイルシステムの作成と管理、マウント

1.　Linuxのインストールと仮想マシン・コンテナの利用

Linuxのインストール、起動、接続、切断と停止

覚えること: UEFI/BIOS、Linuxのインストール、Linuxシステム起動と停止、鍵を用いた接続

BIOS(Basic Input Output System: 入出力基本システム): キーボード、ハードディスクなどのデバイスを制御する基本的な制御プログラム。OS、アプリケーションはBIOSを利用して各ハードウェアにアクセスできる。
マザーボード上のBIOSはシステムBIOS。拡張カード上のBIOSは拡張BIOS

*) PCを起動する際には、まず、BIOSが起動されてOSをディスクに読み込んでから起動されます。

UEFI(Unified Extensible Firmware Interface): BIOSの後継規格。起動ドライブの容量制限（BIOSは2TB）がなくなり、GUIでのセットアップ画面が利用できるなどの拡張がある。

*) BIOSとUEFIを併せて、総称としてBIOSと呼ばれることもあります

1.　Linuxのインストールと仮想マシン・コンテナの利用

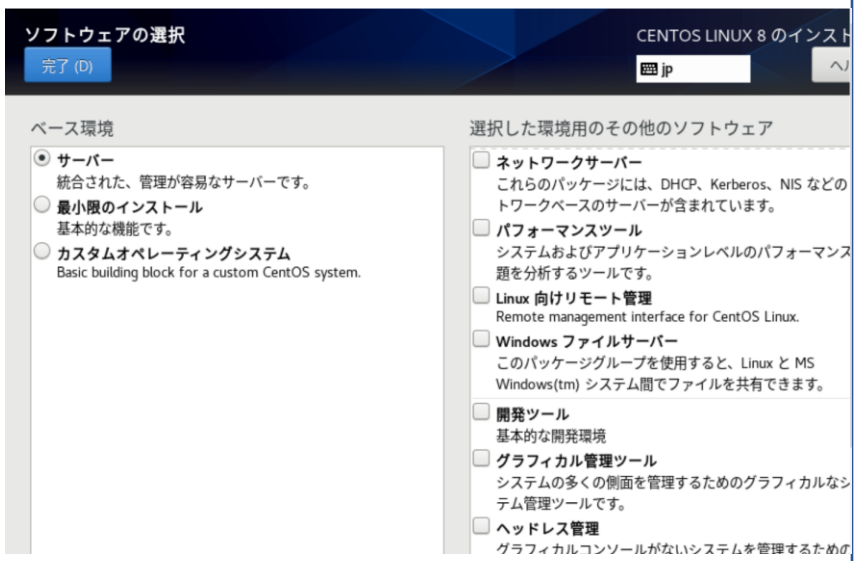
Linuxのインストール、起動、接続、切断と停止

覚えること: UEFI/BIOS、Linuxのインストール、Linuxシステム起動と停止、鍵を用いた接続

Linuxのインストール: Linuxをインストールする際に以下のような設定ができます。

・OSで使用する言語
・キーボードの言語
・インストール先ハードディスク
・タイムゾーン

・ネットワークホスト名
・ユーザ情報、rootユーザ情報
・インストールするソフトウェアパッケージ



1.　Linuxのインストールと仮想マシン・コンテナの利用

Linuxのインストール、起動、接続、切断と停止

覚えること: UEFI/BIOS、Linuxのインストール、Linuxシステム起動と停止、鍵を用いた接続

Linuxの停止をするshutdownコマンド
rootユーザで実行します。サーバ、シャットダウンをコマンドですることは滅多にないですが、ローカルの環境とかだと、実行するときもあります。
shutdown オプション 時間 メッセージ

【オプション】
-P or --poweroff システムを直ちに停止する(何もオプションを指定しない場合のデフォルト)
-h シャットダウンしてプロセスを全て落とした後に電源を落とす
-r or --reboot シャットダウン後にシステムを再起動(-rはreboot(再起動))
-f 次回起動時にfsck(ファイルシステムの整合性チェックと可能なら修復)を実行しない
-F 次回起動時にfsckを実行する
-k シャットダウンせずにメッセージのみユーザに告知
-c 実行中のシャットダウンをキャンセル

【時間】
now 今すぐ実行
HH:MM HH:時MM分に実行
+MIN MIN分後に実行

1.　Linuxのインストールと仮想マシン・コンテナの利用

Linuxのインストール、起動、接続、切断と停止

覚えること: UEFI/BIOS、Linuxのインストール、Linuxシステム起動と停止、鍵を用いた接続

Linuxの停止をするコマンド(reboot, poweroff, halt)

reboot: OSを再起動するコマンド(shutdown -r(--reboot) nowと同じ)

poweroff: OSを直ちに停止して電源を切るコマンド(shutdown -P(--poweroff) nowと同じ)

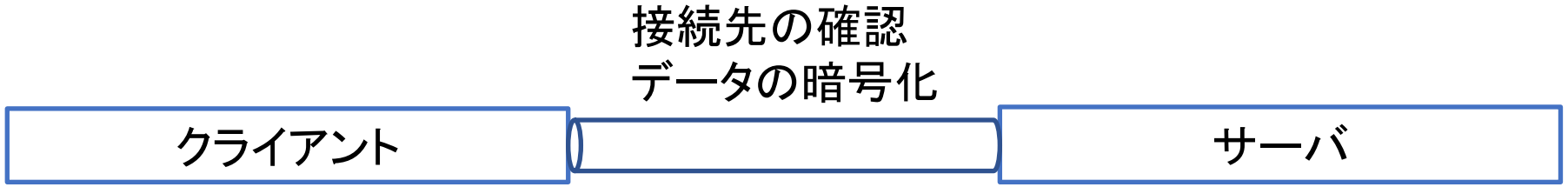
halt: OSをシャットダウンして、電源を**切れる状態にする**コマンド（プロセスは停止しているが電源は落ちていない状態）(shutdown -H(--halt) nowと同じ)

1.　Linuxのインストールと仮想マシン・コンテナの利用

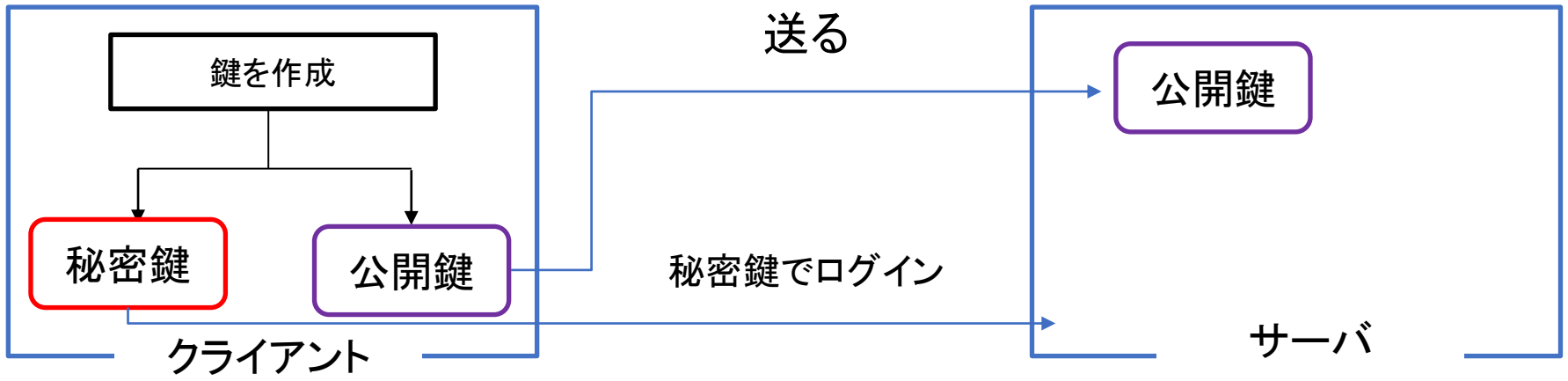
Linuxのインストール、起動、接続、切断と停止

覚えること: UEFI/BIOS、Linuxのインストール、Linuxシステム起動と停止、鍵を用いた接続

SSH



~/.ssh/known_hosts: 一度もssh接続をしていないサーバにssh接続した際に接続したサーバの情報が保存されるファイル
~/.ssh/authorized_keys: 公開鍵認証でクライアントの公開鍵を登録するファイル
~/.ssh/id_rsa: 公開鍵認証で利用する秘密鍵
~/.ssh/id_rsa.pub: 公開鍵認証で利用する公開鍵



1. ファイル・ディレクトリの操作と管理

ファイルの所有者とパーミッション/基本的なファイル管理の実行/ハードリンクとシンボリックリンク/ファイルの配置と検索

2. GNUとUnixのコマンド

コマンドラインの操作/フィルタを使ったテキストストリームの処理/基本的なファイル管理の実行/ストリーム、パイプ、リダイレクトの使用/正規表現を使用したテキストファイルの検索/エディタを使った基本的なファイル編集の実行

3. Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用/ブートプロセスとsystemd/ Linuxのインストール、起動、接続、切断と停止/**プロセスの生成、監視、終了**/デスクトップ環境の利用

4. リポジトリとパッケージ管理

apt コマンドによるパッケージ管理/Debianパッケージ管理/yumコマンドによるパッケージ管理/RPMパッケージ管理

5. ハードウェア、ディスク、パーティション、ファイルシステム

ハードウェアの基礎知識と設定/ハードディスクのレイアウトとパーティション/ファイルシステムの作成と管理、マウント

1.　Linuxのインストールと仮想マシン・コンテナの利用

プロセスの生成、監視、終了

用語: ジョブをフォアグラウンドやバックグラウンドで実行する(&, bg, fg, jobs), ログアウト後にも実行が継続されるようにプログラムにシグナルを送信する、活動中のプロセスを監視する(top, ps, pstree, uptime), プロセス群を選択し、並べ替えて表示する(pgrep), プロセスにシグナルを送信する(kill, pkill, killall)

コマンド &: コマンドをバックグラウンドプロセスとして実行

jobs: ログインユーザの実行中と停止中のジョブを表示するコマンド、
ctrl + z: 実行中のコマンドを停止する

bg: 一時停止中のジョブをバックグラウンドで再開する。
例) bg %1 とするとjobsで表示される1のコマンドをバックグラウンドで実行する

fg: バックグラウンド実行中のジョブをフォアグラウンドで実行。
例) fg %1 とするとjobsで表示される1のコマンドをフォアグラウンドで実行する

kill %〇:〇番目のジョブを削除する
kill -19 %〇:〇番目のジョブを停止する

1.　Linuxのインストールと仮想マシン・コンテナの利用

プロセスの生成、監視、終了

用語: ジョブをフォアグラウンドやバックグラウンドで実行する(&, bg, fg, jobs), ログアウト後にも実行が継続されるようにプログラムにシグナルを送信する, 活動中のプロセスを監視する(top, ps, pstree, uptime), プロセス群を選択し、並べ替えて表示する(pgrep), プロセスにシグナルを送信する(kill, pkill, killall)

top: 実行中のプロセスのCPU使用率などを継続で表示する
-d 間隔: 表示する間隔を指定
k: プロセスに対してkillする
q: topの終了
r: プロセスの優先度を変更

ps: 実行中のプロセスを表示する
-a: すべてのプロセスを表示(×-a, ○-A)
-f: プロセスをツリーで表示
-p プロセスID: プロセスを指定して表示
-u: プロセスを実行しているユーザの表示
-e: コマンドの後の環境変数などを表示する

pstree: 親プロセスとその親プロセスから派生した子のプロセスの階層構造を表示する
yum install psmiscでインストール

uptime: システムの稼働時間を表示する
①現在の時刻　②システムの稼働時間　③ログイン中のユーザ数　④ロードアベレージ(システムの負荷量)過去1分、過去5分、過去15分

1. Linuxのインストールと仮想マシン・コンテナの利用

プロセスの生成、監視、終了

用語: ジョブをフォアグラウンドやバックグラウンドで実行する(&, bg, fg, jobs), ログアウト後にも実行が継続されるようにプログラムにシグナルを送信する, 活動中のプロセスを監視する(top, ps, pstree, uptime), プロセス群を選択し、並べ替えて表示する(pgrep), プロセスにシグナルを送信する(kill, pkill, killall)

pgrep: プロセス名を指定してプロセスIDを検索する

-f: プロセス名のみだけでなく、コマンドライン全体が検索対象となる
-l: プロセスIDだけでなく、プロセス名も表示

kill: プロセスにシグナルを送信する

kill -1(-SIGHUP) プロセスID: 再起動
kill -2(-SIGINT) プロセスID: 割り込み(ctrl+C)
kill -9(-SIGKILL) プロセスID: 強制終了
kill -15(-SIGTERM) プロセスID: 終了
kill -19(-STOP): 一時停止

pkill: プロセス名を指定してプロセスにシグナルを送信する。
killall: プロセスの停止などシグナルを送れる。killと似ているがコマンド名を指定できる

nohup: ユーザがログアウトした後も、指定したコマンドの標準出力をバックグラウンドで継続して処理させ、$(HOME)/nohup.outファイルに出力する

1.　Linuxのインストールと仮想マシン・コンテナの利用

プロセスの生成、監視、終了

用語: ジョブをフォアグラウンドやバックグラウンドで実行する(&, bg, fg, jobs), ログアウト後にも実行が継続されるようにプログラムにシグナルを送信する, 活動中のプロセスを監視する(top, ps, pstree, uptime), プロセス群を選択し、並べ替えて表示する(pgrep), プロセスにシグナルを送信する(kill, pkill, killall)

tmux: ターミナル上に複数のターミナルを立ち上げる
(yum install tmuxでインストール)

tmux ls: tmuxで立ち上げたセッションの一覧を表示
tmux new-session –s ○○**:** ○○**という名前のセッションを**新たに立ち上げる
tmux kill-session –t ○○**:** セッション名を指定して削除する
tmux attach –t ○**:** セッションに再接続する。
tmux rename-session –t ○○**:** セッションの名前を変更する

【セッション立ち上げ後のコマンド】
ctrl + d ,exit: ウィンドウ、セッションを終了
ctrl + b → s: セッションの一覧を表示
ctrl + b → c: 新規ウィンドウの作成
ctrl + b → w: ウィンドウの一覧を表示
ctrl + b → &: セッションの破棄
ctrl + b → “: 上下に分割
ctrl + b → %: 左右に分割
ctrl + b → 十字キー: ウィンドウを移動する
ctrl + b → d:セッションを一時的に中断してメインに戻る

1. ファイル・ディレクトリの操作と管理

ファイルの所有者とパーミッション/基本的なファイル管理の実行/ハードリンクとシンボリックリンク/ファイルの配置と検索

2. GNUとUnixのコマンド

コマンドラインの操作/フィルタを使ったテキストストリームの処理/基本的なファイル管理の実行/ストリーム、パイプ、リダイレクトの使用/正規表現を使用したテキストファイルの検索/エディタを使った基本的なファイル編集の実行

3. Linuxのインストールと仮想マシン・コンテナの利用

仮想マシン・コンテナの概念と利用/ブートプロセスとsystemd/ Linuxのインストール、起動、接続、切断と停止/プロセスの生成、監視、終了/**デスクトップ環境の利用**

4. リポジトリとパッケージ管理

apt コマンドによるパッケージ管理/Debianパッケージ管理/yumコマンドによるパッケージ管理/RPMパッケージ管理

5. ハードウェア、ディスク、パーティション、ファイルシステム

ハードウェアの基礎知識と設定/ハードディスクのレイアウトとパーティション/ファイルシステムの作成と管理、マウント

1. Linuxのインストールと仮想マシン・コンテナの利用

デスクトップ環境の利用

用語: X Window System の構成要素についての基本的な理解と知識(startx, X サーバー, X クライアント, Display Manager, Window Manager, X Window System, 統合デスクトップ環境), X11 環境でのGUIをローカルおよびリモートで起動する(xauth, DISPLAY, ターミナルプログラム)

X11: X.Org Foundationが開発しているLinuxをGUIで使用するためのツール。X Window Systemと呼ばれる(Xとも呼ばれる)。

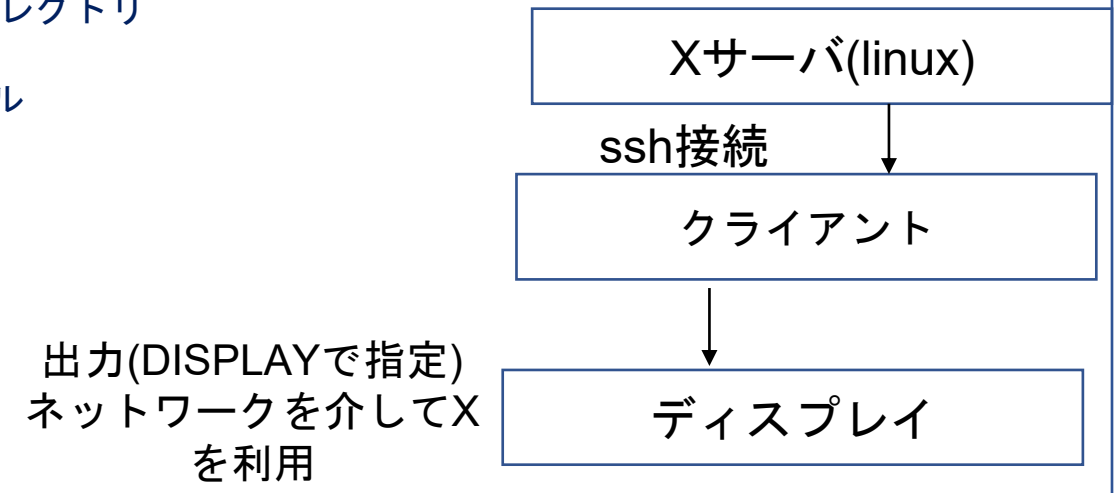
/etc/X11/xrog.conf: Xの設定ファイル。中身はSection “セクション名”~EndSectionで記載する
/etc/X11/xorg.conf.d/: Xのユーザ固有の設定ファイルを格納しているディレクトリ

/etc/ssh/sshd_config: sshd(ssh接続を待ち受けるプロセス)の設定ファイル
X11Forwarding yes　←x11転送を許可
X11DisplayOffset 10　←xサーバのディスプレイ番号
X11UseLocalhost : X11 をlocalhost のみ許可

~/.xsession-errors: Xが出力するエラーログファイル

startx: CLIモードでXを起動するコマンド

xhost: Xサーバの接続サーバの利用を許可するコマンド



DISPLAY: Xサーバのディスプレイの表示先を指定する環境変数。設定内容は、ディスプレイのホスト名:ディスプレイ番号.スクリーン番号

xwininfo: ディスプレイのウィンドウ位置、サイズなどの情報を表示するコマンド

xdpyinfo: ディスプレイ番号、スクリーン番号、解像度などのディスプレイ情報を表示するコマンド

1. Linuxのインストールと仮想マシン・コンテナの利用

デスクトップ環境の利用

用語: X Window System の構成要素についての基本的な理解と知識(startx, X サーバー, X クライアント, Display Manager, Window Manager, X Window System, 統合デスクトップ環境), X11 環境でのGUIをローカルおよびリモートで起動する(xauth, DISPLAY, ターミナルプログラム)

ディスプレイマネージャ: Linuxにグラフィカルなインターフェースを提供するX Windowシステムの1つ

【代表的なディスプレイマネージャ】

XDM: x標準のディスプレイマネージャ。設定ファイルは**/etc/X11/xdm/xdm-config**
Xaccess: XDMに接続するホストを設定。**Xresources**: ログイン画面を設定。**Xservers**: Xサーバとディスプレイを設定。**Xsession**: XDM起動後に動作するスクリプトを設定する。**Xsetup_0**: XDM起動時に動作するスクリプトを設定。

KDM: **KDE**(X Windowシステムで動作アプリケーション郡)で動作する代表的なディスプレイマネージャ。**/etc/X11/kdm**ディレクトリに設定ファイルがある。

GDM : **Gnome**(X windowシステムで動作するアプリケーション郡)の標準の代表的なディスプレイマネージャ。 **/etc/X11/gdm**ディレクトリに設定ファイルが格納されている

xfce:豪華な見た目と簡単な使用感を保ちながら、軽量・高速なデスクトップ環境を目指しているディスプレイマネージャ。

LightDM:軽量で高速、さらに拡張性に富むマルチデスクトップを目標としているディスプレイマネージャ。**/etc/lightdm**に設定ファイルを格納する

VNC: ヴァーチャルネットワークコンピューティング。ネットワーク上の離れたコンピュータを遠隔操作するプロトコル

RDP: リモートデスクトッププロトコル。サーバコンピュータの画面をクライアントに転送する