インフラ構築基礎実習最終課題

サーバ構築の手順書

3-I 30 番　彌園和志

目次

環境情報

・ハードウェアモデル: ASUSTeK COMPUTER INC. VivoBook 12_ASUS Laptop E203MA
・メモリ: 4.0GiB
・プロセッサー: intel® Celeron(R) N4000 CPU ` 1.10GHz × 2
・ディスク情報: 62.5GB
・OS 名: Ubuntu 22.04LTS
・OS の種類: 64 ビット
・GNOME のバージョン: 42.0
・ウィンドウシステム: Wayland

第 1 回目

Ubuntu のインストール

まず、ubuntu を PC にインストールする。その次に Rufus をインストールする。その次に F2 を押しながら ASUS を起動させる。そして Boot Priority の中にある UEFI と Windows Boot Manager の順番を入れ替える。Secure Boot Control を Disabled にする。そして Save & Exit を実行する。その後画面が表示されると Try or Install Ubuntu が選択された状態で e キーを押し、Ctrl + X を押して起動を再開する。また図 1-1 に示している部分は PC 内にデータがない場合は中央の部分を選んでもよいが基本的に一番上の部分を選ぶ。

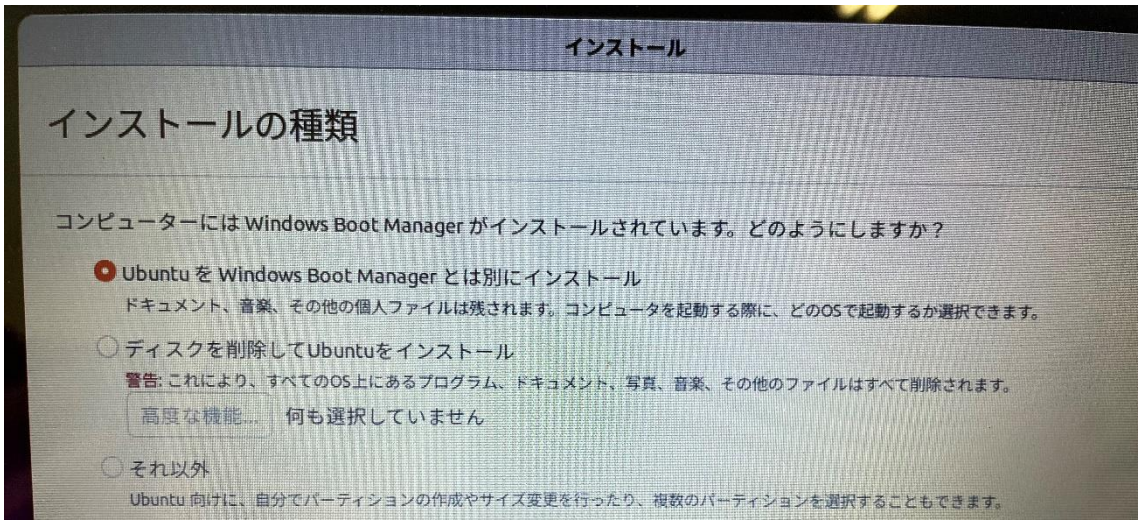

図 1-1 インストール画面

Ubuntu の設定

Ubuntu が起動出来たらキーボードの設定は Japanese にし、Wi-Fi の設定をする。その後通常のインストールを選択し、インストールタイプを選ぶ。その後地域を東京に選択しユーザー設定をする。最後にインストールを待ち、完了すると再起動する。

参考資料

ASUS ROG ZEPHYRUS G16（GU603VI-I9R4070G）に Ubuntu をデュアルブートでインストールした
https://qiita.com/masakinoda111/items/e557b554a1816b1d1eed
Windows で Ubuntu のインストール USB メディアを作成する
https://diagnose-fix.com/topic2-003/
ubuntu をインストールする
https://yumeno.me/ubuntu-install#google_vignette

第 3 回目

Apache のインストール

インストール前にパッケージの更新をするために、$ sudo apt update のコマンドを実行し、$ sudo apt -y install apache2 のコマンドで Apache HTTP Server のインストールができる。その後、$ sudo systemctl start apache2 と $ sudo systemctl enable apache2 と $ sudo systemctl status apache2 のコマンドを実行するとサーバのステータスが確認できる。

第 4 回目

Apache の設定

まず、リスト 4-1 のコードを使用し仮想ホストの設定ファイルを作成する。

リスト 4-1 仮想ホストの設定ファイルの作成

```
$ sudo touch /etc/apache2/sites-available/ファイル名.conf
```

次に、vi コマンドを使用し設定ファイルの編集をする。また、設定内容のコードをリスト 4-2 に示し、ディレクトリパスは/var/www/html などのようにする。

リスト 4-2 設定ファイルの編集

```
<VirtualHost *:80>
  ServerAdmin webmaster@localhost
  DocumentRoot ディレクトリパス
  DirectoryIndex ファイル名
  ErrorLog ${APACHE_LOG_DIR}/error.log
  CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

そして、「$ sudo mkdir -p ディレクトリパス」を入力するとディレクトリが作成され、「$ sudo touch ディレクトリパス/ファイル名」と「$ sudo vi ディレクトリパス/ファイル名」を入力すると html が編集できる。その後、作成した仮想ホストを Web サーバに反映させるためには、仮想ホストの有効化をする必要があるのでリスト 4-3 の仮想ホストの有効化コードを入力し、リスト 4-4 のデフォルトの仮想ホストの無効化コードを入力する。

リスト 4-3 仮想ホストの有効化

```
$ sudo a2ensite ファイル名.conf # 仮想ホストの有効化
$ sudo systemctl reload apache2 # 有効化した仮想ホストをサーバに反映
```

リスト 4-4 デフォルトの仮想ホストの無効化

```
$ sudo a2dissite 000-default.conf # 仮想ホストの無効化
$ sudo systemctl reload apache2 # 有効化した仮想ホストをサーバに反映
```

ファイアーウォールの設定

ファイアウォールの設定をする。「$ sudo ufw enable」を入力するとファイアウォールが有効化され、「$ sudo ufw allow 'Apache' 」を入力すると 80 番ポートが開放される。また、リスト 4-5 のコードを使用すると 80 番ポートと 443 番ポートも使用できる。また、HTTP が 80 番ポートで、HTTPS が 443 番ポートである。

リスト 4-5 ファイアウォール設定

```
# 80 番ポートの開放
$ sudo ufw allow 'Apache'
$ sudo ufw allow 80/tcp
# 443 番ポートの開放
$ sudo ufw allow 'Apache Secure'
$ sudo ufw allow 443/tcp
# 80 番と 443 番ポートの同時開放
$ sudo ufw allow 'Apache Full
```

パスワード認証の設定

まず、「$ sudo apt install apache2-utils」を入力して apache2-utils パッケージのインストールする。次にリスト 4-6 のコードを入力してユーザーの登録をする。また、初回以外はリスト 4-5 の「-c」を除く。

リスト 4-6 ユーザー登録

```
$ sudo htpasswd -c /etc/apache2/.htpasswd ユーザ名 # 初回
$ sudo htpasswd /etc/apache2/.htpasswd ユーザ名 # 次回以降
```

その後リスト 4-7 のコードを入力し設定ファイルの編集をする。また、このコードは VirtualHost の中に入力する。その後「$ sudo systemctl reload apache2」を入力して Apache の再起動をする。

リスト 4-7 設定ファイル編集

```
<Directory "認証を利用するディレクトリパス">
AuthType Basic
AuthName "Restricted Content"
AuthUserFile /etc/apache2/.htpasswd

42

Require valid-user
</Directory>
```

第 5 回目

SSH のインストール

リスト 5-1 のコードを使用し、SSH をインストールする

リスト 5-1 OpenSSH のインストール

```
$ sudo apt install openssh-server
```

また、インストールした後、systemctl コマンドで ssh サービスが起動しているか確認し、起動していない場合はリスト 5-2 のコードを使用して手動で起動させる。

リスト 5-2 OpenSSH を手動で起動

```
$ sudo systemctl start ssh
$ sudo systemctl enable ssh
$ sudo systemctl status ssh
```

ポート番号の設定

まず、変更を加える前に、まず設定ファイルのバックアップをリスト 5-3 のコードを使用して作成する。

リスト 5-3 設定ファイルのバックアップ

```
$ sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config_backup
```

次にセキュリティリスクが高まるため 22 番以外のポートを利用する。慣習的には 5022 や 10022 のように、22 番に高い桁の数を合わせた番号にすることが多い。
設定ファイル(/etc/ssh/sshd_config)から 「sudo vi sshd_config」と入力し、「#Port 22」と記述されている行を探し、行頭にあるコメントアウトを削除する。そして 22 番から任意のポート番号に変更する。5022 や 10022 のように、22 番に高い桁の数を合わせた番号にすることが多い。
次にファイアウォールに リスト 5-4 の SSH のポート利用許可のルールを追加する。

リスト 5-4 SSH のポート利用許可

```
$ sudo ufw allow 任意のポート番号/tcp
```

ログインの制限

ログインを禁止する場合は、リスト 5-5 の設定ファイルに以下のパラメータを記述する。
また、root ユーザに限らず、特定のユーザのみログイン可能にする場合は、設定ファイルにリスト 5-6 以下のパラメータを記述する。

リスト 5-5 ログイン禁止

```
PermitRootLogin no
```

リスト 5-6 許可ユーザの設定パラメータ

```
AllowUsers ユーザ名
```

SSH ログイン

SSH の基本設定が完了した後、リスト 5-7 を使用し SSH 接続可能かどうかテストする。
また、Ubuntu での IP アドレスは「hostmane -i」で確認できる。

リスト 5-7 SSH のテスト

```
$ ssh ユーザ名@ホスト名(もしくは IP アドレス) -p ポート番号
```

2 段階認証の利用

まず、リスト 5-8 を使用して SSH を利用するクライアント PC で鍵生成を行う。また、例もリスト 5-8 に示す。

リスト 5-8 鍵生成

```
$ ssh-keygen -t 暗号化アルゴリズム -f ファイル名
$ ssh-keygen -t ed25519 -f ./.ssh/id_ed25519
```

次に、先ほど生成した鍵をリスト 5-9 のコードを使用してサーバに登録する。

リスト 5-9 鍵登録

```
$ cat $HOME¥.ssh¥公開鍵のファイル名 | ssh ユーザ名@ホスト名
(もしくは IP アドレス)
-p ポート番号 "mkdir -p ~/.ssh && chmod 700 ~/.ssh && cat >>
~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys"
```

公開鍵の登録を行った後、2 段階認証を有効化するために設定ファイルを開き、編集を行う。追加する内容はリスト 5-10 に示す。その後、ssh を再起動させる。

リスト 5-10 2 段階認証の有効化

```
AuthenticationMethods publickey,password
```

Docker のインストール

Ubuntu で Docker をインストールする場合はリスト 5-11 の手順でコマンドを実行する。

リスト 5-11 Docker のインストール

```
# Add Docker's official GPG key:
$ sudo apt-get update
$ sudo apt-get install ca-certificates curl
$ sudo install -m 0755 -d /etc/apt/keyrings
$ sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
$ sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
$ echo
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu ¥
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | ¥
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

その後、Docker のバージョン確認と状態確認を行う。使用するコードはリスト 5-12 に示す。リスト 5-12 のコードを入力し成功していたら図 5-1 のような分が表示される。
また、「sudo docker」 を docker コマンドをユーザで実行したい場合は、docker グループにユーザを追加しておく。使用するコードはリスト 5-13 に示す。

リスト 5-12 バージョン確認と状態確認

```
$ sudo docker --version
$ sudo systemctl status docker
```

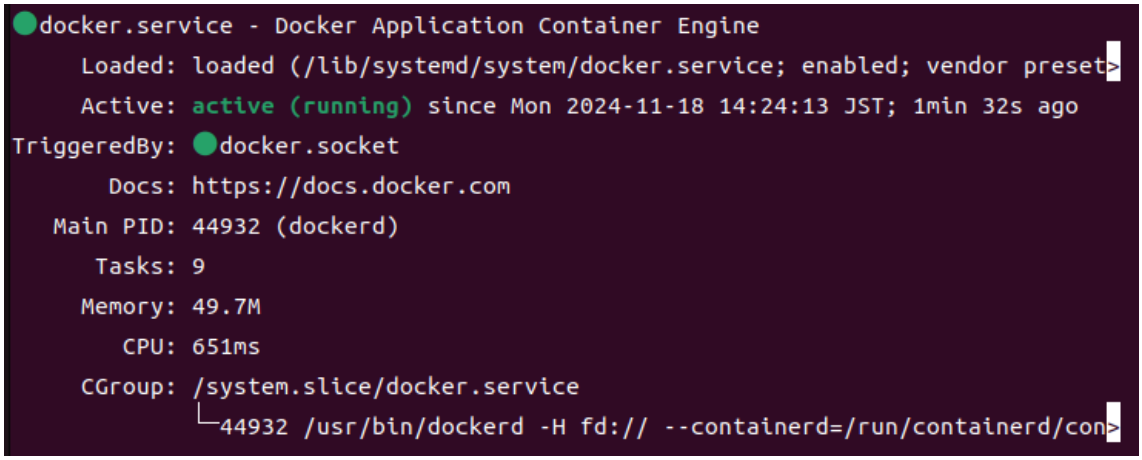

図 5-1 Docker インストール成功後

リスト 5-13 ユーザの追加

```
$ sudo usermod -aG docker $USER
$ newgrp docker
```

参考資料

Install Docker Engine on Ubuntu
https://docs.docker.com/engine/install/ubuntu/

第 6 回目

コンテナの起動

まず、コンテナ起動の確認のために、HelloWorld コンテナを利用して確認する。使用コードと実行結果をリスト 6-1 に示す。

リスト 6-1 コンテナの起動

```
$ docker container run hello-world
Hello from Docker!
This message shows that your installation appears to be working correctly.
To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub. (amd64)
3. The Docker daemon created a new container from that image which runs the
executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent
it
to your terminal.
To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/
For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

Docker を用いた Web サーバの構築

httpb を用いてコンテナを起動し、Web サーバを起動する。まず、Web サーバのコンテンツの管理などのためのディレクトリを作成する。使用コマンドをリスト 6-2 に示す。その後、httpd イメージを使いコンテナを起動させる。使用コマンドをリスト 6-3 に示す。

次に作成したディレクトリの下に htocs や conf が作成され、その下に index.html を作成し、内容を入力するとその内容がアクセス時に表示されるようになる。

リスト 6-2 ディレクトリの作成

```
$ mkdir -p ~/ディレクトリ名/htdocs ~/ディレクトリ名/conf
```

リスト 6-3 httpd コンテナの起動

```
$ cd ~/ディレクトリ名
$ docker run -dit --name my-apache-app -p 8080:80 -
v ./htdocs:/usr/local/apache2/htdocs/ -v ./conf:/usr/apache2/conf httpd:2.4
```

参考資料一覧

ASUS ROG ZEPHYRUS G16（GU603VI-I9R4070G）に Ubuntu をデュアルブートでインストールした
https://qiita.com/masakinoda111/items/e557b554a1816b1d1eed
Windows で Ubuntu のインストール USB メディアを作成する
https://diagnose-fix.com/topic2-003/
ubuntu をインストールする
https://yumeno.me/ubuntu-install#google_vignette
Install Docker Engine on Ubuntu
https://docs.docker.com/engine/install/ubuntu/