

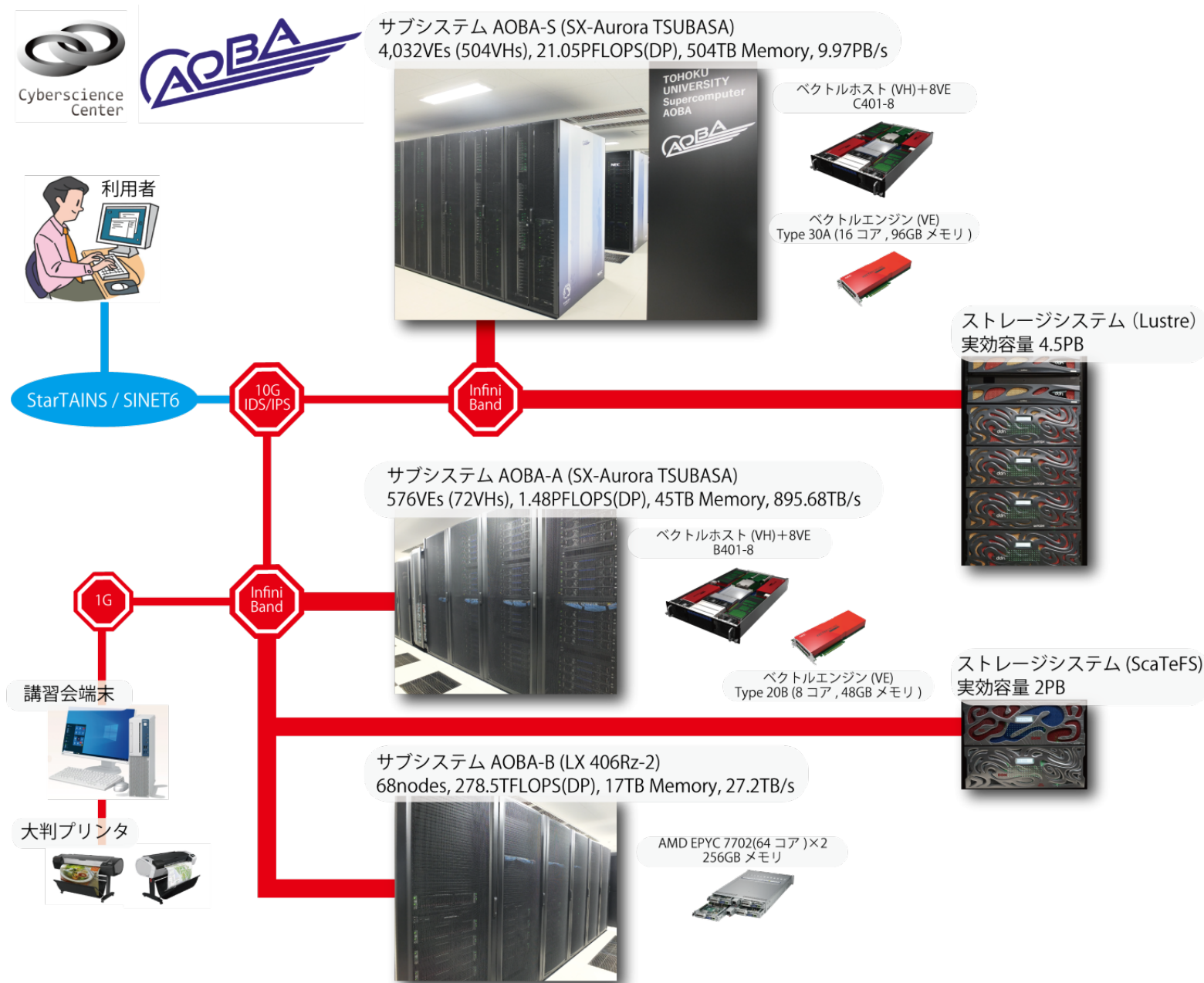


# 東北大学サイバーサイエンスセンター 講習会 はじめてのLinux



東北大学 情報部デジタルサービス支援課  
スーパーコンピューティングサポートユニット  
<https://www.ss.cc.tohoku.ac.jp/>

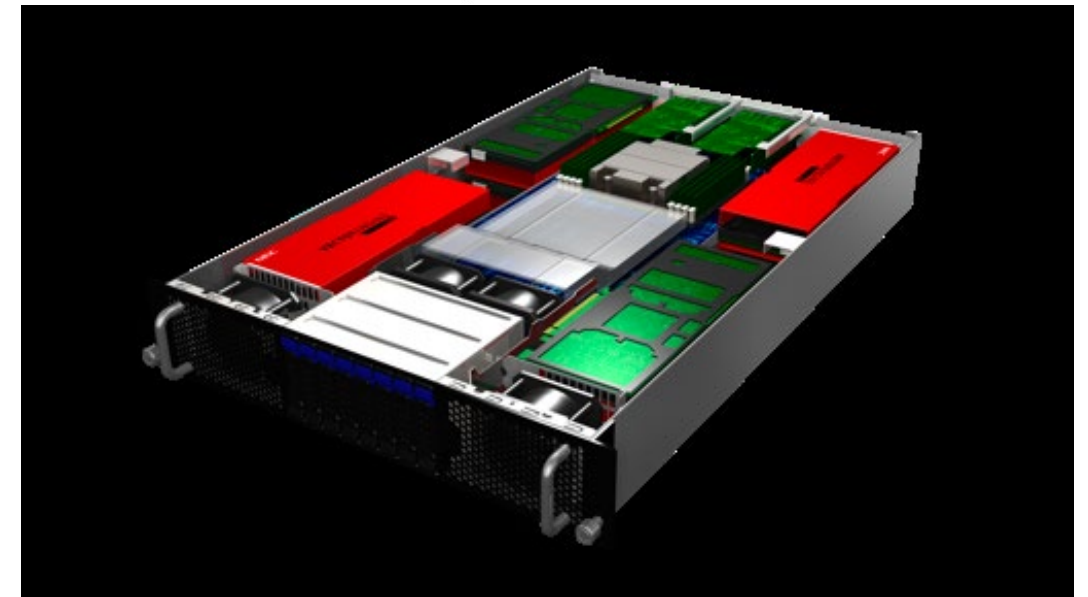
1. スーパーコンピュータAOBAの概要
2. Linuxとは
3. 基本的なLinuxコマンドと演習
4. X Window Systemについて
5. エディタ(nano)の簡単な使い方



- スーパーコンピュータ「AOBA」サブシステムAOBA-S、AOBA-A、AOBA-B  
**3種類の計算機(ベクトル型、スカラー型)を提供**
- ログインサーバ、フロントサーバ、ファイルサーバシステム  
**ログイン環境、大規模データの高速ファイルI/O環境を提供**

# 1. スーパーコンピュータAOBAの概要

## ○ サブシステムAOBA-S (SX-Aurora TSUBASA)

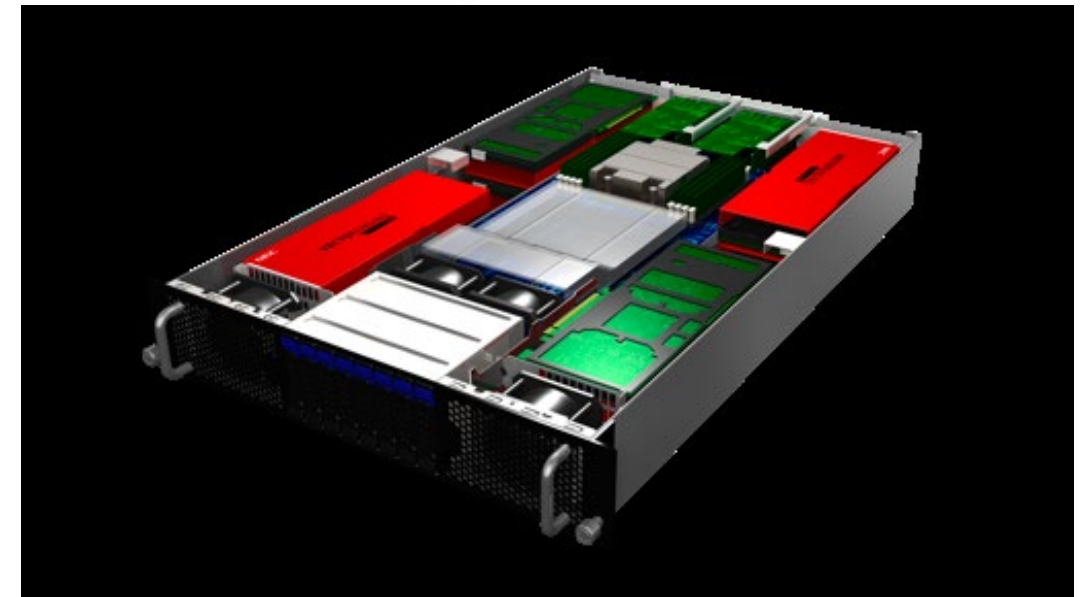


- 日本電気株式会社製
- ベクトルプロセッサ+X86/Linuxアーキテクチャ
- 基本システム単位は 1ベクトルホスト (VH) + 8ベクトルエンジン (VE)
- VHはOS処理, VE制御, I/O制御、VEは演算処理
- 主記憶容量は256 GB + 768 GB(96GB×8)
- 演算性能は2.5 TFLOPS + 39.28 TFLOPS (DP)
- センターでは504ノードを導入 (21.05PFLOPS、504TB)
  - 最大256ノードを利用した大規模並列実行が可能



# 1. スーパーコンピュータAOBAの概要

## ○ サブシステムAOBA-A (SX-Aurora TSUBASA)



- 日本電気株式会社製
- ベクトルプロセッサ+X86/Linuxアーキテクチャ
- 基本システム単位は 1ベクトルホスト (VH) + 8ベクトルエンジン (VE)
- VHはOS処理, VE制御, I/O制御、VEは演算処理
- 主記憶容量は256 GB + 384 GB(48GB×8)
- 演算性能は1.075 TFLOPS + 19.6 TFLOPS (DP)
- センターでは72ノードを導入 (1.48PFLOPS、45TB)
  - 最大32ノードを利用した大規模並列実行が可能

# 1. スーパーコンピュータAOBAの概要

## ○ サブシステムAOBA-B



- 日本電気株式会社製
- 1ノード（2CPU・128コア）あたり4.096TFLOPSの理論演算性能を持つEPYCプロセッサ
- 1ノードあたり256GBのメモリを搭載
- センターでは68ノードを導入（278.5TFLOPS、17TB）
  - ・ 最大16ノードを利用した大規模並列実行が可能
- Linux向けアプリケーション、オープンソフトウェアの環境構築と実行が容易

- オペレーティングシステム（OS）のひとつ
  - ー OSとは、コンピューターを動かすためのソフトウェアのこと
  - ー OSには、他にWindows, MacOS, iOS, Android OS などがある
  
- 特徴
  - ー マルチユーザ、マルチタスク
    - ・ 大勢が同時に使える
    - ・ 複数のプログラムを同時に実行出来る
    - ・ root（管理者）は何でも出来る
  - ー キャラクタユーザインターフェース（CUI）
    - ・ コマンドを入力して作業を行う（テキスト編集、ファイル操作、コンパイル、実行）
    - ・ グラフィカルユーザインターフェース（GUI）も利用可能
  - ー ネットワークとの親和性が高い

- Linuxが利用されている場面
  - ー 研究室のワークステーションや個人のパソコン
    - ・ MacOS、WindowsとLinuxのマルチブート、Virtual Box 上の仮想環境
  - ー 大規模科学計算システム、ネットワークサーバ
  - ー スマートフォン、ゲーム機、家電製品
- 研究、業務では
  - ー スパコン、演算サーバ、ワークステーションなどを使った数値計算
  - ー Webサーバ、メールサーバ、共有ディスクサーバなどの管理・運営





#### ○ Linuxマシンへの接続

##### ー ユーザの登録が必要

- ・ 事前にユーザ登録が必要（センターでは利用申請が必要）

<https://www.ss.cc.tohoku.ac.jp/apply-for-use/>

##### ー ユーザ名とパスワード認証、または鍵認証を行う

（センター計算機は公開鍵暗号方式のみ利用可能）

<https://www.ss.cc.tohoku.ac.jp/first-use/>

#### ○ ログイン（利用開始）

##### ー ローカル端末にログイン、SSHでネットワーク経由のログイン

#### ○ 作業

##### ー プログラム作成、実行、アプリケーション利用、設定変更など

#### ○ ログアウト（利用終了）

##### ー exitコマンドで切断

- SSHを利用したログイン (AOBA-A/B側への接続)
  - ー 端末アプリケーション (Powershell、Tera Term、Terminal、端末等)
    - ・ ログインした環境はログインサーバ、フロントエンドサーバのCUI環境

```
localhost $ ssh 利用者番号@login.cc.tohoku.ac.jp -i ~/.ssh/id_rsa
Login $ ssh 利用者番号@front.cc.tohoku.ac.jp
```
- SSHを利用したログイン (AOBA-S側への接続)
  - ー 端末アプリケーション (Powershell、Tera Term、Terminal、端末等)
    - ・ ログインした環境はログインサーバ、フロントエンドサーバのCUI環境

```
localhost $ ssh 利用者番号@sfront.cc.tohoku.ac.jp -i ~/.ssh/id_rsa
```
- SSHとX Window Systemを利用したログイン  
(研究室などのマシンからネットワーク経由でGUIアプリケーションを利用する場合)
  - ー 端末アプリケーションとX Window System環境 (Linuxからの利用がおすすめ)

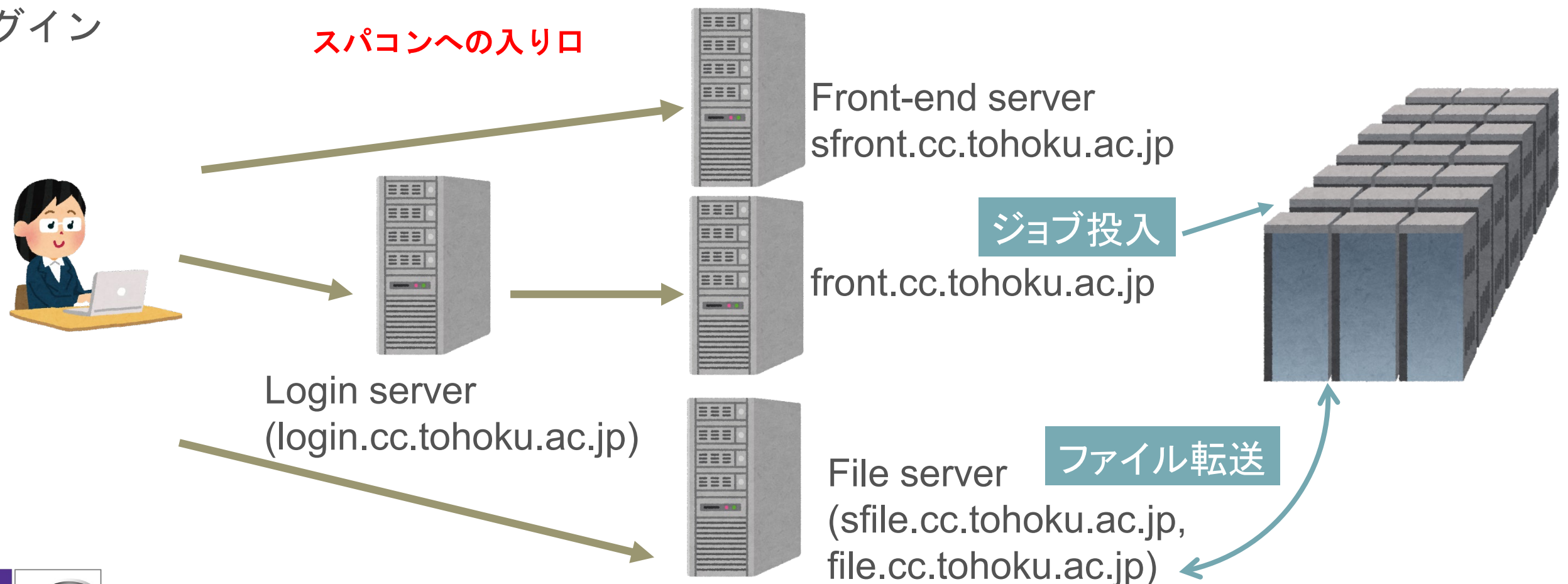
```
localhost $ ssh -X 利用者番号@login.cc.tohoku.ac.jp -i ~/.ssh/id_rsa
Login $ ssh -X 利用者番号@front.cc.tohoku.ac.jp
```
- X Window System を利用したログイン (端末機室、利用相談室のマシンからの利用)
  - ー Exceedの利用
    - ・ ログインした環境はフロントエンドサーバのGUI環境

- サイバーサイエンスセンターの計算機に（ログインサーバ、ファイルサーバ）ログインするには、初回だけ認証鍵ペアの作成が必要

<https://www.ss.cc.tohoku.ac.jp/first-use/> （利用方法の詳細）

- 利用者ポータルでSSH暗号鍵ペアの作成

- ローカルPCに保存した秘密鍵でログインサーバを経由し，フロントエンドサーバへログイン



### 3. 基本的なLinuxコマンドと演習

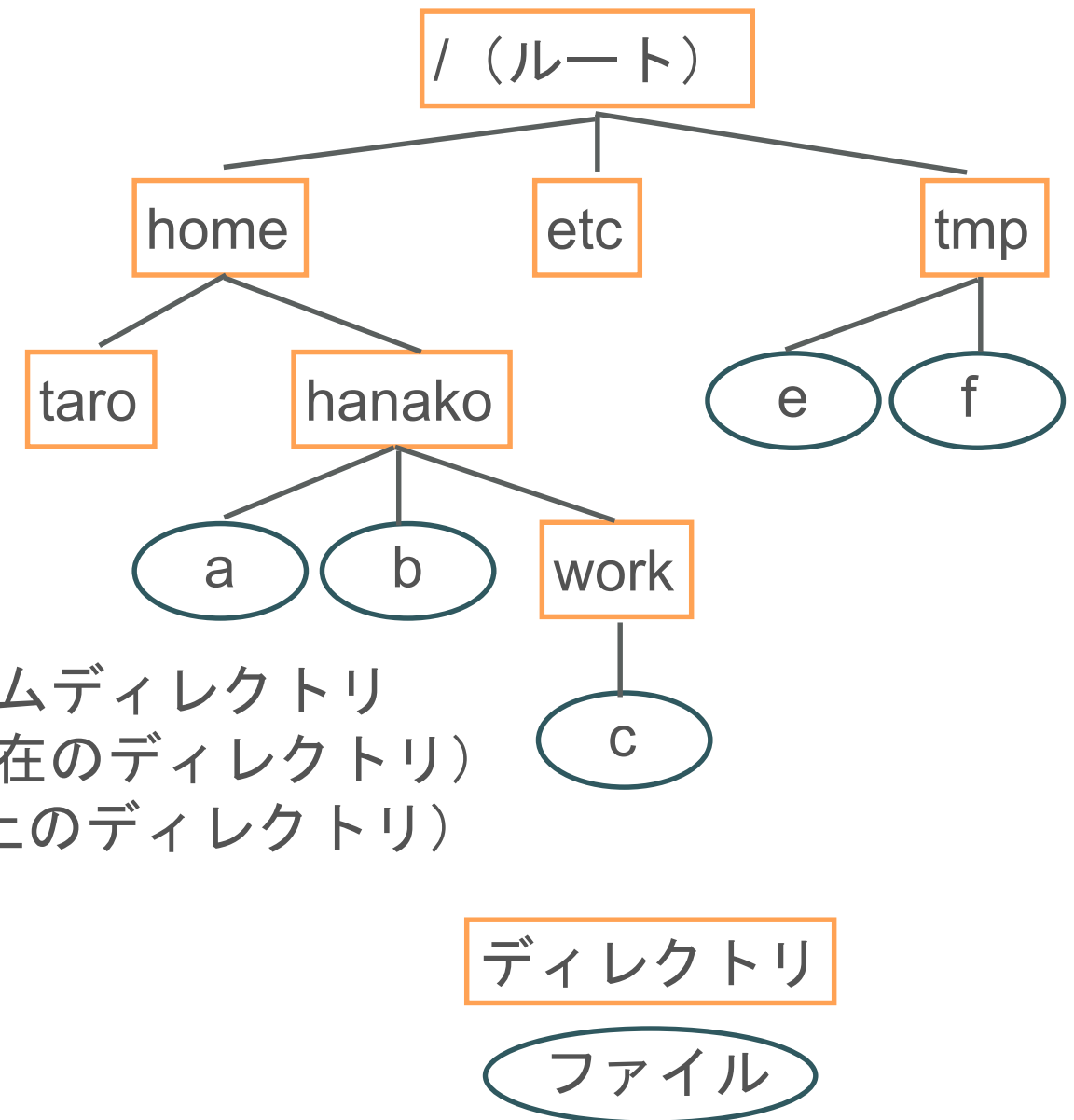
---

powershellを利用して、ログインサーバ、フロントエンドサーバにログインしてみましょう。

利用者番号：事前配布のもの

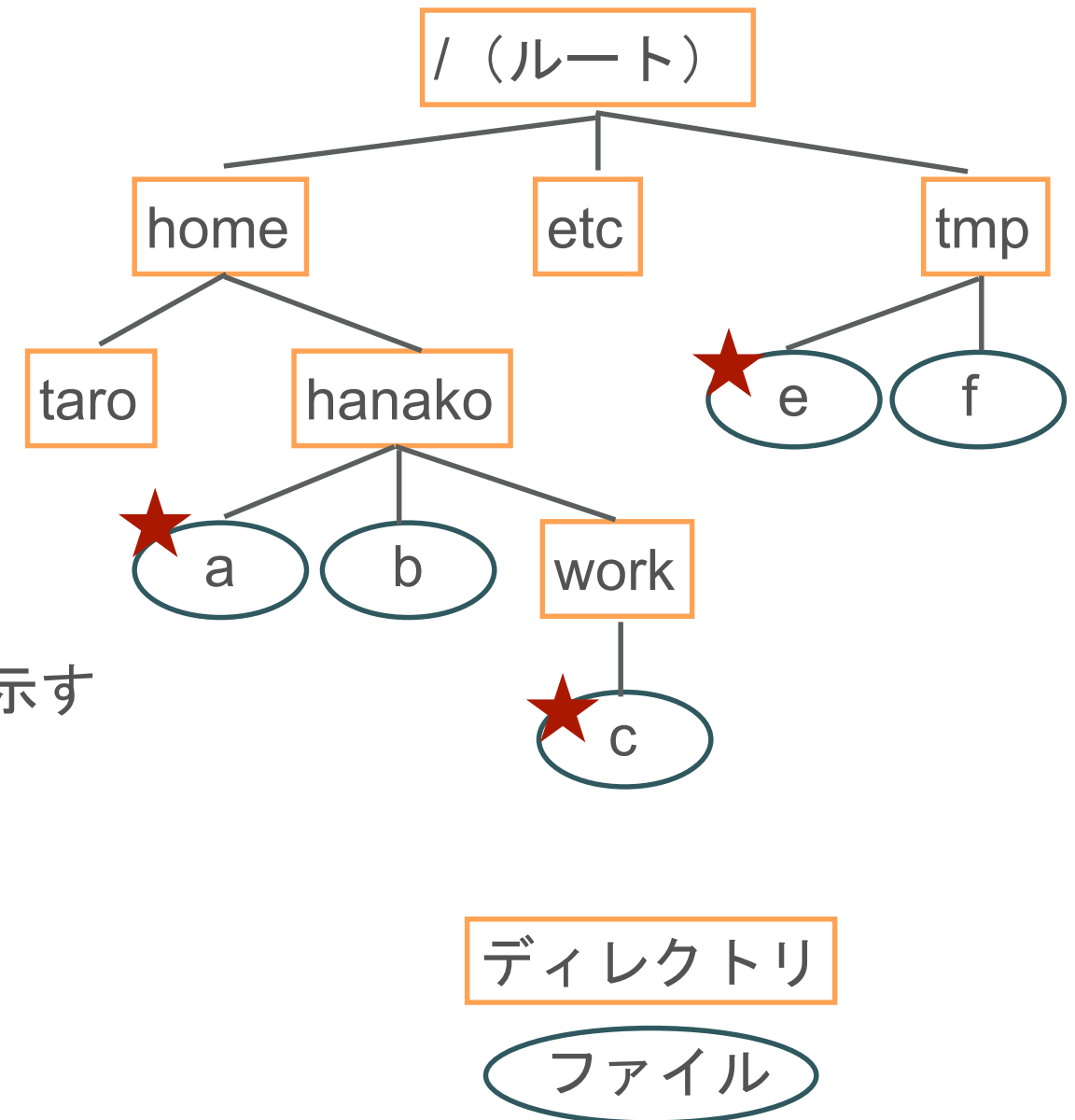
パスワード：事前にお知らせしたもの

- ファイル
  - ー プログラムやデータを扱うときの単位の一つ
- ディレクトリ (=フォルダ)
  - ー ファイルを分類・整理するための管理の単位
- Linuxのファイルシステム「木構造」
  - ー ルート「/」を最上位として管理される
  - ー その他の省略記号
    - 「~ (チルダ)」 ログインしたユーザのホームディレクトリ
    - 「. (ドット)」 カレントディレクトリ (現在のディレクトリ)
    - 「.. (ドットx2)」 親ディレクトリ (一つ上のディレクトリ)
    - 「\*」 ワイルドカード(任意の文字列)





- ファイル/ディレクトリの指定
  - ー Linuxの木構造の中での位置を示したもの
- 絶対パス
  - ー 「/ (ルート)」からの位置で示す
  - 例) /home/hanako/a (/記号で区切る)
  - /home/hanako/work/c
  - /tmp/e
- 相対パス
  - ー 「. (カレントディレクトリ)」からの位置で示す
  - 例) /home/hanako/ にいる場合
  - a (./a)
  - work/c (./work/c)
  - ../../tmp/e



\$は「プロンプト」と言い（環境によって表示が異なります）コマンドの待ち状態を示します。コマンドはこの後に入力し、(Enter)キーを押すと実行されます。

- pwd コマンドー カレントディレクトリ名を絶対パスで表示

```
$ pwd (Enter)
```

- cd コマンドー カレントディレクトリの変更（移動）

```
$ cd /tmp (Enter)
```

```
$ cd /usr/local/bin (Enter)
```

```
$ cd (Enter) ← 移動先を指定しないと、ホームディレクトリに戻る
```

コマンドの入力中に(TAB)キーを押すと、コマンド名・ファイル名などが補完されます。  
カーソルの↑キーを押すと、過去のコマンド履歴が表示されます。

- ls コマンドー ファイルやディレクトリの情報・内容を表示

\$ ls (Enter)

\$ ls /usr/local/bin (Enter)

\$ ls -ltr (Enter) ← カレントディレクトリのファイルやディレクトリを  
タイムスタンプが古いものの順に表示 (更新ファイルの確認に便利)

- ls コマンドのオプション (代表的なもの)

-l 詳細情報を表示 -a 「.」で始まる隠しファイル、ディレクトリも表示

-t タイムスタンプでソート -r ソート順を逆にする

- ※ コマンドのオプションを調べるには、「コマンド名 オプション」でWeb検索、もしくは

\$ man コマンド名 (Enter)

「スペースキー (半角)」 ページ送り、「b」 ページ戻し

「/キーワード(Enter)」 検索、「n」 次のヒット箇所、「N」 前のヒット箇所、「q」 終了

- find コマンドー ファイルやディレクトリの検索

\$ find . -name “\*tmp\*” (Enter) ← カレントディレクトリ以下tmpを含むものを検索

\$ ls -l (Enter) ← カレントディレクトリ中のリストを詳細形式で表示する

**-rw-r--r--** 1 user group 25673 7 18 10:39 file

- : 普通のファイル

d : ディレクトリ

r=読む, w=書く, x=実行

r=4, w=2, x=1  
の足し算で表記

- **r w** - **r** - - **r** - -

所有者(u)      グループ(g)      その他(o)

このファイルは・・・所有者は読み書き可能  
同グループのユーザとその他のユーザは読み込みだけ可能  
アクセス権は 644

○ スクリプトファイルに実行権を付与する

\$ chmod u+x file (Enter)

\$ chmod 744 file (Enter)

- <, >, >> (リダイレクト) — ファイルから入力、ファイルへ出力、ファイルに追記
- | (パイプ) — コマンドの結果を次のコマンドに渡す
- 実習用のファイルをコピー
  - \$ cp -r /mnt/lustre/ap/lecture/linux .
  - \$ cd linux
- cat コマンド — ファイルの内容を表示 (本来の目的はファイルの連結)
  - \$ cat TUTORIAL (Enter)
  - \$ cat TUTORIAL TUTORIAL.ja > TUTORIAL.bi (Enter)
- less コマンド — ファイルの内容を表示
  - \$ less ファイル名 (Enter)
  - 「スペースキー (半角)」 ページ送り、「b」 ページ戻し
  - 「/キーワード(Enter)」 検索 「n」 次のヒット箇所 「N」 前のヒット箇所 「q」 終了
- head コマンド — ファイルの先頭の内容を表示
  - \$ head ファイル名 (Enter)
  - 「-n 行数」 表示する行数を指定 (デフォルト10行)
- tail コマンド — ファイルの末尾の内容を表示
  - \$ tail ファイル名 (Enter)
  - 「-n 行数 -F」 ファイルが更新されると追加分を表示する (ログの監視に便利)
  - \$ tail -n 30 -F TUTORIAL



- grep コマンドー テキストファイル中のワード検索  
\$ grep 検索文字 ファイル名 (Enter)  
「-i」大文字小文字を区別しない 「-r」ディレクトリ内も検索 「-v」その文字を含まない
- sed コマンドー 文字列の置換を行う  
\$ sed 正規表現 ファイル名 (Enter)
- awk (スクリプト言語)ー テキストファイルのフィルタリングなど  
\$ cat ファイル名 | awk '{print \$1}' ← スペース区切りテキストの1カラム目を抽出
- sort コマンドー 行頭の文字をキーにして行を並び替える  
\$ sort -u ファイル名 ← 重複行を無くす
- wc コマンドー テキストファイルの行数などの数え上げ  
\$ wc -l ファイル名 ← 行数のみを表示する
- diff コマンドー 2ファイル間の相違を表示  
\$ diff file\_old file\_new (Enter)
- vimdiffー 2ファイルの相違を並べて表示、編集 (Viエディタの機能)

- mkdir コマンドー ディレクトリを作成

\$ mkdir test (Enter)

- cp コマンドー ファイルやディレクトリをコピー

\$ cp TUTORIAL test (Enter) ← testディレクトリにTUTORIALのファイルをコピー

\$ cp TUTORIAL.ja test/japanese.txt (Enter) ← testディレクトリに名前を変えて  
コピー

\$ cp -r test test2 (Enter) ←ディレクトリを丸ごとコピー

- mv コマンドー ファイルやディレクトリ名の変更・移動

\$ mv test2 test3 (Enter) ← test2をtest3にディレクトリ名変更

\$ mv test/TUTORIAL test3 (Enter) ←testディレクトリのTUTORIALを  
test3ディレクトリに移動

#### ○ rm コマンドー ファイル・ディレクトリの削除

```
$ cd test3(Enter)
$ ls (Enter)
TUTORIAL japanese.txt ←ファイルがある
$ rm TUTORIAL (Enter) ← ファイルの削除
$ ls
japanese.txt ← TUTORIAL が削除された
$ cd .. (Enter)
$ ls (Enter)
$ rm -r test3 (Enter) ← ディレクトリの削除
$ ls (Enter)
```

rmコマンドはゴミ箱への移動ではなく、データはすぐに削除されます。  
センターではデータのバックアップは行っていないので十分注意して下さい。

## 4. X Window Systemについて

---

- Linuxで標準的に使われるウィンドウシステム
  - ー 略して「X（エックス）」
- 遠隔地のサーバで実行したGUIアプリケーションを、ローカル端末に表示する
  - ー センターのGUIアプリケーション（Matlab, Mathematica）の利用に必要
- Linuxには標準でインストールされている
- MacOSはXQuartz（無料）をインストールする
- WindowsからXを利用する場合
  - ASTEC-X, OpenText Exceed 等のXサーバソフトをインストールする
  - Virtual Box 等の仮想化ソフトをインストールし、その上にLinuxの仮想環境を構築する
  - Windows 10 であれば、WSL2(Windows Subsystem for Linux)、XcXsrv を利用

### ○ エディタ

- ー テキストファイルを編集するためのソフトウェア
- ー フロントエンドサーバ上のファイルは、フロントエンドサーバ上のエディタで編集する
  - ・ ローカル端末上で編集、ファイルのアップロードを行うと、文字化けの原因になる

### ○ 代表的なエディタ

- ー nano
  - ・ 一番簡易
  - ・ コマンドが下段に表示されるためわかりやすい
- ー vi (ブイアイ)
  - ・ 軽量コンパクトなのでどんなLinuxシステムにもインストールされている
  - ・ 全てCUIで操作する
  - ・ 文字の入力を始めるのにも、コマンドを覚える必要がある
  - ・ 置換機能、マクロ機能、矩形選択機能を覚えるとプログラム作成効率がアップ
- ー Emacs (イーマックス)
  - ・ 高機能でカスタマイズ性が高い
  - ・ メールの読み書き、Webブラウズ、ファイル操作もできる
  - ・ CUI版はファイルを閉じるコマンド等を覚える必要がある

どちらかのエディタでソースコードの作成・編集が出来ると便利です。

## 5. エディタ(nano)の簡単な使い方

---

### ○ nano の起動と終了

\$ nano test.txt (Enter) ← 指定したファイルがあるときは編集、無いときは新規作成

### ○ 画面下部にコマンドが表示される

- ^ → Ctl キー
- M → Esc キー
- - → 順次押す

### ○ 代表的なコマンド

- Ctl+S ファイルを保存
- Ctl+O ファイルに名前をつけて保存
- Ctl+X nanoエディタを終了する
- Ctl+W 文字列を検索
- Ctl+D カーソルのある文字列を削除
- M-6 コピー
- Ctl+U ペースト
- M-U やり直し

コマンドは全て半角です。全角（日本語入力）でのコマンド入力は出来ません。



### ○ vi の起動と終了

\$ vi test.txt (Enter) ← 指定したファイルがあるときは編集、無いときは新規作成

### ○ コマンドモードとインサートモード

ー コマンドモード 「:」に続いてexコマンドを入力する

例) :w ファイルの上書き保存 :q エディタの終了 :wq も可能

:q! 保存しないで終了

ー インサートモード キーボードで入力した文字を入力する

### ○ コマンドモードからインサートモードへの切り替え

例) i カーソル位置から I カーソル行の先頭から

a カーソル位置の右側から A カーソル行の最後尾から

o カーソル行の次に空行を挿入してから O カーソル行の前に空行を追加してから

### ○ インサートモードからコマンドモードへの切り替え

[ESC]キーを押す 何度押してもコマンドモードになる

コマンドは全て半角です。全角（日本語入力）でのコマンド入力は出来ません。

### ○ その他のexコマンド

- x カーソル位置の文字を削除 dd カーソル行を削除 . 事前動作の繰り返し
- r カーソル位置の文字を置換 s カーソル位置の文字を削除してインサートモードに
- 例) カーソル位置の文字から5文字を削除 5x  
カーソル行から100行を削除 100dd
- u やり直し (undo) [ctrl]+r やり直しのやりなおし (redo)
- gg ファイルの先頭行にジャンプ G ファイルの最終行にジャンプ 10G 10行目にジャンプ

### ○ vi の便利な機能（コマンドモードで）

検索            /検索対象文字 (Enter) 「n」で次のヒット箇所に「N」で前のヒット箇所に  
一括置換    :%s/before/after/g (Enter) ファイル中の”before”を全て”after”に置換  
マクロ       qa 記録の開始 ～任意の操作～ q 記録の終了 @a マクロの実行  
             a-zまで記録可能 10@a 10回繰り返して実行  
矩形選択 [ctrl]+v の後、カーソル移動で矩形選択 x で選択箇所の削除など  
行番号表示 :set number    ウィンドウの分割（縦） :vs ファイル名（横） :sp ファイル名

### ○ 2ファイルの比較

```
$ vimdiff file.txt file_new.txt
```

### ○ シェルスクリプトを作成して実行してみよう

シェルスクリプトの動作：

1. 画面に文字列 "Hello, world!" と、シェルスクリプトを実行したときの日時を出力する

```
$ nano hello.sh (Enter) ← ファイルの編集
```

```
. . .
```

```
$ chmod u+x hello.sh ← ファイルに実行権を付与する
```

```
$ ./hello.sh ← シェルスクリプトの実行
```

解答例 注：行番号は入力しません

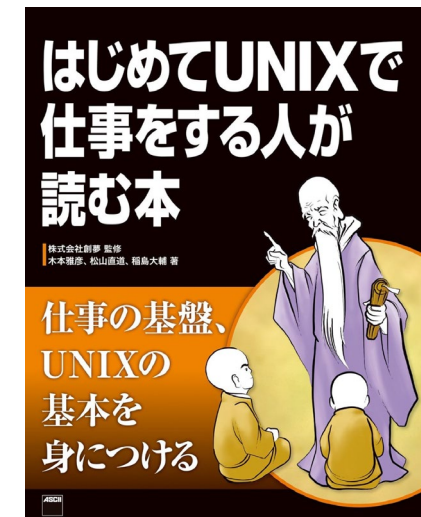
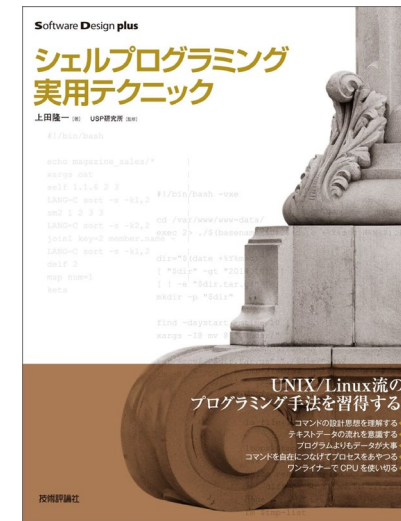
```
1 #!/bin/sh
2 STR="Hello, world!"
3 echo $STR
4 date
```

「シェルプログラミング 実用テクニック」  
上田隆一 著 USP研究所 監修 技術評論社

「はじめてUNIXで仕事をする人が読む本」  
木本雅彦 他 アスキー・メディアワークス

「入門vi 第6版」  
リンダ・ラム 他 オライリー・ジャパン

「入門UNIXシェルプログラミング  
ー シェルの基礎から学ぶUNIXの世界」  
ブルース・ブリン 他 技術評論社



---

本日の講習会は以上で終了です。

おつかれさまでした。