サーバー構築を実習形式で学べる決定版

Linux
サーバー構築標準教科書

Ubuntu版

Ver. 1.0.0

LinuC
Linux技術者認定
https://linuc.org

open your NEXT future
LPI-JAPAN
Linux Professional Certification

Linuxサーバー構築標準教科書 (Ubuntu版)

2026年7月1日　Ver.1.0.0
LPI-Japan　発行

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 サーバー構築標準教科書 (Ubuntu 版)」（以下、本教科書）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。

本教科書は、多くの教育機関から Linux によるサーバーの構築を「基礎」から学習するための教材や学習環境の整備に対するご要望があり、開発したものです。

公開にあたっては、本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。

本教科書の最新情報は以下の Web ページをご参照ください。

`https://linuc.org/textbooks/server/`

本教科書の目的

本教科書の目的は、LinuC レベル 2 の 201 試験と 202 試験の学習範囲に含まれるサーバー構築の知識を、構築の実習を通しながら学習することにあります。

サーバーを構築した環境で、実際に Web アクセスをしたりメールの送受信をしたりすることで、サーバーの動作原理やプロトコルの仕組みを理解することができます。

本教科書でカバーされない範囲

実習手順の記述を簡潔にするため、LinuC レベル 1 の学習範囲に含まれる Linux の基本的な操作やシステム管理についての記述は必要最低限になっています。また、自分で調べることも学習において重要であるため、あえて調べてみる余地を残しています。分からない部分は別途調べて理解を深めてください。

想定している実習環境

本教科書は一人で独学自習できることを想定しています。実習環境として、以下の環境を構築しています。

仮想マシンを利用

仮想マシンを利用して学習環境を構築します。仮想マシンを利用すると、Windows や Linux、macOS 上の仮想マシンに Linux をインストールし、動作させることができます。

仮想マシンは複数同時に動作させることができるので、複数のサーバーをネットワークで接続して行う DNS やメールの実習も 1 台の物理マシンで行えます。

仮想マシン環境を実現するソフトウェアとして、以下のようなものがあげられます。

- VirtualBox（Windows、Linux、macOS）
- VMware Workstation（Windows）
- VMware Fusion（macOS）
- Parallels Desktop（macOS）
- UTM（macOS）
- Linux KVM（Linux）
- Proxmox（Linux）

本教科書では、VirtualBox を Windows 上で実行して解説を進めます。

OS

本教科書では、Linux ディストリビューションとして Ubuntu 24.04.2 LTS(Server) を利用します。

実習例では Intel/AMD x86_64 アーキテクチャに対応したバージョンを利用していますが、ARM 版などその他のアーキテクチャに対応したバージョンでも実習を行うことができます。

ネットワーク

実習を行うネットワークはインターネットに接続できることを前提としています。

教室での実習

教室などで複数の受講者が実習を行う場合でも、基本的に各受講者がそれぞれ自身の仮想マシンを使って個別に実習を行うことを想定しています。

受講者同士がネットワークを経由して相互に接続する実習を行いたい場合、講師が仮想マシンや OS の設定変更を個別に指示するか、各受講者が教科書の内容を一通り実習した後に、受講者相互で接続する実習を行うようにしてください。その際には特に以下の点について留意してください。

- VirtualBox で作成する仮想マシンのネットワークをブリッジ接続して相互接続できるようにする
- 各受講者に異なる IP アドレスとホスト名・ドメイン名を割り当てる
- DNS による名前解決の設定を変更する

全体の流れ

本教科書では、以下の通りに実習を進めます。

1 章 Linux サーバーの概要

本教科書で行う実習の全体像や事前に説明しておくべき事項を解説しています。

2 章 VirtualBox のインストールと仮想マシンの作成

仮想マシンについての解説と、VirtualBox のインストール、仮想マシンの作成を行います。

3 章 Linux のインストールと設定

仮想マシンに Linux をインストールします。

4 章 Web サーバーの構築

Linux に Web サーバーとして nginx をインストールします。

5 章 DNS サーバーの構築

Linux に DNS サーバーとして BIND をインストールして、ドメインを設定し、名前解決が行えるようにします。複数の仮想マシンを用意し、相互に名前解決で接続できるようにします。

6 章 メールサーバーの構築

Linux にメールサーバーとして Postfix や Dovecot をインストールし、メールの送受信が行えるように設定します。

7 章 ネットワークとセキュリティの設定

Linux のネットワークやセキュリティを設定します。

執筆者・制作者紹介

本教科書は、オープンなプロジェクト形式で開発を行っています。企画段階から意見交換を行い、事前の技術的な調査、執筆、レビューなどをプロジェクトのメンバーで分担して行っています。

鯨井 貴博（バージョン 1 執筆担当、LinuC エヴァンジェリスト/株式会社ゼウス・エンタープライズ）

本教科書は、Linux/オープンソースソフトウェアをこれから勉強する皆さんと、熱心に指導に当たられている先生方の一助になればと思い、作成いたしました。

開発にご協力をいただいた方々

- 宮原 徹
- 小笠原 種高
- 大澤 文孝
- LinuC Open Network

また、レビュアー、そして利用者の皆様からフィードバックをいただきました。厚く御礼申し上げます。

著作権

使用に関する権利

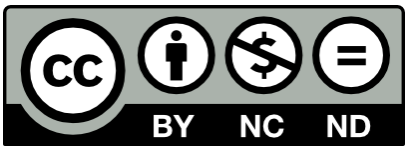

図 1: CC BY-NC-ND 4.0

表示

本教科書は、特定非営利活動法人エルピーアイジャパンに著作権が帰属するものであることを表示してください。

非営利

本教科書は、非営利目的で教材として自由に利用することができます。

商業上の利得や金銭的報酬を主な目的とした営利目的での利用は、特定非営利活動法人エルピーアイジャパンによる許諾が必要です。ただし、本教科書を利用した教育において、本教科書自体の対価を請求しない場合は、営利目的の教育であっても基本的に使用できます。その場合も含め、LPI-Japan 事務局までお気軽にお問い合わせください。

＊営利目的の利用とは以下のとおり規定しております。営利企業または非営利団体において、商業上の利得や金銭的報酬を目的に本教科書の印刷実費以上の対価を受講生に請求して本教科書の複製を用いた研修や講義を行うこと。

改変禁止

本教科書は、改変せず使用してください。本教科書に対する改変は、特定非営利活動法人エルピーアイジャパンまたは特定非営利活動法人エルピーアイジャパンが認める団体により行われています。

フィードバック

フィードバックは誰でも参加できる Slack で受け付けていますので、積極的にご参加ください。Slack 参加の詳細は以下の本教科書の Web ページを参照してください。

`https://linuc.org/textbooks/server/`

本教科書の使用に関するお問合せ先

特定非営利活動法人エルピーアイジャパン（LPI-Japan）事務局

お問い合わせ：`https://lpij.tayori.com/f/textbookinfo/`

図 2: https://linuc.org/textbooks/server/

図 3: https://lpij.tayori.com/f/textbookinfo/

Linux 技術者認定「LinuC（リナック）」のご紹介

Linux 技術者認定「LinuC（リナック）」とは、クラウド／ DX 時代の IT エンジニアに求められるシステム構築から運用管理に必要なスキルを証明できる技術者認定です。アーキテクチャ設計からシステム構築、運用管理までの技術領域を広くカバーしており、４つのレベルの認定取得を通じて一歩ずつ確実に求められるスキルを習得し、それを証明することができます。

LinuC の出題範囲策定や試験開発は、実際に現場で活躍しているハイレベルな IT エンジニアが参加するコミュニティによって行われています。そのため、グローバルで業界標準として利用されている技術領域をカバーし、システム開発や運用管理の現場で本当に必要とされる知識や実践的なスキルを問う内容になっています。その結果として従来型の Linux 領域にとどまった技術認定とは異なり、国内・海外を問わず活躍を目指す IT エンジニアにとっても十分役立つ技術者認定となりました。

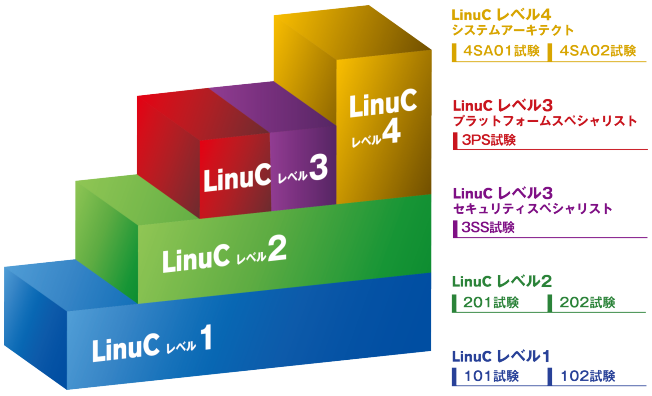

図 4: LinuC の体系図

LinuC レベル 1

コンピュータシステムを理解し、仮想環境を含む Linux システムの基本操作とシステム管理が行える即戦力エンジニアの証明（ITSS レベル 1）

LinuC レベル２

仮想環境を含む Linux のシステム設計・ネットワーク構築において、アーキテクチャに基づいた設計・導入・保守・問題解決ができるエンジニアの証明（ITSS レベル 2）

LinuC レベル３ プラットフォームスペシャリスト・セキュリティスペシャリスト

仮想化・自動化などにより柔軟性と可用性を備えた Linux プラットフォームの構築運用や、セキュア Linux システムのための OS からミドルウェアまでの各層堅牢化や認証認可基盤の構築および攻撃対策を実現できるスペシャリストの証明（ITSS レベル 3）

LinuC レベル４ システムアーキテクト

オンプレ／クラウド、物理／仮想化を含むシステムのライフサイクル全体を俯瞰して最適なアーキテクチャを設計・構築ができる上級エンジニアの証明（ITSS レベル 4）

LinuC の詳細については、以下の Web サイトを参照してください。

https://linuc.org/about/01.html

図 5: https://linuc.org/about/01.html

目次

1 Linux サーバーの概要

本教科書では、Linux をインストールしてサーバー環境を構築する実習をしながら、LinuC レベル 1 からレベル 2 の出題範囲に含まれる主要な技術の理解を目指します。

第 1 章では Linux サーバーの概要について解説し、2 章以降に行う実習に必要な環境と知識の確認を行います。

1.1 用語集

Linux

Linus Torvalds 氏により開発された、UNIX 互換を目指した OS の総称を Linux といいます。ソースコードは公開されており、世界中の開発者の協力により、日々開発が継続されています。

Linux ディストリビューション

Linux は狭い意味では OS の核となる中心部（＝カーネル）のみを指しますが、Linux カーネルだけではシステムは動作しません。カーネル以外のさまざまなソフトウェアやインストーラを追加して、利用できるようにしたのが Linux ディストリビューションです。Linux ディストリビューションごとに開発方針があり、それに沿ってソフトウェアがまとめられ、リリースが行われています。

Ubuntu

Linux のディストリビューションの 1 つです。Debian という Linux ディストリビューション互換の環境を無償で提供している Linux ディストリビューションで、Ubuntu コミュニティによって開発されています。

IP アドレス

インターネットにおいて IP で通信が行われる場合、ホスト一つ一つに IP アドレスが割り当てられます。IP アドレスとはインターネット上でのホストの所在地を示す「住所」にあたります。現在では IPv4 と IPv6 の 2 種類がありますが、本教科書では IPv4 を使います。IPv4 では 32 ビットの IP アドレスを、8 ビット毎に.(ドット) で区切って 10 進数で表記します。

ネットワークアドレス

IP ネットワークは 1 つ以上のホストの集まりであるネットワークを形成し、そのネットワーク内の通信と、ネットワーク間の通信で構成されています。このネットワークの識別に利用されるのが、ネットワークアドレスです。IP アドレスは、所属しているネットワークを表すネットワーク部と、ホスト個別に割り当てられるホスト部に分けることができます。

サブネットマスク

IP アドレスのうちネットワークアドレス部を識別するための数値のことをいいます。32 ビットのうち、先頭からネットワーク部となる部分のビットを 1 として表します。たとえば先頭から 24 ビットがネットワーク部になる場合、IP アドレスの後ろに「/24」と表記したり、10 進数に変換して「255.255.255.0」と表記したりします。

ホスト名

ネットワークに接続されたコンピュータに割り当てられた名称のことをいいます。ドメイン名を省略した短い名前を指す場合と、ドメイン名を含んだ FQDN(Fully Qualified Domain Name) の形式のものがあります。

ドメイン名

インターネット上に存在するコンピュータやネットワークを識別するために付けられている名前です。会社などの組織毎に独自のドメイン名を取得して利用します。ドメイン名はアルファベット、数字、一部の記号の組み合わせで構成されますが、日本語.jp のような国際化ドメインも使われるようになっています。

DNS サーバーアドレス

FQDN（Fully Qualified Domain Name ＝ホスト名＋ドメイン名）と IP アドレスの変換を「名前解決」と呼びます。この名前解決を行うのが DNS（Domain Name System）です。ホストは設定した DNS サーバーアドレスに対して名前解決を依頼します。

ハードディスク（HDD）

磁気を用いた記憶媒体であり、パソコンの記憶媒体の他、ビデオレコーダーなどの記憶媒体としても用いられています。

SSD(Solid State Drive)

半導体メモリであるフラッシュメモリを用いた記憶媒体であり、ハードディスクよりも読み書きの処理性能に優れています。

DVD

光学メディアの 1 種類で、ビデオ再生での利用で普及し、現在ではデータ記録の用途でも利用されています。約 700MB の CD-ROM に比べ、約 4.7GB と大容量でも利用できることから、OS のインストールディスクとしても利用されています。

USB メモリ

USB に接続して利用する外部ストレージです。OS のインストールディスクとして利用できます。

1.2 実習で利用するハードウェア

本教科書の実習では、仮想マシンを使って実習環境を構築します。

仮想マシンの詳細については第 2 章で解説しますが、ここでは実習用に用意するハードウェアの仕様などについて解説します。

1.2.1 マシン本体

Windows や Linux、macOS が動作する、いわゆる「パソコン」（PC）を想定しています。用意した PC に「VirtualBox」のような仮想マシンソフトウェアをインストールして実習環境を構築します。

1.2.2 CPU

実習では CPU 負荷の高い処理は行わないため、高性能な CPU は必要ありません。ただし、仮想マシンを実行するために仮想化支援技術が必要となります。古い CPU でない限り仮想化支援技術を備えていますが、UEFI（BIOS）で有効にする必要があります。

本教科書では VirtualBox を利用するため、CPU アーキテクチャは IA（Intel Architecture）の CPU を想定しています。

ARM アーキテクチャの CPU に対応した仮想マシンソフトウェアを利用し、ARM 対応の Linux ディストリビューションをインストールすれば同様の実習を行うこともできます。仮想マシンの設定などが異なりますので、適宜読み替えて実習を行ってください。

1.2.3 メモリ

Ubuntu 24.04.2 LTS(Server) では最小 2.0GB 以上のメモリが推奨されています。

DNS やメールの実習では 3 台の仮想マシンを使って実習を行うので、3 台分で 6.0GB のメモリが必要です。メモリは仮想マシンの他に、OS（ホスト OS と呼ぶ）やその他のアプリケーションを同時に動作させるために必要となるので、最低でも 8GB、快適に実習を行うには 16GB 以上のメモリが搭載されていることが望ましいでしょう。

メモリの搭載量が少ない場合には、仮想マシンへのメモリの割り当てを減らす、余計なアプリケーションを動作させないようにするなどのメモリ効率改善や、メモリスワップ発生による速度低下を許容する必要があります。

1.2.4　ストレージ

仮想マシンが利用する仮想ハードディスクはファイルとして PC に接続されたハードディスクや SSD などのストレージに保管されます。仮想ハードディスクファイルは利用した分だけストレージを消費します。

実習で利用する環境はおよそ 5GB の容量を消費します。VirtualBox のデフォルトでは最大 25GB の仮想ハードディスクファイルが作られるので、3 台の仮想マシンで少なくとも 5GB、最大 75GB の容量を用意しておく必要があります。

仮想ハードディスクファイルの読み書きは頻繁に行われるため、ハードディスクよりも高速な SSD を利用するのが望ましいでしょう。

1.2.5　ネットワーク

実習ではインターネットに接続できる必要があります。実習用に用意する PC は有線 LAN、無線 LAN、どちらの方式で接続されていても構いません。

1.3　利用する Linux のディストリビューション

本教科書では、Ubuntu 24.04.2 LTS(Server) の Intel/AMD x86_64 アーキテクチャに対応したバージョンを利用します。

```
https://ubuntu.com/
```

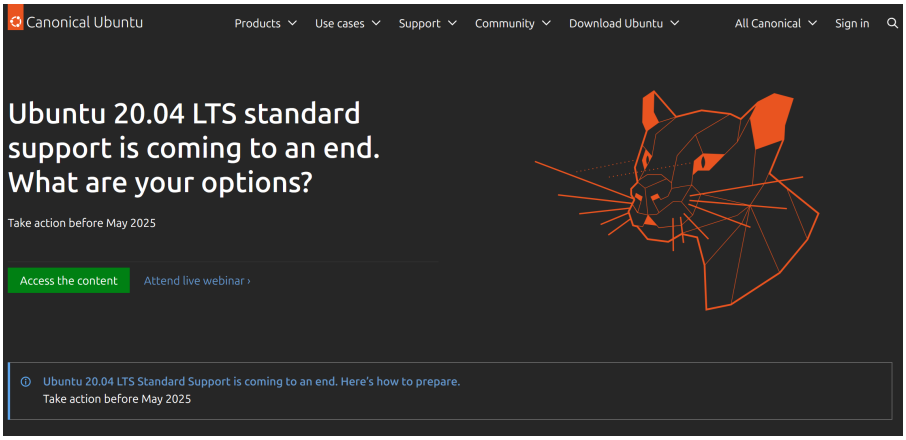

図 6: Ubuntu の Web サイト

Ubuntu は、Linux ディストリビューションである Debian をベースにしたディストリビューションとして提供されています。利用に際し費用が発生しない、無償で提供されているディストリビューションです。

1.3.1　インストール用 ISO イメージの入手

Ubuntu が配布している ISO イメージを、ダウンロードします。仮想マシンは、ISO イメージを仮想光学ドライブにセットすることでインストールが行えるので、インストール用の DVD/USB メモリを作成する必要がありません。

ダウンロード方法　ISO イメージをダウンロードするには、以下の URL にアクセスします。

```
https://ubuntu.com/download/server
```

1.4　ISO イメージのファイル名

ISO イメージは以下のようなファイル名になっています。

```
ubuntu-バージョン-live-イメージの種類-アーキテクチャ.iso
```

1.5 バージョン

本教科書では、本教科書の作成時点で最新であった Ubuntu 24.04.2 LTS(Server) を利用した構築方法について解説しています。

今後のバージョンアップで、より新しいバージョンの Ubuntu が入手可能になっているかもしれません。バージョン 24.04 系であればマイナーバージョンによる大きな差は無いので同様の手順でサーバーの構築ができると推測されますが、セキュリティ対応などの関係でデフォルトの設定が変更されることで動作が変わる場合があります。

初めて本教科書の内容を学習する際には、まずはバージョンを合わせて動作を確認し、その後異なるバージョンで同様に動作するか確認してみてください。バージョンによる動作の違いについては、本教科書の情報交換を行う Slack で情報共有を行い、本教科書の今後のバージョンアップで対応していく予定です。

1.6 アーキテクチャ

Linux カーネルは様々な種類の CPU アーキテクチャに対応しています。アーキテクチャによってバイナリが異なるため、アーキテクチャに合わせた ISO イメージを選択する必要があります。主なアーキテクチャには以下のものがあります。

1.6.1 arm64

Intel や AMD の CPU アーキテクチャです。64 ビット版になります。

この他に ARM、IBM POWER、IBM Z (s390x) や RISC-V などのアーキテクチャに対応した ISO イメージが用意されています。

1.7 ISO イメージの種類

ISO イメージには、いくつかの種類があります。インストールの目的によってイメージを選択します。Ubuntu には、以下の 4 種類の ISO イメージが用意されています。

- Server(CLI 環境で操作するサーバ用途)
- Desktop(GUI 環境を含むデスクトップ用途)
- Core(組み込み用途)
- Cloud(クラウド環境用途)

本教科書の演習では、Server(CLI 環境で操作するサーバ用途) を使ってインストールします。ubuntu-24.04.2-live-server-amd64.iso をダウンロードしてください。

実務では最小限のインストールを行い、必要なパッケージを追加してサーバーを構築することが多いですが、今回は作業に必要となるエディタやツールなどを予めインストールすることとしています。

物理マシンに直接インストールしたい場合には、USB メモリに書き込んで起動用 USB メモリを作成します。

1.8 ネットワーク環境について

本教科書では仮想マシンを利用して実習を行うため、特別なネットワークは必要ありませんが、以下の点について確認しておいてください。

1.8.1 インターネットへの接続

ソフトウェアのインストールを行うため、インターネット上に用意されているリポジトリサーバーへのアクセスが必要になります。実習で利用する VirtualBox では、ブリッジで外部ネットワークに接続します。

プロキシーを経由して接続する必要がある場合には、接続のための設定情報を入手するか、ネットワーク管理者に相談してください。

1.8.2 仮想マシン間の接続

仮想マシン間の接続は仮想ネットワークを経由して行うため、利用している仮想マシンソフトウェアの仕様によって異なります。本教科書で利用する VirtualBox はブリッジ接続を行うため、IP アドレスなどは実習環境に合わせたものとしてください。

1.9 ネットワークの設定項目

サーバーを構築していくにあたり、以下のような設定項目をどう設定するか決める必要があります。

1.9.1 ドメイン名

ドメイン名は、DNS サーバーを設定するときに必要になります。このドメイン名は、あくまでこのネットワーク内だけで有効なドメイン名で、外部の DNS とは隔離された状態にあります。

本教科書では、2 台の仮想マシンのために example1.test と example2.test の 2 つのドメイン名を使用します。また、DNS による名前解決を行うためにもう 1 台の仮想マシンを作成し、test ドメインとして設定します。

マシン	設定値
1 台目	example1.test
2 台目	example2.test
3 台目	test

1.9.2 ホスト名

自分の PC に設定するホスト名です。ドメイン名に合わせて host1 と host2 を設定します。test ドメイン用には host0 を設定します。

ホスト名は、ドメイン名と合わせて FQDN(Fully Qualified Domain Name) で記述する場合もあります。FQDN 表記の場合には、それぞれ以下のようになります。

マシン	設定値
1 台目	host1.example1.test
2 台目	host2.example2.test
3 台目	host0.test

1.9.3 IP アドレス

IP アドレスは、VirtualBox のブリッジ接続に合わせて、それぞれ以下のように設定しています。

※お使いの環境に合わせて設定ください。

マシン	設定値
1 台目	192.168.1.11
2 台目	192.168.1.12
3 台目	192.168.1.13

1.9.4 サブネットマスク

サブネットマスクは、IP アドレスのネットワーク部とホスト部を分ける値です。

本教科書では 255.255.255.0(/24) とします。

1.9.5 ネットワークアドレス

ネットワークアドレスは、PC が含まれているネットワーク全体を示すアドレスです。

本教科書では 192.168.1.0 となります。

1.9.6 デフォルトゲートウエイ

異なるサブネットとの通信に必要な値です。

ネットワークの設定を確認の上、適切なデフォルトゲートウェイのアドレスを確認、設定してください。

1.9.7　DNS サーバーアドレス

ホスト名と IP アドレスの対応を解決する、DNS（ドメインネームシステム）という機構があります。DNS を利用するためには DNS サーバーの IP アドレスが必要です。

本教科書の第 5 章で実際に DNS サーバーを設定し動作させます。実習では、まず自分自身（ローカル）で動作させている DNS サーバーを参照するため、DNS サーバーアドレスを自分自身の IP アドレスとします。

マシン	設定値
1 台目	192.168.1.11
2 台目	192.168.1.12
3 台目	192.168.1.13

1.10　ネットワーク設定まとめ

各仮想マシン毎のネットワークの設定は以下のようになります。

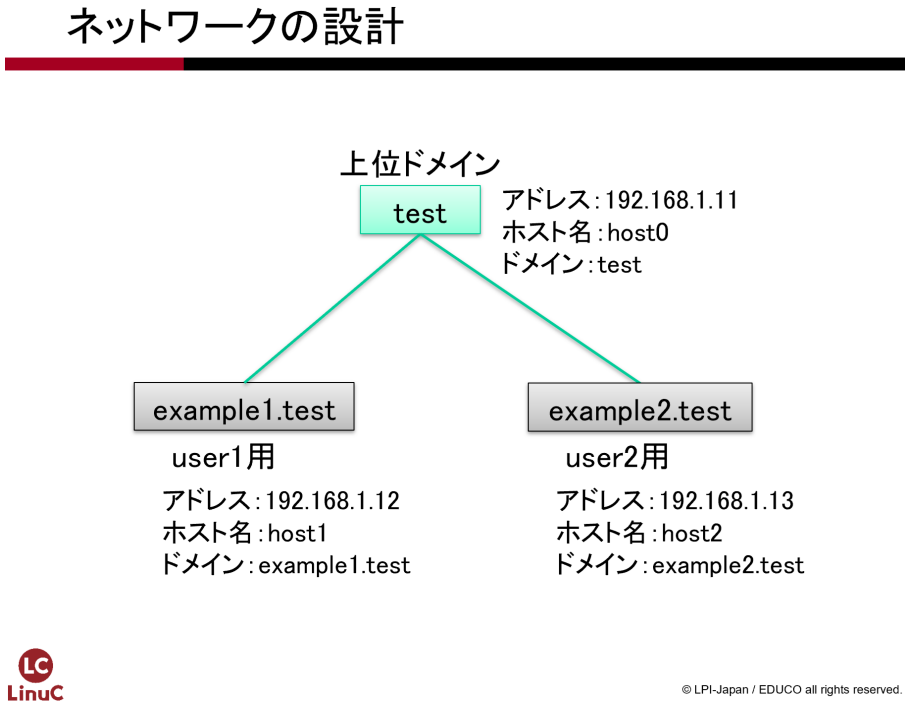

図 7: ネットワークの設計

1.10.1　1 台目の仮想マシン

設定項目	設定値
ホスト名	host1.example1.test
IP アドレス	192.168.1.12
サブネットマスク	255.255.255.0(/24)
ネットワークアドレス	192.168.1.0
デフォルトゲートウエイ	192.168.1.1
DNS サーバーアドレス	192.168.1.12

1.10.2　2 台目の仮想マシン

設定項目	設定値
ホスト名	host2.example2.test

設定項目	設定値
IP アドレス	192.168.1.13
サブネットマスク	255.255.255.0(/24)
ネットワークアドレス	192.168.1.0
デフォルトゲートウエイ	192.168.1.1
DNS サーバーアドレス	192.168.1.13

1.10.3 3 台目の仮想マシン

設定項目	設定値
ホスト名	host0.test
IP アドレス	192.168.1.11
サブネットマスク	255.255.255.0(/24)
ネットワークアドレス	192.168.1.0
デフォルトゲートウエイ	192.168.1.1
DNS サーバーアドレス	192.168.1.11

2 VirtualBox のインストールと仮想マシンの作成

第 2 章では、実習環境の準備として VirtualBox をインストールして仮想マシンを作成します。

2.1 用語集

仮想化技術

ハードウェアの機能をソフトウェアで実現する技術。仮想マシンや仮想ネットワーク、仮想ストレージなど様々な種類の技術が存在しています。

仮想マシン

仮想化技術を利用して、コンピュータのハードウェアをソフトウェアで実現したもの。OS をインストールし、様々なアプリケーションを実行できます。

仮想ホスト

仮想マシンを実行するマシンのこと。仮想ホストでは通常の OS が動作しており、仮想マシンは通常のアプリケーションのように動作します。

ゲスト OS

仮想マシンにインストールする OS のこと。仮想ホストで動作している OS とは異なる種類の OS をインストールして実行できます。

2.2 仮想化環境とは

仮想化環境とは、仮想化技術を活用して構築、動作させるシステムの環境のことです。仮想化技術には、仮想マシン、仮想ネットワーク、仮想ストレージ、コンテナなどがあります。

2.2.1 仮想化技術とクラウドの関係

現在では、クラウドを活用してシステムを構築、運用することが増えています。多くのクラウドサービスが仮想化技術を活用する事で、効率良くシステムリソースをユーザーに対して提供することでそのサービスを実現しています。クラウドを理解し、活用するためにも、仮想化技術の基礎を理解しておく必要があります。

2.3 仮想マシンとは

仮想マシンは、ソフトウェアで仮想的なマシンを実行する仕組みです。仮想マシンにはゲスト OS をインストールし、アプリケーションを実行できます。

2.3.1 ホスト OS 型とハイパーバイザー型

仮想マシンの実行環境には、ホスト OS 上でアプリケーションのように実行される「ホスト OS 型」と、仮想マシン実行専用のソフトウェアであるハイパーバイザーが動作する「ハイパーバイザー型」に大別されます。

本教科書では、簡単に実行できるホスト OS 型を取り上げています。

2.4 仮想マシンのメリット

仮想マシンの利用には、いくつかのメリットがあります。

2.4.1 ホスト OS と共存できる

仮想マシンはホスト OS 上でアプリケーションのように実行されます。そして同時にその他のアプリケーションも実行できるので、たとえば Web ブラウザで何かを調べながら、ターミナルで仮想マシン上の Linux にログインして操作する、ということが簡単に行えます。

2.4.2 ハードウェアを用意する必要がない

Linux を使ったサーバー構築を学習する際に大きな障害となるのが学習用ハードウェア環境の準備です。ハードウェアを用意できたとしても、Linux をインストールするには設定を変更するなどのハードウェアの知識が要求されます。メーカーの違いによって設定方法も異なるため、学習の助けを得るのも難しくなります。

仮想マシンではそのような前提知識が要求されることが少ないので、容易に Linux をインストールして学習環境を構築できます。

2.4.3 柔軟性

仮想マシンは必要に応じて簡単に CPU 数やメモリ容量、ディスク容量などを変更できる柔軟性を備えています。

2.4.4 OS 選択

仮想マシンにはそれぞれ別々の OS をインストールできます。たとえば Linux と Windows がそれぞれインストールされた仮想マシンを同時並行で実行できます。

2.5 仮想マシンのデメリット

仮想マシンの利用には、メリットだけでなくデメリットもあります。

2.5.1 速度

仮想マシンは、コンピュータの動作をソフトウェアで実現しているので、ハードウェアそのもので OS やアプリケーションを動作させるよりも動作が遅くなります。

2.5.2 リソースの消費

仮想マシンを複数同時に実行することができますが、その分だけ CPU やメモリ、ストレージなどのリソースを消費します。ホスト OS を快適に動かすのにもリソースが必要となるため、使用するハードウェアにはより多くのリソースを用意する必要があります。

2.6 仮想マシンのリソース

仮想マシンを作成する際に、CPU やメモリ、ネットワークインターフェースなどのリソースを利用者自らが調整する必要があります。物理マシンを利用する場合にも同様の作業が必要ですが、仮想マシンの場合、ある程度後から変更できるメリットがあります。まずはデフォルト設定のまま動かしてみて、後から必要に応じて変更しても構いません。

2.6.1 CPU

仮想マシンには必要なだけの数の CPU を割り当てられます。ただし、CPU をホスト OS やその他の仮想マシンとの間で取り合うことになるため、沢山割り当てればよいというわけではありません。

2.6.2 メモリ

仮想マシンに設定したメモリ容量は、その分だけホストマシンから仮想マシンに対して割り当てられます。ホストマシンにはホスト OS だけでなく、仮想マシンに割り当てられるだけの容量のメモリを搭載しておく必要があります。

2.6.3 ディスク

HDD や SSD にあたる仮想ディスクは、仮想ディスクファイルという形でホスト OS 上に作成されます。可変式ファイルの場合には、仮想マシンが使用した分だけのサイズのファイルになります。固定式ファイルの場合には、作成時に設定しただけの容量のファイルが作成されます。可変式にすることで、ホストマシンのディスク使用を節約できます。

2.6.4 光学ドライブ

仮想マシンに割り当てられた仮想光学ドライブは、物理的な光学ドライブにセットされた CD/DVD メディアを割り当てる他、ISO イメージを割り当てることができます。仮想マシンに OS をインストールするのが容易な理由の一つです。

2.6.5 ネットワーク

仮想マシンに割り当てられるネットワークインターフェースを割り当てることができます。仮想マシンソフトウェアによって機能が異なるため、それぞれの機能に応じた利用が必要になります。

2.7 VirtualBox のインストール

本教科書の実習では、仮想マシンソフトウェアである「VirtualBox」上に Linux を導入して、その環境上に様々なサーバーを導入し実際に動作させます。

2.7.1 使用するコンピュータの仮想化支援技術の有効化

VirtualBox を実行するには、使用するコンピュータが搭載しているプロセッサの仮想化支援技術が有効になっている必要があります。仮想化支援技術は、Intel の CPU では Intel VT、AMD の CPU では AMD-V と呼ばれます。

仮想化支援技術を有効にするには、コンピュータの UEFI 設定画面で設定を行います。設定方法の詳細は、使用するコンピュータの説明書などを確認してください。

2.7.2 VirtualBox のダウンロード

VirutalBox は、次の URL からダウンロードできます。

```
https://www.virtualbox.org/
```

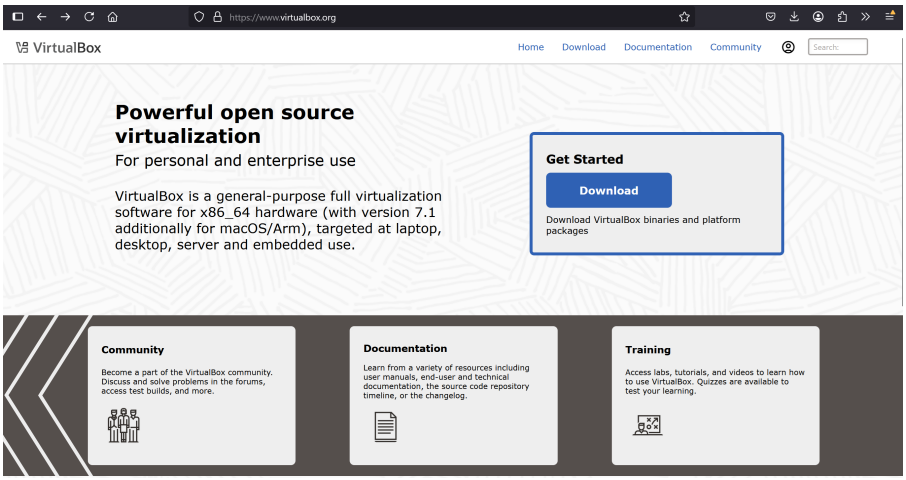

図 8: VirtualBox の Web サイト

ホスト OS の種類に合わせてダウンロードを行えます。今回は Windows 環境にインストールします。左のメニューから Downloads をクリックし、VirtualBox binaries から Windows hosts をクリックして、インストーラをダウンロードします。

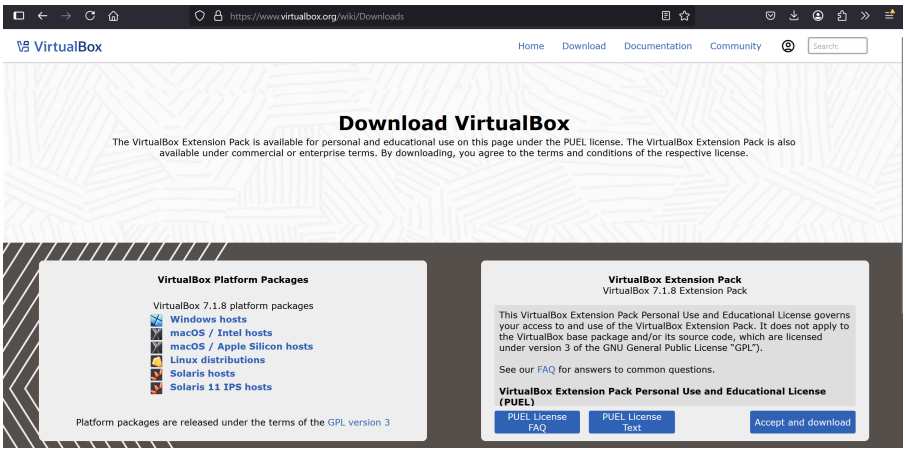

図 9: VirtualBox のダウンロードサイト

今回はバージョン 7.1.8 を使用しています。

2.7.3 VirtualBox のインストーラの実行

VirtualBox のインストーラをダウンロードが完了したら、インストーラを実行してインストールします。

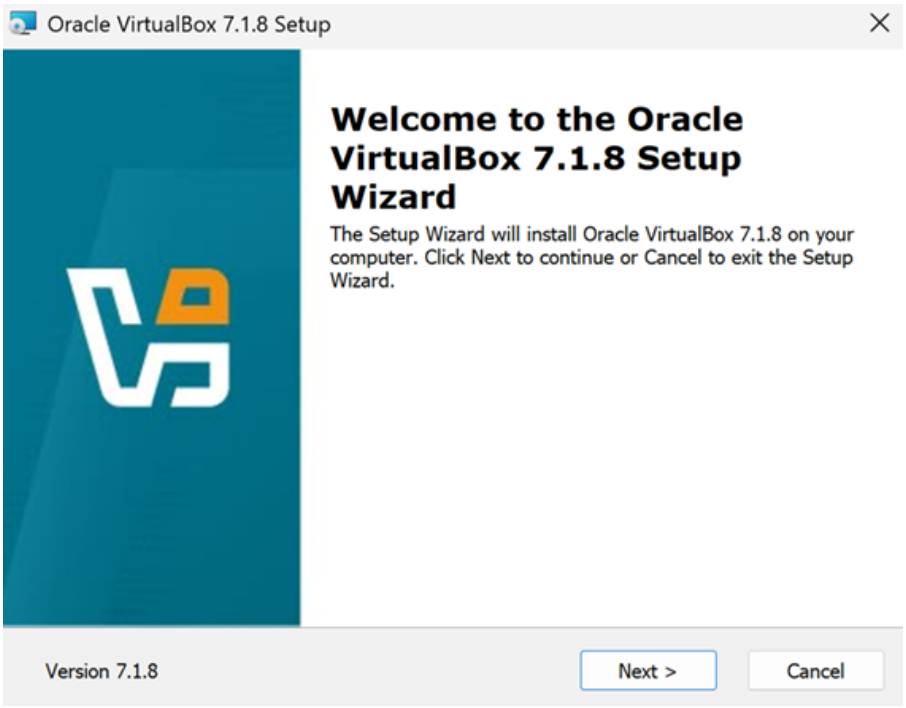

図 10: VirtualBox のインストーラ画面

インストール時の特別な設定は行いませんので、インストーラの指示に従ってインストールします。インストーラが仮想ネットワークのインストールについて、また Python との連携について確認を取ってきますが、そのまま進めても問題ありません。

インストール終了後、Finish ボタンをクリックするとインストーラが終了し、VirtualBox が起動します。

2.8 VirtualBox の起動

VirtualBox を起動します。

インストーラが自動で起動した状態ではない場合には、デスクトップ上に追加された VirtualBox をダブルクリックして起動します。

VirtualBox が起動すると、VirtualBox マネージャーが表示されます。

2.8.1 VirtualBox マネージャー

VirtualBox マネージャーは、VirtualBox による仮想マシン実行環境全体を管理するツールです。主に以下のことが行えます。

- VirtualBox 自身の各種設定
- 仮想マシンの管理
- メディア（ISO イメージ等）の管理
- 仮想ネットワークの管理

2.9 ホストキーによるホスト OS の操作への復帰

仮想マシンをマウスで操作している際に、ホスト OS にマウス操作を戻したくなった場合にはホストキーを押します。デフォルトではキーボード右側の Ctrl キーがホストキーに設定されています。ホストキーの設定は、仮想マシンのウインドウの右下にも表示されています。

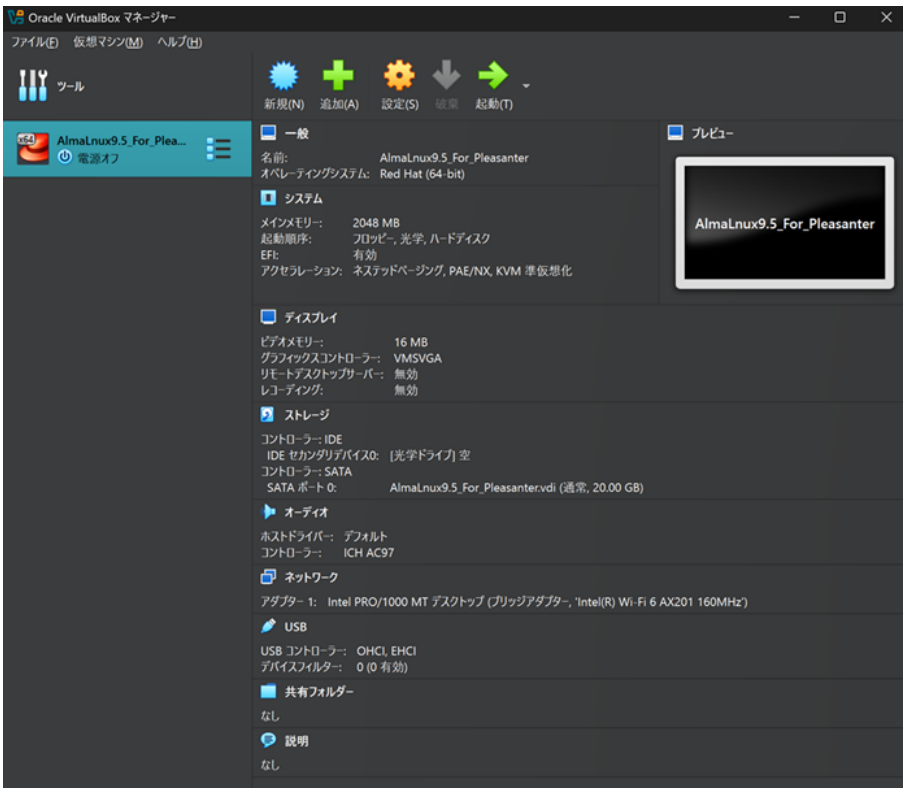

図 11: VirtualBox マネージャーの画面

2.9.1 ホストキーが押せない場合のホスト OS の操作への復帰

キーボードによっては右 Ctrl キーが無いため、仮想マシンの操作から抜け出せなくなります。そのような場合になった場合には、OS をシャットダウンして仮想マシンを停止すれば操作をホスト OS 側に戻すことができます。

2.9.2 ホストキーの変更

VirtualBox マネージャーの環境設定から、ホストキーを変更できます。

環境設定は、VirtualBox マネージャーの「ファイル (F)」メニューから「環境設定 (P)」を選択して呼び出します。「環境設定」ウインドウが表示されたら、左側の設定項目から「入力」を選択し、「仮想マシン (M)」タブを選択します。一番最初にある「ホストキーの組み合わせ」の「ショートカット」部分をクリックします。新たにホストキーにしたいキーを押すことで、ホストキーがそのキーに切り替わります。

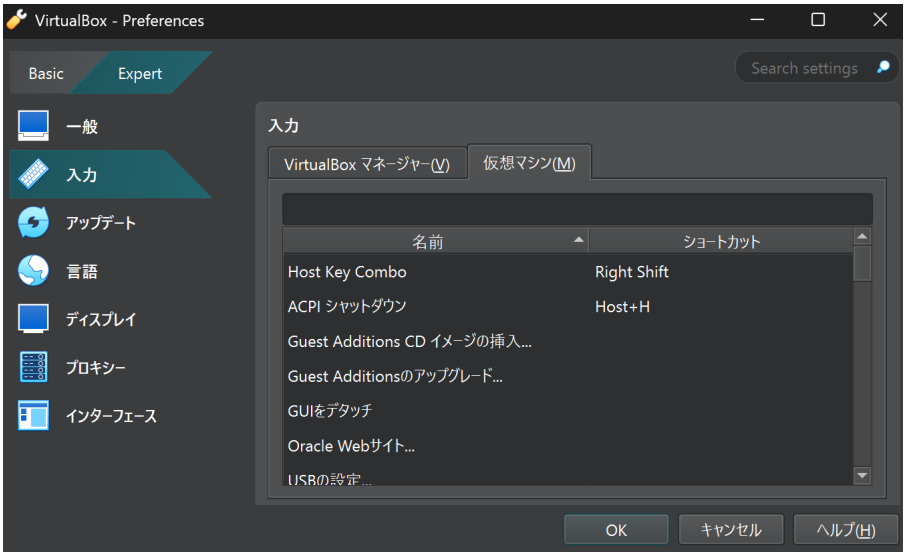

図 12: ホストキーの設定画面

ホストキーに設定できるのは Ctrl キー、Alt キー、Windows キー、Shift キー（単独設定不可）です。左の Ctrl キーをホストキーに設定してしまうと、コマンドライン操作などの際に重複してしまうので、避けた方がよいでしょう。複数のキーを同時に押す組み合わせをホストキーにすることもできるので、2 つないし 3 つのキーの組み合わせをホストキーとして設定するとよいでしょう。

2.10 仮想マシンの作成

仮想マシンを作成します。

VirtualBox マネージャーで「新規 (N)」ボタンをクリックし、新しい仮想マシンの作成を開始します。各種設定を対話形式で設定していくガイド付きモードと、1 つの画面で設定できるエキスパートモードが選択できます。

ここではデフォルトのガイド付きモードで作成します。

2.10.1 仮想マシンの名前と OS の設定

まず最初に、仮想マシンの名前と OS の種類などを設定します。これらの設定は後から変更することもできるので、その他の項目で必要なものは後ほど設定を行います。また、第 1 章でダウンロードした ISO イメージを指定します。

設定する項目と、設定する内容は以下の通りです。

項目	設定値
名前	host1.example1.test
タイプ	Linux
バージョン	Ubuntu (64-bit)
ISO イメージ	ubuntu-24.04.2-live-server-amd64.iso

設定したら、「次へ」ボタンをクリックします。

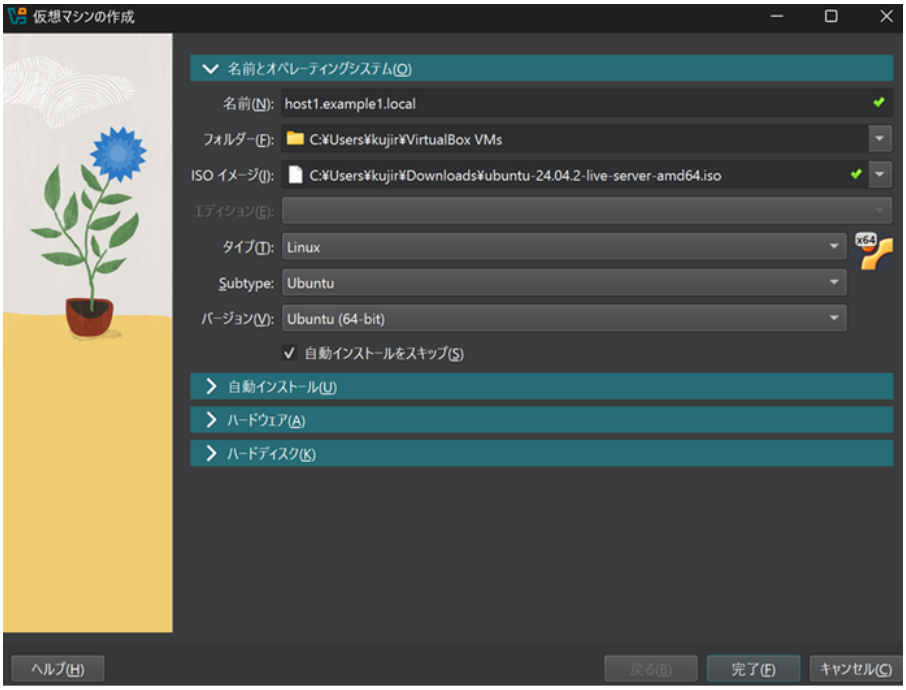

図 13: 仮想マシンの名前と OS 選択の画面

2.10.2 仮想マシンのハードウェアの設定

仮想マシンのハードウェアの設定として、メインメモリーの容量とプロセッサの数を設定します。また、EFI の設定も行います。

項目	設定値
メインメモリー	2048MB
Processors	1

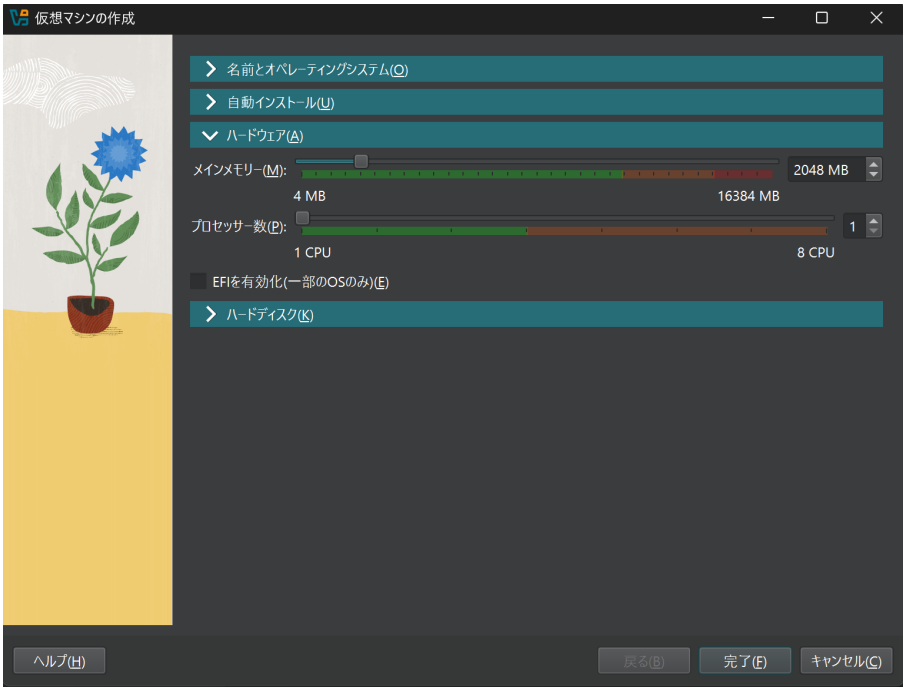

図 14: 仮想マシンのメモリや CPU 設定の画面

メインメモリーの容量は、使用しているコンピュータが搭載しているメモリ容量以下に制限されます。仮想マシンにある程度多めのメインメモリーを割り当てることで快適に動作させることができます。1024MB でも動作しますが、余裕があれば 2048MB（2GB）以上を割り当ててもよいでしょう。

プロセッサの数は、使用しているコンピュータが搭載しているプロセッサのコア数以下に制限されます。また、プロセッサによっては仮想 CPU 機能により見かけ上コア数が 2 倍になっている場合もあります。デフォルトの 1 つでも動作しますが、余裕があれば 2 つ割り当ててもよいでしょう。

設定しましたら、「次へ」ボタンをクリックします。

2.10.3 仮想ハードディスクの設定

OS のインストール先となる仮想ハードディスクの設定をします。

項目	設定値
Disk Size	20.00GB

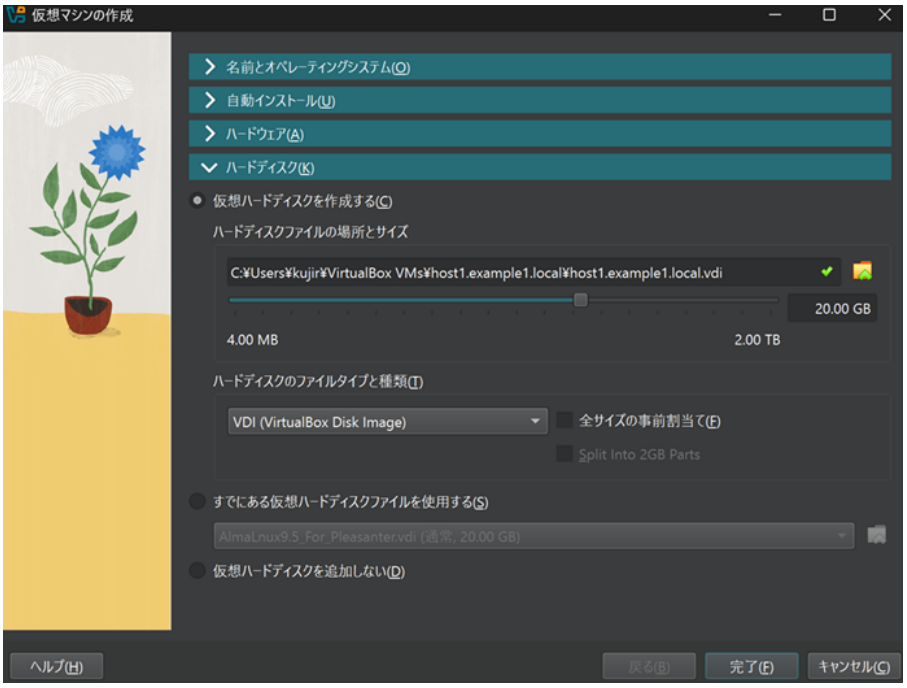

図 15: 仮想ハードディスクの設定画面

仮想ハードディスクの容量は、使用しているコンピュータが搭載しているストレージの容量以下に制限されます。また、「全サイズの事前割当て」をチェックしない限り、仮想ハードディスクの容量はゲスト OS が使用した分だけしか消費しませんので、設定は最大消費容量を指定することになります。

OS のインストールとサーバーの設定だけではそれほど沢山の容量を必要としませんが、実際に運用するサーバーを作る場合には必要な容量を見積もって設定する必要があります。また、後から仮想ハードディスクを追加することもできるので、追加方法について学習してもよいでしょう。

設定しましたら、「完了」ボタンをクリックします。

2.10.4 仮想マシンの設定を確認

最後に仮想マシンの設定を確認します。ここまでの手順で設定した内容が反映されているかどうか、あらためて確認します。

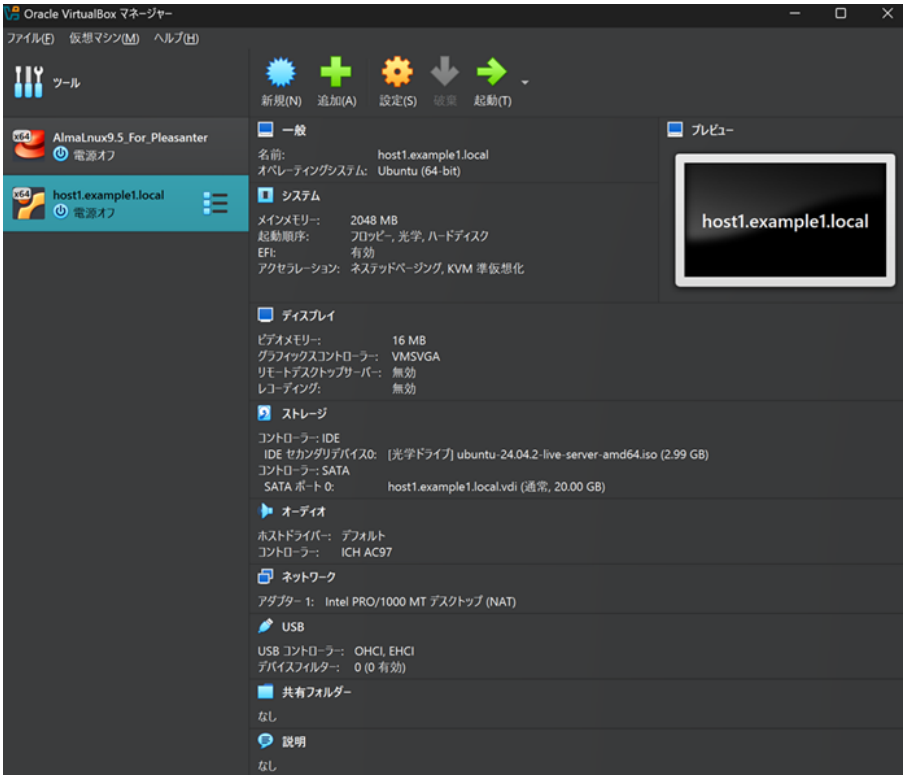

図 16: 仮想マシンの設定確認画面

VirtualBox マネージャーに、新しく作成した仮想マシンが追加されたことを確認します。

2.11 仮想マシンの設定方法

仮想マシンの設定を確認したり、変更するには、VirtualBox マネージャーで仮想マシンを選択し、右側に表示される各種設定情報から変更が行えます。設定項目の見出し部分（「一般」や「システム」など）をクリックすると、その項目の設定画面が表示されます。

図 17: 仮想マシンの設定画面

3　Linux のインストールと設定

第 3 章では、VirtualBox で作成した仮想マシンに Ubuntu 24.04.2 LTS(Server) をインストールします。インストール後、初期設定とネットワーク接続の確認を行います。

3.1　用語集

ISO イメージ

CD-ROM や DVD-ROM は ISO9660 という形式でファイルを保存しています。この ISO9660 形式のディスクをそのまま 1 つのファイルに保存したものを ISO イメージと呼びます。

ブートローダー

OS の起動時に最初に読み込まれるプログラムです。ブートローダーが Linux カーネルを読み込んでシステムが起動します。

パーティション

Linux はディスクを複数の区画（パーティション）に分割して利用します。データを保存するパーティションの他、仮想メモリが利用するスワップ用のパーティションなどがあります。

NAT

NAT（Network Address Translation）は IP 通信時に IP アドレスを変換して通信を行う仕組みです。IPv4 ではインターネットに直接接続できるグローバル IP アドレスが枯渇しているため、1 つのグローバル IP アドレスを複数のホストで共有して利用できるように NAT を活用しています。

root

Linux のシステム管理者のユーザー名です。

3.2　仮想マシンの起動

VirtualBox マネージャーで仮想マシンを選択し、「起動」をクリックします。別ウインドウで仮想マシンが起動します。

3.2.1　ISO イメージファイルのマウント

あらかじめ仮想マシンの仮想光学ドライブに ISO イメージを読み込ませる設定をしていない場合、起動用の ISO イメージを設定するダイアログが表示されます。

画面の指示に従って、ISO イメージのファイルを指定し、「マウントとブートのリトライ」ボタンをクリックして起動します。

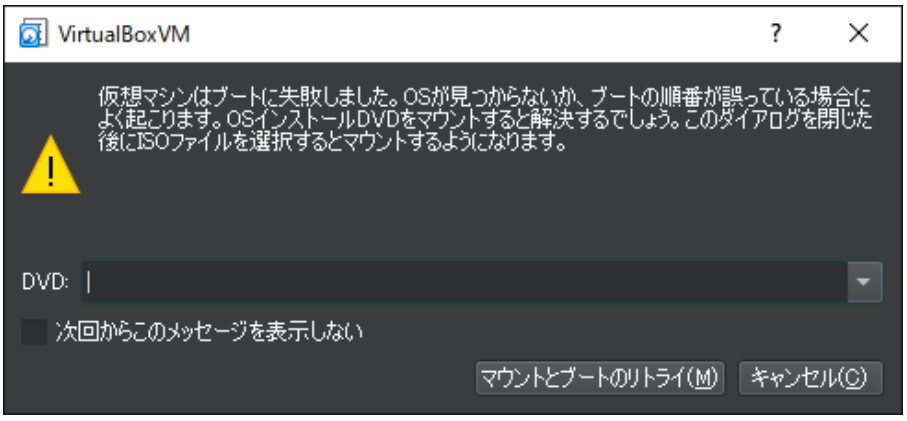

図 18: マウントとブートのリトライダイアログ画面

3.3　OS のインストール

以下の手順に従って、OS のインストールを行います。

3.3.1　インストーラ起動オプションの選択

OS インストール用の ISO イメージを読み込んで起動すると、最初に GRUB というブートローダーが起動します。ここでインストーラの起動オプションを選択できます。

デフォルトの「Try or Install Ubuntu Server」で起動し、インストールを行います。すでに選択されているので、Enter キーを押します。

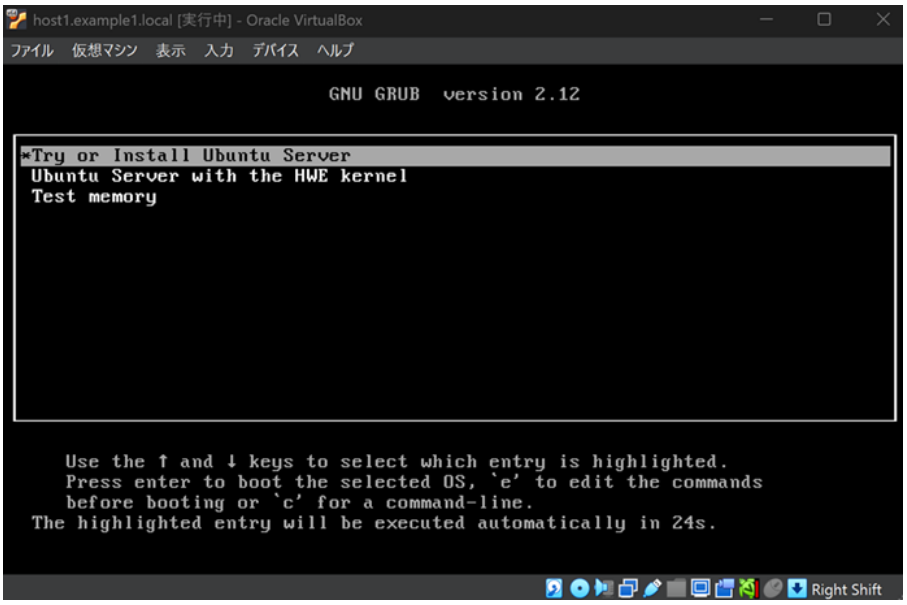

図 19: GRUB 画面

3.3.2　言語選択

言語の選択画面が表示されるので、メニューから「English」を選択します。

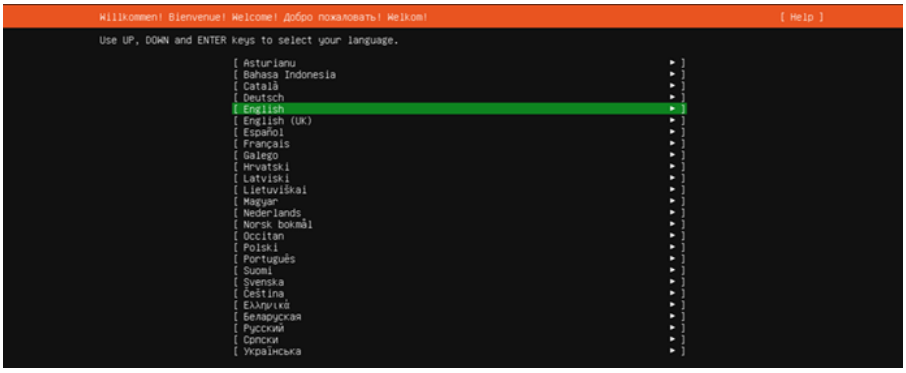

図 20: 言語選択画面

3.3.3　キーボードレイアウト選択

キーボードレイアウトの選択画面が表示されるので、メニューから「Japanese」を選択します。

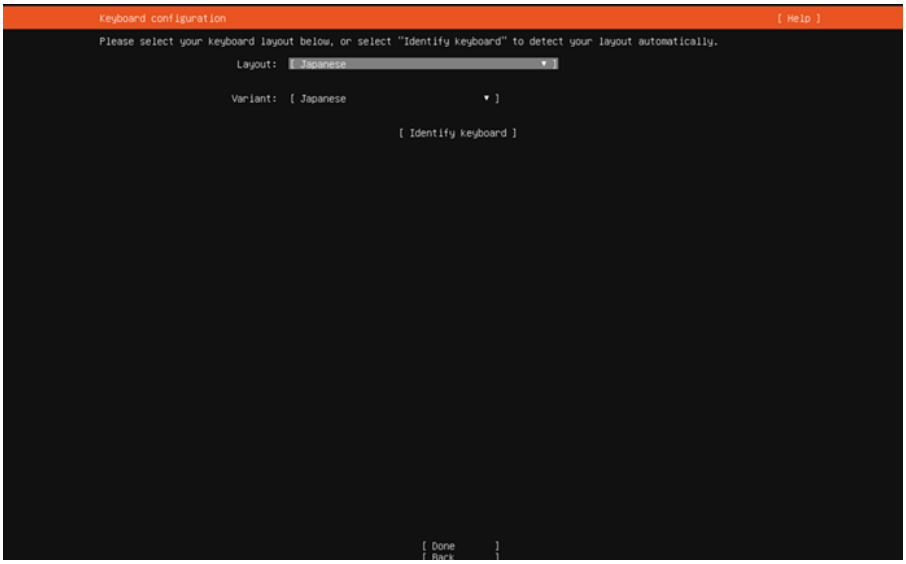

図 21: キーボードレイアウト選択画面

3.3.4　インストールタイプ選択

インストールタイプの選択画面が表示されるので、メニューから「Ubuntu Server」を選択します。

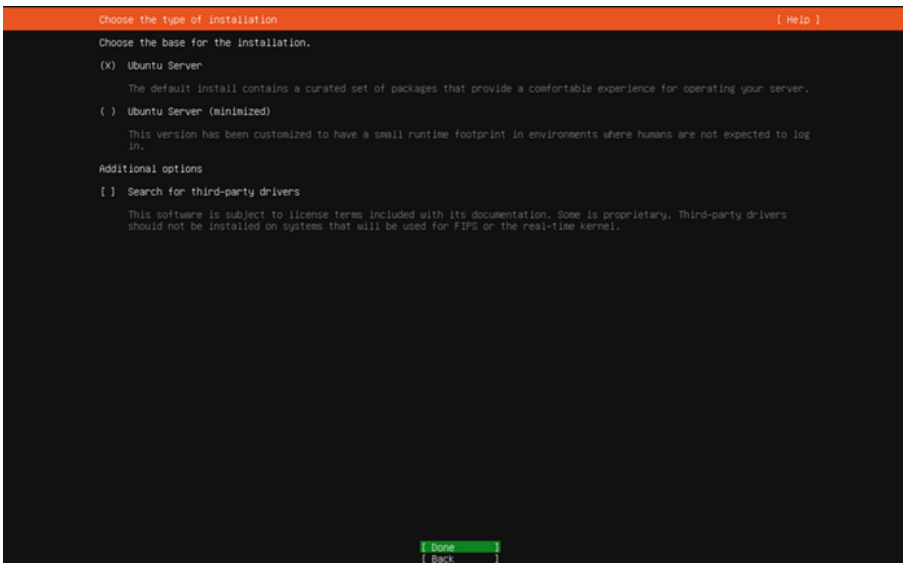

図 22: インストールタイプ選択画面

3.3.5 ネットワーク設定

ネットワーク設定画面が表示されるので、認識されたネットワークインターフェイスに DHCP により IP アドレスが付与されていることを確認します。なお、サーバとして使用するため、enp0s8 は固定 IP アドレスを設定します。

固定 IP アドレスの設定方法

1. 「enp0s3 eth」を選択肢、Enter 押下
2. 「Edit IPv4」を選択後、Enter 押下
3. 「IPv4 Method」を選択後、「Manual」に変更
4. 以下のように各項目を設定
   - Subnet: 192.168.1.0/24
   - Address: 192.168.1.12
   - Gateway: 192.168.1.1
   - Name servers: 192.168.1.12

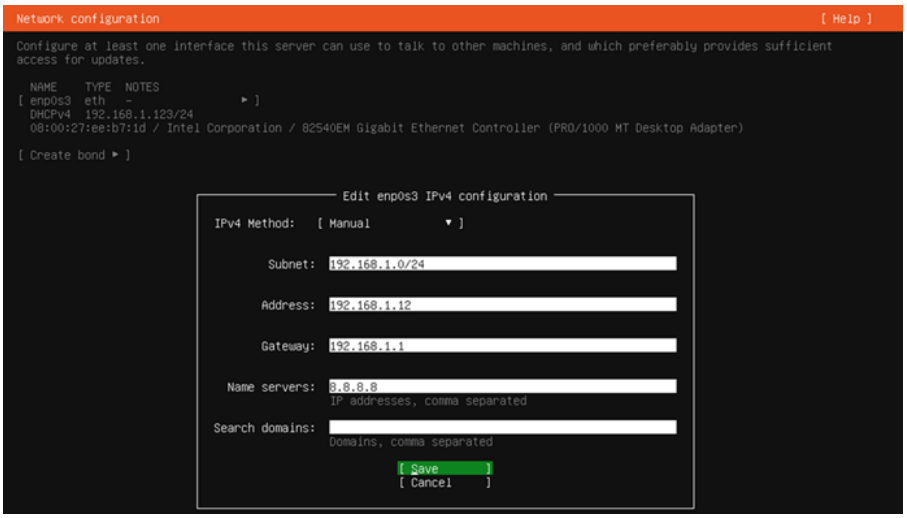

図 23: ネットワーク設定画面

3.3.6　プロキシ設定

プロキシ設定画面が表示されるので、必要に応じてプロキシを設定します。今回の環境では、未設定としています。

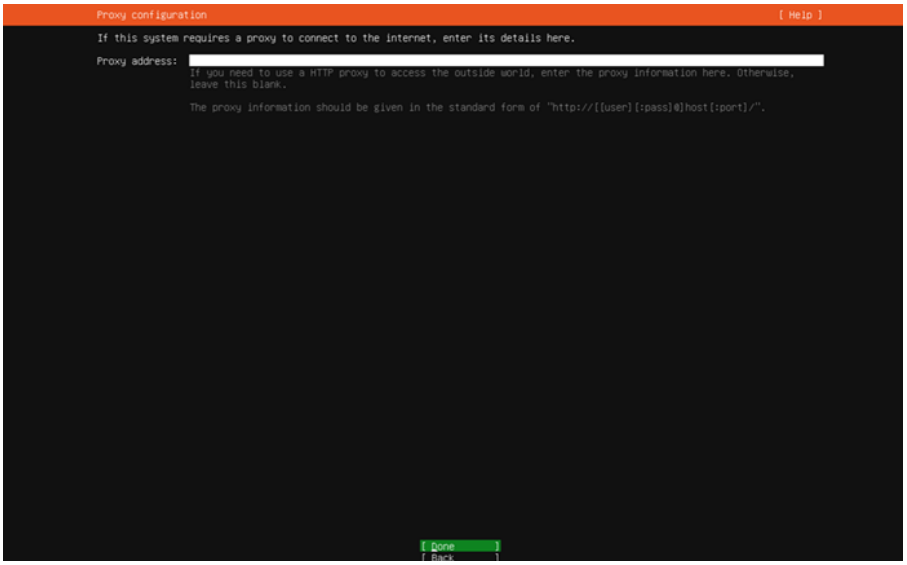

図 24: プロキシ設定画面

3.3.7　ミラーサイト設定

ミラーサイト設定画面が表示されるので、必要に応じてミラーサイトを設定します。今回は、未設定としています。

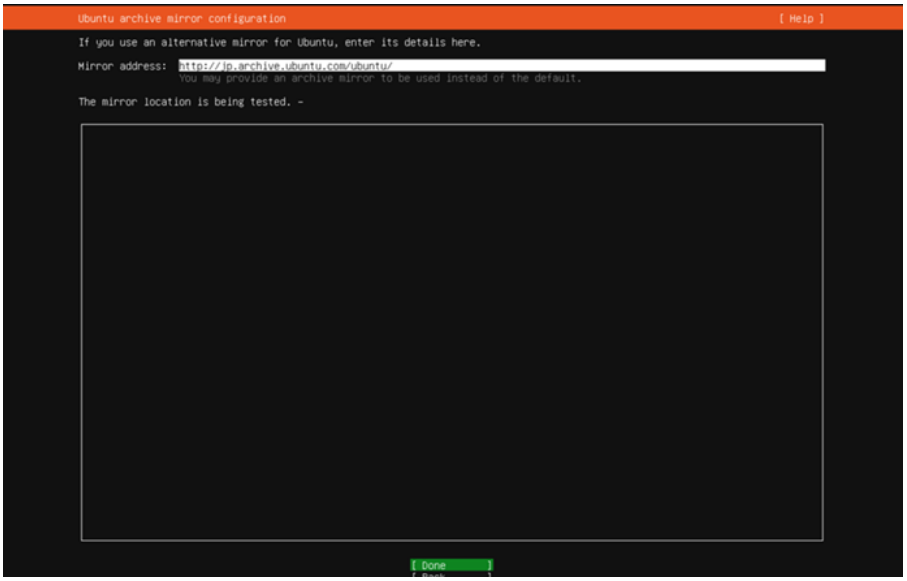

図 25: ミラーサイト設定画面

3.3.8 ストレージ設定

ストレージ設定画面が表示されるので、構築するサーバにで必要となるを設定します。今回は、特にストレージ設定は変更なくデフォルトのまま進めます。

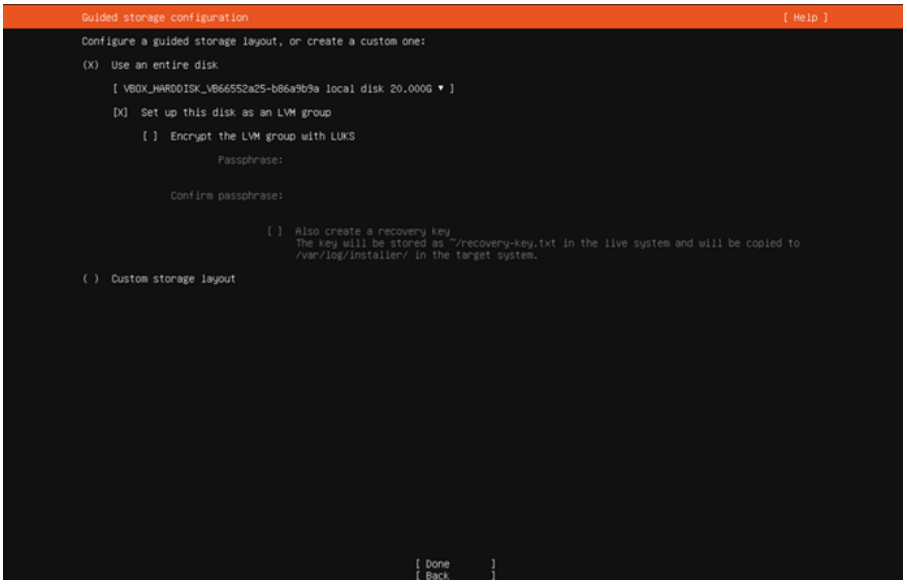

図 26: ストレージ設定画面

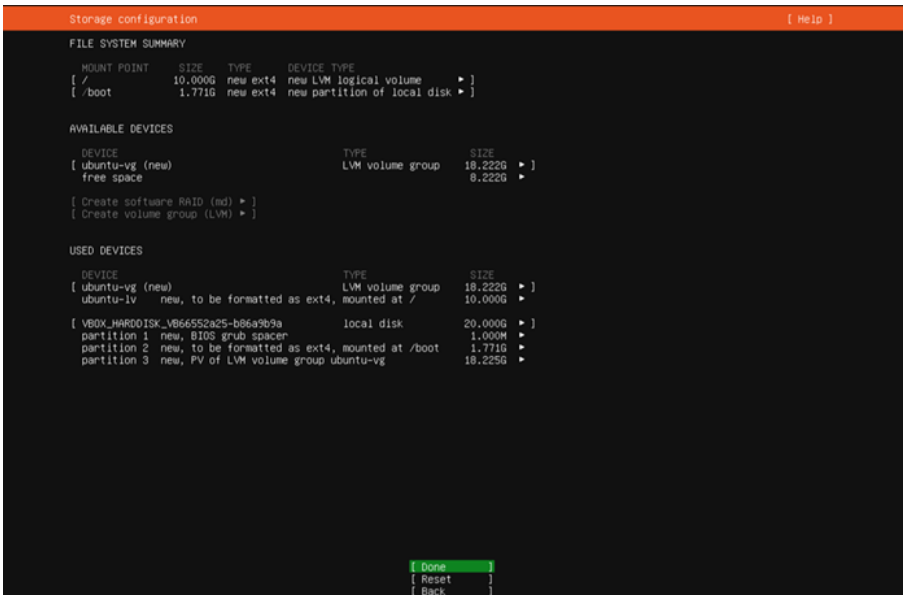

図 27: ストレージ設定画面

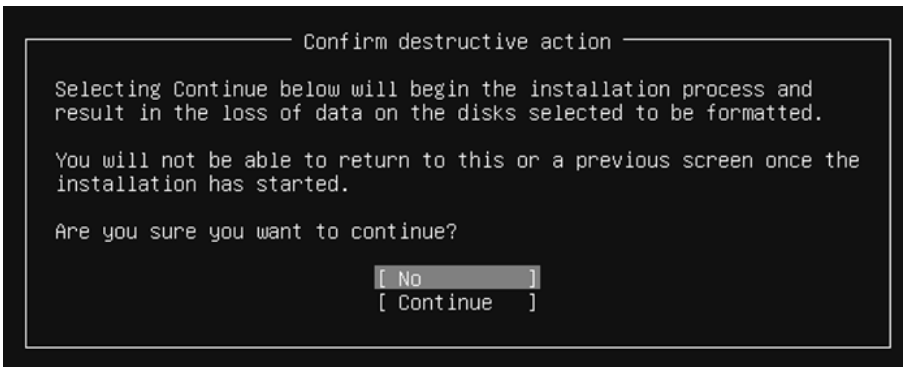

図 28: ストレージ設定画面

3.3.9　サーバ名・ユーザ名・ユーザパスワード設定

サーバ名・ユーザ名・ユーザパスワードの設定画面が表示されるので、それぞれ設定します。今回は、ユーザ名 ubuntu と設定しています。

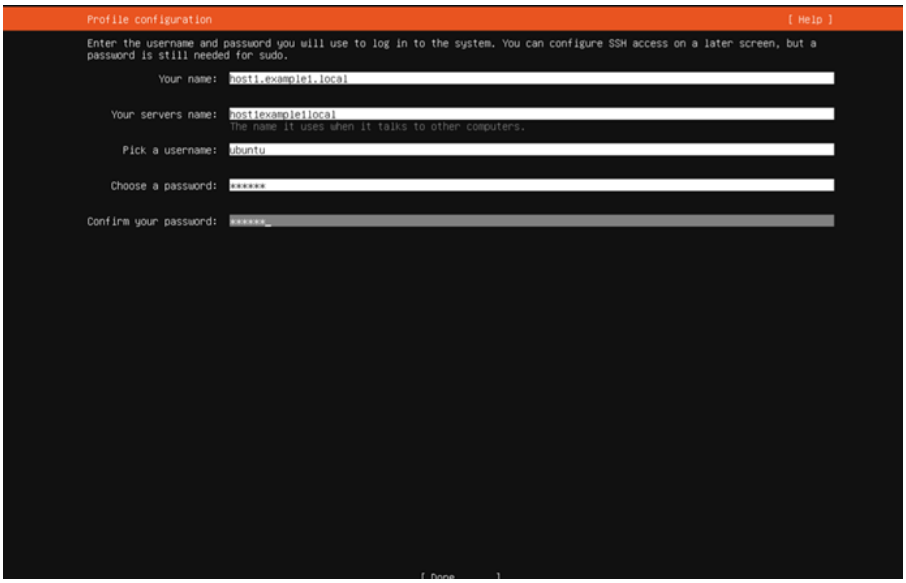

図 29: サーバ名・ユーザ名・ユーザパスワード画面

3.3.10　Ubuntu Pro 選択

Ubuntu Pro(有償サポート) の選択画面が表示されるので、選択します。今回は、サポート無し (Skip for now) としました。

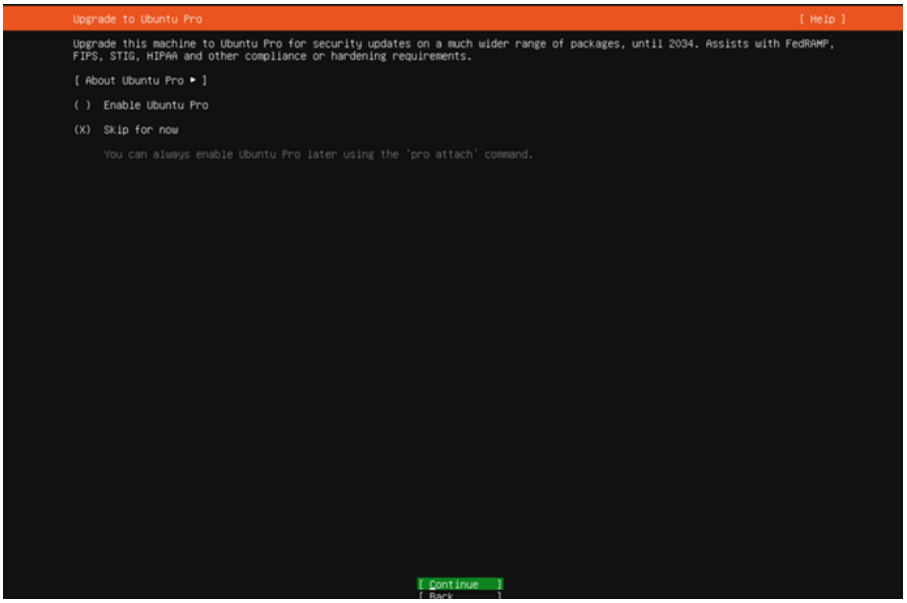

図 30: UbuntuPro 画面

3.3.11　OpenSSh のインストール選択

OpenSSH のインストールを選択します。

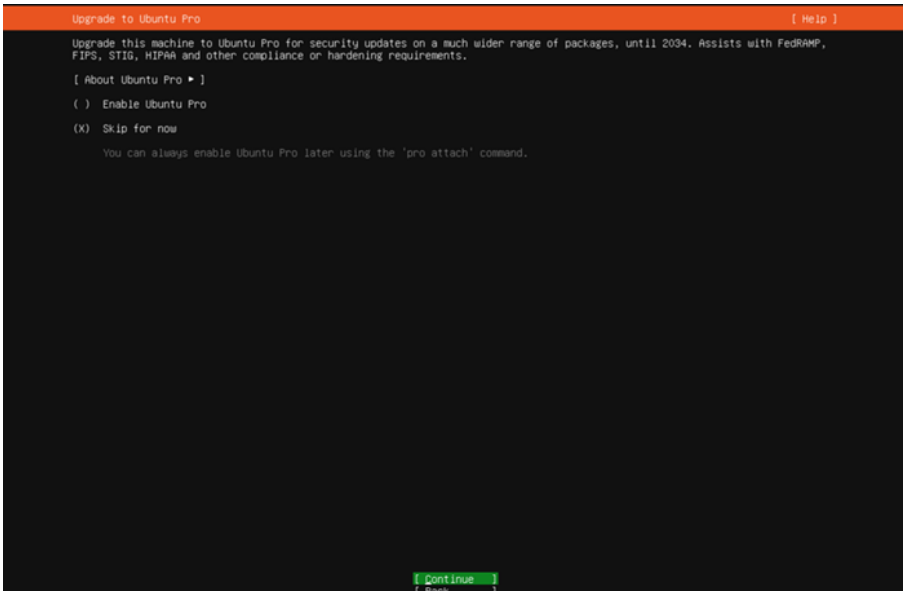

図 31: OpenSSHInstall 画面

3.3.12　追加パッケージ選択

追加するパッケージを選択します。今回は、パッケージの追加はなしとしました。

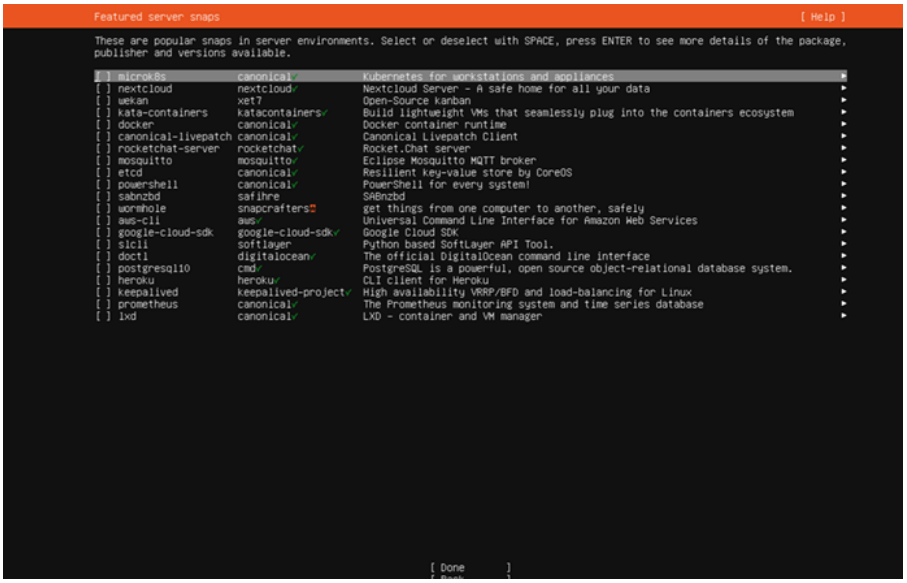

図 32: 追加パッケージ選択画面

3.3.13　インストールプロセスの進捗

インストールプロセスが開始されました。全てのインストールプロセスが終了後、「Installation Cpmplete!」と表示されるので、「Reboot Now」を選択し、Enter を押下します。再起動が行われ、ユーザ名を入力するプロンプトが表示されればインストールは完了です。

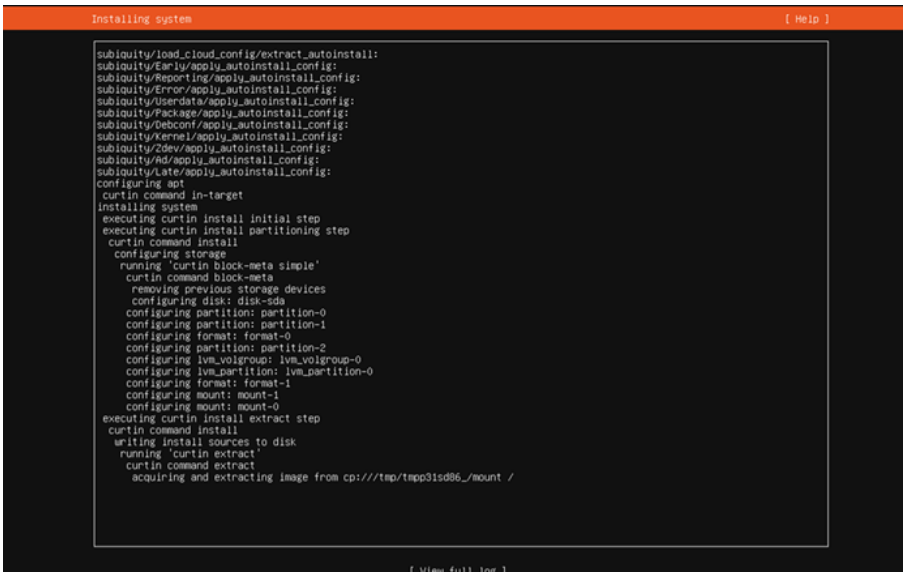

図 33: Process 画面

3.4 ログインとログアウト

Ubuntu 24.04.2 LTS(Server) を使い始めるにはログインを、使い終わったらログアウトを行います。

3.4.1 ログインする方法

「host1example1test login:」というプロンプトのあとに設定したユーザ名 (ubuntu) 入力し Enter 押下、その後表示される「Password:」というプロンプトのあとに設定したパスワードを入力し Enter 押下します。

ログイン出来た場合は、「ubuntu@host1example1test:~$」というプロンプトが表示されます。

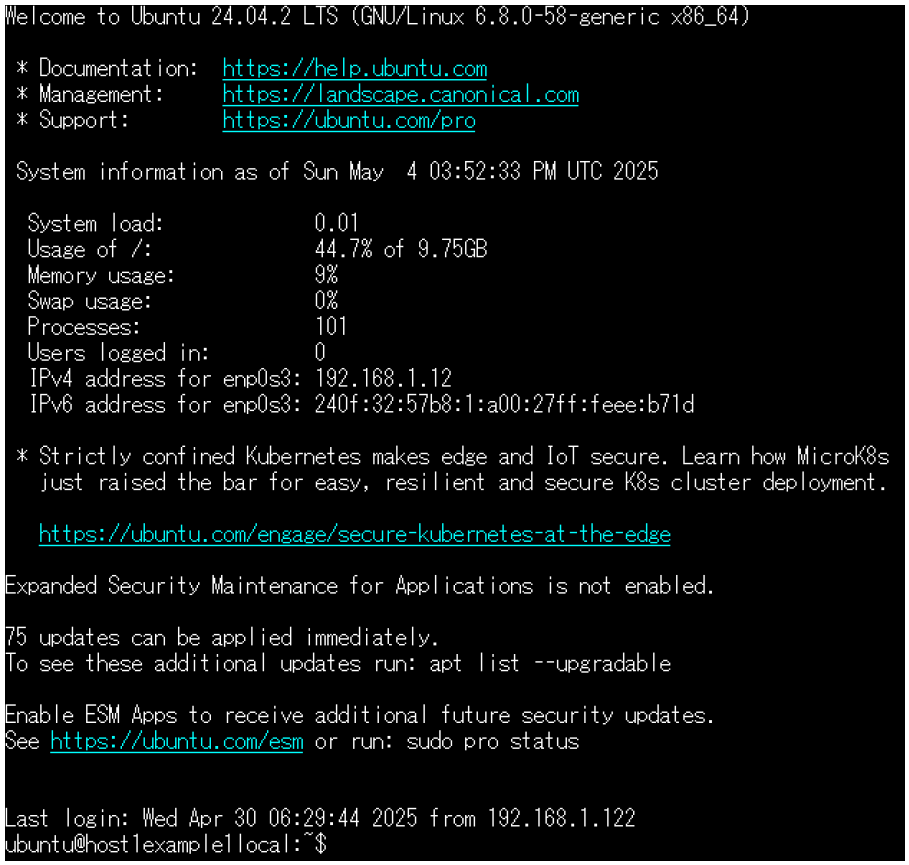

図 34: ログイン画面

3.4.2 ログアウトする方法

ログアウトするには、exit コマンドを実行します。

3.5 コマンドの実行方法

コマンドを実行して Linux を操作するには、ログイン後のプロンプトが表示された状態でコマンドを実行します。また、仮想マシンで直接コマンド実行するのではなく、ホスト OS から SSH を使ってリモート接続して操作することもできます。方法については第 7 章の解説を参考にしてください。

3.6 ネットワーク接続の確認

ネットワークに正しく接続されているかの確認を行います。それぞれについてテストします。

3.6.1 名前解決の確認

dig コマンドを使って、DNS による名前解決を確認します。

```
ubuntu@host1example1test:~$ dig lpi.or.jp

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> lpi.or.jp
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11707
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;lpi.or.jp.                     IN      A

;; ANSWER SECTION:
lpi.or.jp.              300     IN      A       219.94.215.12

;; Query time: 14 msec
;; SERVER: 127.0.0.53#53(127.0.0.53) (UDP)
;; WHEN: Sun May 04 15:54:30 UTC 2025
;; MSG SIZE  rcvd: 54
```

きちんと DNS の名前解決が行えていることが分かります。

3.6.2 インターネットへの接続の確認

Ping コマンドを使って、インターネット上のサーバーへの接続を確認します。

```
ubuntu@host1example1test:~$ ping lpi.or.jp
PING lpi.or.jp (219.94.215.12) 56(84) bytes of data.
64 bytes from 12.215.94.219.static.www232b.sakura.ne.jp (219.94.215.12):
    icmp_seq=1 ttl=54 time=12.9 ms
64 bytes from 12.215.94.219.static.www232b.sakura.ne.jp (219.94.215.12):
    icmp_seq=2 ttl=54 time=15.3 ms
64 bytes from 12.215.94.219.static.www232b.sakura.ne.jp (219.94.215.12):
    icmp_seq=3 ttl=54 time=31.8 ms
64 bytes from 12.215.94.219.static.www232b.sakura.ne.jp (219.94.215.12):
    icmp_seq=4 ttl=54 time=12.6 ms
^C
--- lpi.or.jp ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 12.619/18.172/31.825/7.950 ms
```

ping コマンドの実行を停止するには Ctrl + c を押します。正しく通信できていることがわかります。宛先の IP アドレスが dig コマンドで調べた IP アドレスと同じになっていることも分かります。

ping コマンドでの接続に対する応答を返さないサーバーもあるので、返答が無い場合にはその他のサーバーへの接続も試してみるか、Linux の GUI で Web ブラウザを起動して Web サイトへ接続するなども試してみるとよいでしょう。

3.6.3 ゲスト OS からホスト OS への接続の確認

ping コマンドで、ホスト OS への接続を確認します。ホスト OS 側の IP アドレスは、VirtualBox のネットワーク設定で確認した「192.168.1.12」になります。

```
ubuntu@host1example1test:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
    default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
    group default qlen 1000
    link/ether 08:00:27:ee:b7:1d brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.12/24 brd 192.168.1.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 240f:32:57b8:1:a00:27ff:feee:b71d/64 scope global dynamic mngtmpaddr
        noprefixroute
       valid_lft 266sec preferred_lft 266sec
    inet6 fe80::a00:27ff:feee:b71d/64 scope link
       valid_lft forever preferred_lft forever
```

3.6.4 ホスト OS からゲスト OS への接続の確認

ホスト OS 側でコマンドプロンプトを起動して、ping コマンドでゲスト OS への接続を確認します。ゲスト OS 側の IP アドレスは「192.168.1.12」になります。コマンドプロンプトを起動するには、画面左下の「ここに入力して検索」に「cmd」と入力します。「コマンドプロンプト」が検索されるので、クリックして起動します。

```
Microsoft Windows [Version 10.0.26100.3915]
(c) Microsoft Corporation. All rights reserved.

C:\Users\kujir>ping 192.168.1.12
```

```
192.168.1.12 に ping を送信しています 32 バイトのデータ:
192.168.1.12 からの応答: バイト数 =32 時間 =1ms TTL=64
192.168.1.12 からの応答: バイト数 =32 時間 =1ms TTL=64
192.168.1.12 からの応答: バイト数 =32 時間 <1ms TTL=64
192.168.1.12 からの応答: バイト数 =32 時間 =1ms TTL=64

192.168.1.12 の ping 統計:
    パケット数: 送信 = 4、受信 = 4、損失 = 0 (0% の損失)、
ラウンド トリップの概算時間 (ミリ秒):
    最小 = 0ms、最大 = 1ms、平均 = 0ms
```

確認が行えたら Ctrl+C を入力して動作を停止します。

4 Web サーバーの構築

第 4 章では、ホームページや Web システムを公開するための Web サービスを設定します。パッケージのインストールやシステム管理権限などについても触れます。なお、セキュリティを考慮したネットワークのアクセス制限については第 7 章で紹介します。

4.1 用語集

HTML(HyperText Markup Language)

Web ページを書くためのタグを使って文章を構造的に記述できるマークアップ言語です。他ドキュメントへのハイパーリンクを書いたり、画像を利用したり、リストや表などの高度な表現も可能です。現在では、ページレイアウトなどを定義する CSS（Cascading Style Sheets）や、プログラムを記述できる JavaScript などと組み合わせて高度な Web ページを作成できます。

HTTP(HyperText Transfer Protocol)

Web ブラウザと Web サーバーの間で HTML などのコンテンツ (データ) 送受信に使われる通信手順です。ファイルのリクエスト (要求) とファイルのレスポンス (返送) が組でセッションになります。現在では、SSL/TLS で通信を暗号化するなどしてセキュリティを高めた HTTPS（Hypertext Transfer Protocol Secure）が使用されています。

Nginx

Igor Sysoevs によって開発されたオープンソースソフトウェアでありあり、現在もっとも世界中で利用されている Web サーバーです。Web サーバーとしてだけではなく、リバースプロキシサーバ、ロードバランサーとしても利用できるソフトウェアです。

Apache HTTP サーバー

従来から世界中で利用されている使われている Web サーバーであり、大規模な商用サイトから自宅サーバーまで幅広く利用されています。Apache ソフトウェア財団の Apache HTTP サーバープロジェクトで開発が行われているオープンソースソフトウェアです。

URL(Uniform Resource Locator)

インターネット上のリソースを指定するための記述方法で、ホームページのアドレスやメールのアドレスなどを指定できます。リソースを特定するスキーム名とアドレスを「://」でつないで書きます。

パッケージ

プログラムの本体であるバイナリや設定ファイル、ドキュメントなどを一式まとめてインストール可能なようにまとめたものです。大きなプログラムの場合、本体とライブラリ、各種拡張機能などを別々のパッケージとしてまとめて選択的にインストールできるようにしていることがあります。Linux ディストリビューションは、このパッケージの集合体と考えることができます。

システム管理権限

Linux は複数のユーザーが同時に利用できるマルチユーザー型の OS です。システム管理権限は、システム全体に対する変更などが行える権限で、一般的なユーザーには与えられていません。Linux では root ユーザーか、sudo コマンドを使える権限が与えられたユーザーのみシステム管理権限を行使できます。本教科書で利用する Linux ディストリビューションである Ubuntu では、後者の sudo コマンドによる管理者権限の操作を行います。

4.2　Web サーバーの仕組み

Web システムとは、インターネット環境で最も代表的なクライアントサーバー型のシステムで、Web サーバーとクライアントの Web ブラウザとで構成されます。Web サーバーは要求されたファイルを Web クライアントに提供し、クライアントは受け取ったファイルを表示します。

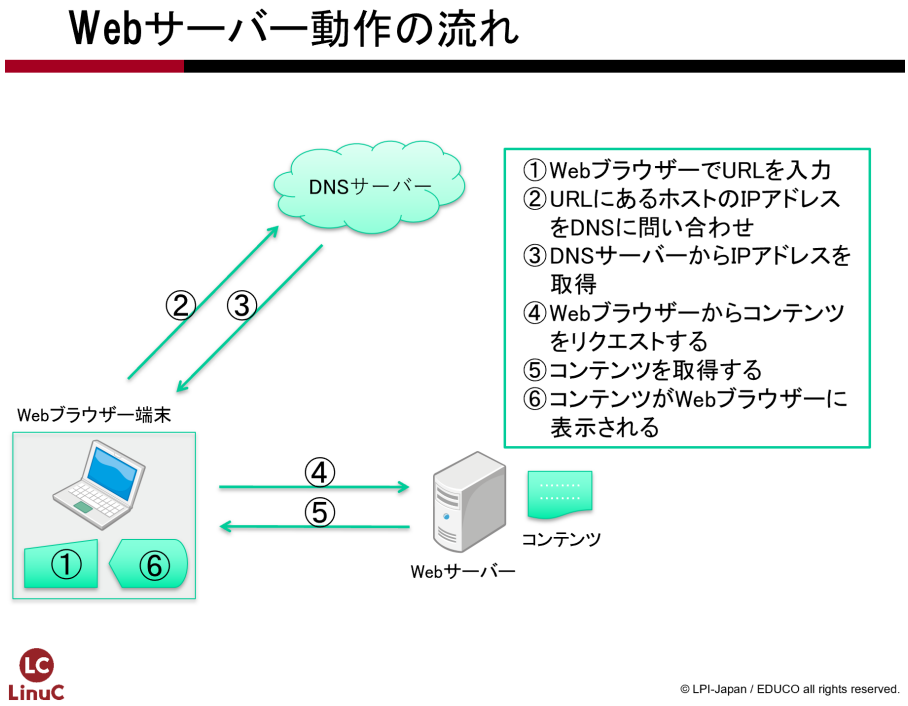

図 35: Web サーバー動作の流れ

提供される情報はテキストから画像や動画と幅広く、クライアントが対応しているデータならば広く扱えます。Web システムの文章データとしては HTML が一般的に使われています。

4.3　パッケージのインストール

Linux で Web サーバーを動作させるには、まず Web サーバーのインストールを行います。ソースコードをコンパイルして動かすこともできますが、パッケージを使えば簡単にインストールが行えます。実習で使用している Ubuntu 24.04.2 LTS(Server) では DEB 形式のパッケージを使用しており、パッケージ管理ツールとして apt コマンドが使用できます。

4.3.1　apt コマンド

apt コマンドを使うと、パッケージのインストールや削除、アップデートなどが行えます。

4.3.2　apt コマンドと apt-get コマンド

apt コマンドは、従来は apt-get コマンド・apt-cache などが行っていたパッケージ管理を置き換えるコマンドです。apt-get install コマンド等、使用しているサブコマンドの多くは apt コマンドでも実行できます。

4.3.3　sudo コマンドによる root 権限の取得

apt コマンドによるパッケージのインストールは、システムの変更を伴うため管理者である root ユーザーの権限が必要になります。コマンド実行時に root 権限を取得するには、sudo コマンドを使用します。OS インストール時、あるいは初期設定時に作成したユーザーには sudo コマンドを実行する権限が与えられています。

4.3.4　apt コマンドが参照するパッケージリポジトリ

apt コマンドは、インストールに使用するパッケージをリポジトリと呼ばれる場所から取得します。通常はインターネット上にリポジトリサーバーが用意されており、インターネット経由でパッケージをダウンロードします。今回はインターネットにアクセスできる前提で作業を行います。

4.3.5 プロキシーが必要な場合には

インターネットへのアクセスにプロキシーを経由する必要がある場合、apt コマンドの設定でプロキシを設定することでアクセスできるようになります。設定方法は、別途マニュアル等を参照してください。

4.3.6 インターネットにアクセスできない環境でのパッケージ管理

インターネットにアクセスできない環境ではインターネット上のリポジトリにアクセスできないため、以下のような方法でインストールを行う必要があります。本教科書ではインストール時に必要なソフトウェアを選択してインストールする方式を採用していますが、その他の方法が必要となる場合もあります。

- OS インストール時にあらかじめ必要なソフトウェアを選択してインストールしておく
- ISO イメージに含まれている DEB パッケージファイルを apt コマンドなどを使って手動でインストールする
- ISO イメージをリポジトリとして扱うように設定ファイルを作成する
- アクセス可能なローカルネットワーク上にリポジトリサーバーを用意する

ただし、リポジトリサーバーを用意する以外の方法ではセキュリティアップデートなどが行われた最新のパッケージをインストールするのが難しくなります。実際のシステムを運用するには、継続的にパッケージをアップデートできる方法を考えておくべきでしょう。

4.3.7 依存関係の解消

動作するために複数のパッケージが必要となる場合、apt コマンドはインストールするパッケージが必要とするパッケージも同時にインストールを行います。必要となるパッケージがあることを依存関係と呼びます。

4.4 apt install コマンドによるパッケージのインストール

Nginx のパッケージ名は nginx です。まず、sudo コマンドを頭に付けて apt update コマンドを実行し、ローカル環境に保持するレポジトリのパッケージリストを最新化します。その後、sudo apt install コマンドを実行します。

```
ubuntu@host1example1test:~$ sudo apt update
[sudo] password for ubuntu:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://jp.archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://jp.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://jp.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [782
    kB]
Get:6 http://jp.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages
    [1,028 kB]
Get:7 http://jp.archive.ubuntu.com/ubuntu noble-updates/main Translation-en
    [224 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [147
    kB]
Get:9 http://jp.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components
    [161 kB]
Get:10 http://jp.archive.ubuntu.com/ubuntu noble-updates/restricted
    Translation-en [199 kB]
Get:11 http://jp.archive.ubuntu.com/ubuntu noble-updates/restricted amd64
    Components [212 B]
Get:12 http://jp.archive.ubuntu.com/ubuntu noble-updates/universe amd64
    Packages [1,059 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components
    [21.5 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/restricted
    Translation-en [191 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/restricted amd64
    Components [212 B]
Get:16 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages
    [833 kB]
Get:17 http://jp.archive.ubuntu.com/ubuntu noble-updates/universe
    Translation-en [268 kB]
```

```
Get:18 http://jp.archive.ubuntu.com/ubuntu noble-updates/universe amd64
    Components [376 kB]
Get:19 http://security.ubuntu.com/ubuntu noble-security/universe amd64
    Components [52.2 kB]
Get:20 http://jp.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64
    Components [940 B]
Get:21 http://jp.archive.ubuntu.com/ubuntu noble-backports/main amd64
    Components [7,076 B]
Get:22 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64
    Components [212 B]
Get:23 http://jp.archive.ubuntu.com/ubuntu noble-backports/restricted amd64
    Components [216 B]
Get:24 http://jp.archive.ubuntu.com/ubuntu noble-backports/universe amd64
    Components [16.4 kB]
Get:25 http://jp.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64
    Components [212 B]
Fetched 5,746 kB in 11s (529 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
67 packages can be upgraded. Run 'apt list --upgradable' to see them.

ubuntu@host1example1test:~$ sudo apt install nginx
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  nginx-common
Suggested packages:
  fcgiwrap nginx-doc ssl-cert
The following NEW packages will be installed:
  nginx nginx-common
0 upgraded, 2 newly installed, 0 to remove and 67 not upgraded.
Need to get 551 kB of archives.
After this operation, 1,596 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://jp.archive.ubuntu.com/ubuntu noble-updates/main amd64 nginx-common
    all 1.24.0-2ubuntu7.3 [31.2 kB]
Get:2 http://jp.archive.ubuntu.com/ubuntu noble-updates/main amd64 nginx amd64
    1.24.0-2ubuntu7.3 [520 kB]
Fetched 551 kB in 3s (213 kB/s)
Preconfiguring packages ...
Selecting previously unselected package nginx-common.
(Reading database ... 86743 files and directories currently installed.)
Preparing to unpack .../nginx-common_1.24.0-2ubuntu7.3_all.deb ...
Unpacking nginx-common (1.24.0-2ubuntu7.3) ...
Selecting previously unselected package nginx.
Preparing to unpack .../nginx_1.24.0-2ubuntu7.3_amd64.deb ...
Unpacking nginx (1.24.0-2ubuntu7.3) ...
Setting up nginx (1.24.0-2ubuntu7.3) ...
Setting up nginx-common (1.24.0-2ubuntu7.3) ...
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service →
    /usr/lib/systemd/system/nginx.service.
Processing triggers for ufw (0.36.2-6) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.
```

```
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

sudo コマンドを初めて実行する際には、実行しているユーザーのパスワードが要求されます。sudo コマンド実行後、しばらくの間は再度実行する際にはパスワードが要求されませんが、一定時間経過後は再度要求されます。

インストールを行う nginx パッケージの依存関係を確認し、必要となる追加のパッケージも同時にインストールを行います。実行例では、依存関係にあるパッケージも追加でインストールすることを提案しています。問題なければ “y” を入力してパッケージのダウンロードとインストールを行います。

4.5 Web サーバーの起動

Web サーバーを起動します。systemctl コマンドを使って、Web サーバーをバックグラウンドサービスとして起動します。

4.5.1 systemctl コマンド

systemctl コマンドは systemd を制御するためのコマンドです。systemd は Linux が OS として起動する際、Linux カーネルが起動した後に一番最初に実行されるプロセスで、OS 全体を制御します。systemctl コマンドは管理対象をユニットという単位で管理します。

4.5.2 systemctl start コマンドによる Web サーバーの起動

Web サーバーを起動するには、systemctl start コマンドを実行します。systemd は Web サーバーを nginx.service ユニットとして管理しています。ユニット名の.service は省略形の名称が重複していない限り省略できます。なお、nginx はインストール後に自動的に起動されているはずです。

4.6 systemctl status コマンドによる Web サーバーの動作確認

Web サーバーが正しくバックグラウンドサービスとして実行されているかを確認します。システム的に確認する方法と、Web サーバーとして動作していることの確認の両方を実行します。

4.6.1 systemctl status コマンドによる動作状態の確認

systemctl status コマンドで、管理対象となるユニットの状態やログの一部などを確認できます。

表示を終了するには Q キーを入力します。

```
ubuntu@host1example1test:~$ systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset:
         enabled)
     Active: active (running) since Wed 2025-04-30 06:33:44 UTC; 6min ago
       Docs: man:nginx(8)
    Process: 1496 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on;
        master_process on; (code=exited, stat>
    Process: 1501 ExecStart=/usr/sbin/nginx -g daemon on; master_process on;
        (code=exited, status=0/SUCC>
   Main PID: 1504 (nginx)
      Tasks: 2 (limit: 2272)
     Memory: 1.8M (peak: 2.0M)
        CPU: 49ms
     CGroup: /system.slice/nginx.service
             tq1504 "nginx: master process /usr/sbin/nginx -g daemon on;
                 master_process on;"
             mq1505 "nginx: worker process"
```

```
Apr 30 06:33:43 host1example1local systemd[1]: Starting nginx.service - A high
    performance web server an>
Apr 30 06:33:44 host1example1local systemd[1]: Started nginx.service - A high
    performance web server and>
```

なお、nginx が起動していない場合は、sudo systemctl start コマンドを実行してください。

```
ubuntu@host1example1test:~$ systemctl status nginx
○ nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset:
         enabled)
     Active: inactive (dead) since Mon 2025-05-05 00:24:16 UTC; 2s ago
   Duration: 8h 46min 37.149s
       Docs: man:nginx(8)
    Process: 712 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on;
        master_process on; (code=exited, status=>
    Process: 724 ExecStart=/usr/sbin/nginx -g daemon on; master_process on;
        (code=exited, status=0/SUCCESS)
    Process: 1675 ExecStop=/sbin/start-stop-daemon --quiet --stop --retry
        QUIT/5 --pidfile /run/nginx.pid >
   Main PID: 741 (code=exited, status=0/SUCCESS)
        CPU: 286ms

May 04 15:37:35 host1example1local systemd[1]: Starting nginx.service - A high
    performance web server and >
May 04 15:37:39 host1example1local systemd[1]: Started nginx.service - A high
    performance web server and a>
May 05 00:24:16 host1example1local systemd[1]: Stopping nginx.service - A high
    performance web server and >
May 05 00:24:16 host1example1local systemd[1]: nginx.service: Deactivated
    successfully.
May 05 00:24:16 host1example1local systemd[1]: Stopped nginx.service - A high
    performance web server and a>

ubuntu@host1example1test:~$ sudo systemctl start nginx
```

4.6.2 Loaded の意味

ユニットの設定が systemd に読み込まれているかどうかを表しています。ユニットの定義ファイルの位置や、自動起動の設定がされているかが確認できます。

4.6.3 Active の意味

現在、ユニットがアクティブかどうかを表しています。また、プロセスが動作しているかどうかが確認できます。

4.7 Web サーバーへの接続の確認

Web サーバーが起動できたら、様々な方法で Web サーバーに接続して動作を確認します。

4.7.1 curl コマンドによるローカル接続確認

ローカルで Web サーバーが動作していることを確認します。curl コマンドを実行して、自分自身（ローカル）を指す localhost に接続してみます。

```
ubuntu@host1example1test:~$ curl localhost
<!DOCTYPE html>
<html>
<head>
```

```
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Web サーバーが実行されているので、サンプルページの HTML が返ってきます。

4.7.2 ホスト OS の Web ブラウザによる接続確認

ホスト OS の Web ブラウザを実行し、Web サーバーに接続してみます。ホスト OS 上で Web ブラウザを起動したら、アドレスに「192.168.1.12」を入力し、Enter キーを押します。

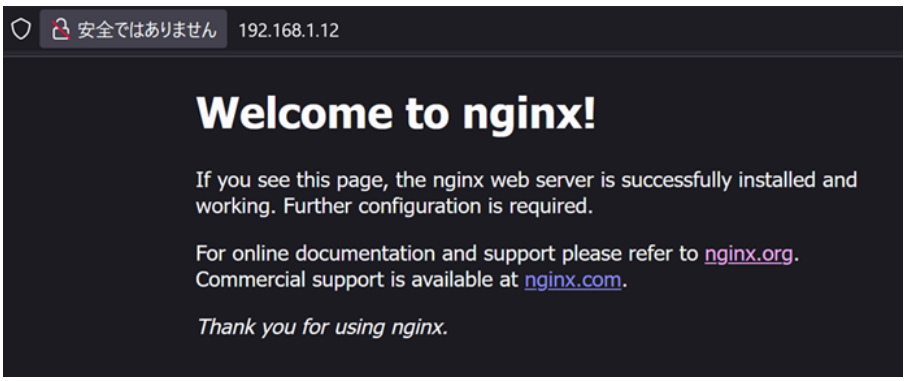

図 36: ホスト OS から VM へ接続した画面

Web ブラウザの要求に対して、Web サーバーが HTML を返し、Web ブラウザがその HTML を解釈して Web ページを表示しているのが分かります。HTML が要求したことにより追加で取得した画像ファイルも埋め込まれています。1 つの Web ページを表示するのに、Web ブラウザの裏側で複数のセッションがやり取りされて画像ファイルなどが取得されるようになっています。

4.8 Web サーバーの停止

Web サーバーを停止してみます。停止には systemctl stop コマンドを実行します。

```
ubuntu@host1example1test:~$ sudo systemctl stop nginx
```

4.8.1 systemctl status コマンドによる動作状態の確認

systemctl status コマンドを実行して、動作状態を確認します。

```
ubuntu@host1example1test:~$ systemctl status nginx
○ nginx.service - A high performance web server and a reverse proxy server
```

```
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset:
         enabled)
     Active: inactive (dead) since Mon 2025-05-05 00:24:16 UTC; 2s ago
   Duration: 8h 46min 37.149s
       Docs: man:nginx(8)
    Process: 712 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on;
        master_process on; (code=exited, status=>
    Process: 724 ExecStart=/usr/sbin/nginx -g daemon on; master_process on;
        (code=exited, status=0/SUCCESS)
    Process: 1675 ExecStop=/sbin/start-stop-daemon --quiet --stop --retry
        QUIT/5 --pidfile /run/nginx.pid >
   Main PID: 741 (code=exited, status=0/SUCCESS)
        CPU: 286ms

May 04 15:37:35 host1example1local systemd[1]: Starting nginx.service - A high
    performance web server and >
May 04 15:37:39 host1example1local systemd[1]: Started nginx.service - A high
    performance web server and a>
May 05 00:24:16 host1example1local systemd[1]: Stopping nginx.service - A high
    performance web server and >
May 05 00:24:16 host1example1local systemd[1]: nginx.service: Deactivated
    successfully.
May 05 00:24:16 host1example1local systemd[1]: Stopped nginx.service - A high
    performance web server and and >
```

Active 行が inactive (dead) になっており、プロセス関係の情報が表示されなくなり、Web サーバーが停止しているのが分かります。

4.8.2 curl コマンドによる接続確認

Web サーバーが停止している状態で、curl コマンドを実行してみます。

```
ubuntu@host1example1test:~$ curl localhost
curl: (7) Failed to connect to localhost port 80 after 0 ms: Couldn't connect
    to server
```

即座にエラーが表示されて、Web サーバーにアクセスできないのが分かります。

4.8.3 Web サーバーを再度起動

Web サーバーを再度起動し、正常にアクセスできるように復旧したことを確認してください。

```
ubuntu@host1example1test:~$ sudo systemctl start nginx
```

```
ubuntu@host1example1test:~$ curl localhost
```

4.9 システム起動時の Web サーバー自動起動

Web サーバーの起動設定を行いましたが、これらの設定が OS を再起動しても自動起動するようになっているか、systemctl is-nenabled コマンドで確認しておきましょう。自動起動が無効である場合、OS の再起動後に Web サーバーが利用出来なくなります。以下のように”enabled”となっていれば自動起動が有効となっています。

```
ubuntu@host1example1test:~$ systemctl is-enabled nginx
enabled
```

自動起動設定が無効 (disabled) となっている場合は、systemctl enable コマンドで有効化します。

```
ubuntu@host1example1test:~$ systemctl is-enabled nginx
disabled

ubuntu@host1example1test:~$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with
    /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service →
    /usr/lib/systemd/system/nginx.service.
```

4.9.1 OS 再起動

まず、OS を再起動し、Web サーバーが自動起動せず、ファイアウォールの設定が行われないことを確認します。OS を再起動するには sudo systemctl reboot コマンドを実行します。

```
ubuntu@host1example1test:~$ sudo systemctl reboot
```

4.9.2 再起動後の動作確認

Web サーバーが自動的に起動していないこと、ファイアウォールの設定で HTTP が許可されていないことを確認します。

```
ubuntu@host1example1test:~$ sudo systemctl status nginx
```

4.10 ログの確認

Web サーバーへのアクセスや発生したエラーはログファイルに記録されています。アクセスログはどのような人がどのようなページにアクセスしているかの分析に、エラーログは発生した障害の解決に利用されます。

それぞれ、どのような情報が記録されているか確認してみましょう。

4.10.1 Web サーバーのログファイルの確認

Web サーバーのログは/var/log/nginx/ディレクトリに記録されています。

```
ubuntu@host1example1test:~$ ls /var/log/nginx/
access.log    error.log
```

access_log と error_log の 2 つのログファイルなどが確認できます。

4.10.2 アクセスログの確認

アクセスログに記録されている内容を確認します。

```
ubuntu@host1example1test:~$ cat /var/log/nginx/access.log
::1 - - [05/May/2025:00:25:54 +0000] "GET / HTTP/1.1" 200 288 "-" "curl/8.5.0"
192.168.1.122 - - [05/May/2025:05:06:33 +0000] "GET / HTTP/1.1" 200 226 "-"
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:138.0) Gecko/20100101
    Firefox/138.0"
192.168.1.122 - - [05/May/2025:05:06:33 +0000] "GET /favicon.ico HTTP/1.1" 404
    134 "http://192.168.1.12/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64;
    rv:138.0) Gecko/20100101 Firefox/138.0"
```

アクセスログには、以下のような情報が記録されています。

- アクセス元の IP アドレス
- アクセスした時間

- アクセスの内容
- アクセスの結果（エラー番号）
- アクセスした際に入力された URL
- アクセスに利用されたブラウザの種類

4.10.3 エラー番号

アクセスログの中で分かりにくいのが、アクセスの結果を示すエラー番号です。エラー番号は様々な種類が定義されていますが、現時点で記録されているのは主に以下の 2 つです。

- 200：アクセスが成功した
- 404：アクセスしたファイルなどが見つからず失敗した

この 2 つ以外にも、304 や 403 などが記録されている場合があります。

4.10.4 エラーログの確認

エラーログに記録されている内容を確認します。

```
ubuntu@host1example1test:~$ cat /var/log/nginx/error.log
```

エラーログには、Web サーバーの起動や停止の際に出力されたログや、アクセスログにも記録されているエラーの詳細などが記録されます。

4.10.5 index.html を配置する

Web サーバーは、Web ブラウザからのリクエストに応じて表示する HTML ファイルなどを/var/www/html ディレクトリ以下に配置するように設定されています。index.html は、Web ブラウザがファイル名を指定しなかった場合にデフォルトで参照される HTML ファイルの名前です。では、デフォルトで用意されたものではなく、独自内容の index.html を作成してみましょう。

以下のコマンドを実行して、/var/www/html ディレクトリに index.html ファイルを作成します。

```
ubuntu@host1example1test:~$ sudo sh -c "echo 'Welcome to My First Web Server' >
    /var/www/html/index.html"
```

書き込みには管理者権限が必要ですが、リダイレクトによってファイルを作成するにはコマンド全体を sudo コマンドで呼び出した sh -c コマンドで実行する必要があります。また、コマンド全体を「“」(ダブルクォーテーション) で囲い、書き込み文字列を「’」(シングルクォーテーション) で囲っています。この違いにも注意してコマンドを入力してください。

4.10.6 Web ブラウザからのアクセスとログの確認

Web ブラウザから再度 Web サーバーにアクセスしてみます。今度はテストページではなく、変更したメッセージが表示されたのではないでしょうか。

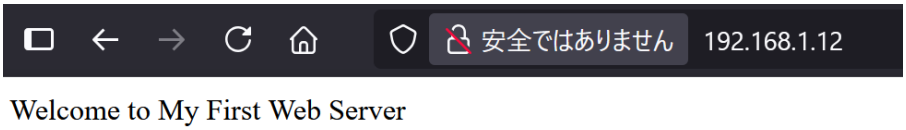

図 37: メッセージ表示が変わった画面

また、アクセスログにどのように記録されているか、エラーログにエラーが記録されていないことも確認してみてください。

5 DNS サーバーの構築

第 5 章では、ネットワークサービスを使うための土台となる名前解決のサービス (DNS) を設定します。自分の DNS サーバーを他のコンピュータから参照できるように設定をします。DNS に問い合わせを行うコマンドに慣れ、ドメインを管理する BIND プログラムの設定ファイルを扱います。

5.1 用語集

DNS

DNS(Domain Name System) は、IP アドレスと対応するホスト名を登録しておき、プログラムからの問い合わせに応じて IP アドレスやホスト名を返答するシステムです。

ドメイン名とゾーン

組織に割り当てられてインターネットで使用する名前をドメイン名と呼びます。ドメイン名は ICANN(Internet Corporation for Assigned Names and Numbers) により管理されています。DNS でドメイン名を設定するときは、ドメインではなく「ゾーン」と呼びます。

ゾーン

DNS の名前空間の一部分を取り出したものをゾーンとよびます。ゾーンは、DNS を管理する単位として使われます。ゾーンには、ドメイン、サブドメイン、ホスト名などが含まれています。DNS 名前空間はツリー構造で表されますが、ゾーンは特定のノード以下の一部または全部を含む部分です。

FQDN

ドメイン名表記で、一番右に「.」（ドット）でルートドメインまでを記述する方式を FQDN と呼びます。

DNS キャッシュサーバー

プログラムからの名前問い合わせを代行して、様々な DNS サーバーへ名前問い合わせを行って結果を返却するサーバーです。調べた結果をキャッシュしておき、次回問い合わせ時にキャッシュの情報を返すことから、DNS キャッシュサーバーと呼ばれます。クライアントのネットワーク設定で DNS サーバーと呼んだときには、DNS キャッシュサーバーのことを指します。

DNS コンテンツサーバー

委託されたゾーンの IP アドレスやホスト名を管理する DNS サーバーです。取得したドメイン名を利用するには、DNS コンテンツサーバーを用意して設定する必要があります。

リゾルバ

ドメイン名をもとに IP アドレス情報の検索をしたり、IP アドレスからドメイン情報の検索を行う、名前解決を行うプログラムのことです。

BIND

BIND(Berkeley Internet Name Domain) は、Linux と組み合わせて多く使用されている DNS サーバーのソフトウェアです。DNS キャッシュサーバーとしても、DNS コンテンツサーバーとしても利用することができます。

グルーレコード

管理を委任しているゾーンについての問合せに対して、DNS サーバーが委任先のゾーンの DNS コンテンツサーバーのアドレスを返す際に、追加情報として必要となる委任先 DNS コンテンツサーバーの A レコードをグルーレコードといいます。

A レコード

名前に対して IP アドレスを指定するためのレコードです。

NS(Name Server) レコード

ゾーンの権威を持つ DNS コンテンツサーバーを指定するためのレコードです。

MX(Mail eXchange) レコード

メールアドレスに利用するドメイン名を定義するためのレコードです。メールサーバーの障害にも対応するために、複数個のメールサーバーを記述でき、プリファレンス値の低いサーバーにメール配信が優先されます。

5.2 DNS の仕組み

インターネットでのコンピュータ同士の通信は、IP(Internet Protocol) を使って行われています。IP 通信には相手の IP アドレスが必要ですが、インターネット上の大量のコンピュータを IP アドレスで識別するのは困難です。そこでドメイン名やホスト名という考え方が導入されました。

ドメイン名は組織を表し、ホスト名はその組織が管理しているコンピュータです。表記するときは「ホスト名.ドメイン名」とドット区切りで表記しますが、両方を合わせてホスト名と呼ぶこともあります。

5.2.1 HOSTS ファイルと DNS

インターネットの研究が始まった当初は IP アドレスが割り当てられたコンピュータの数も数えるほどだったので、ホスト名と IP アドレスの対応関係はファイルに記述されて、定期的に更新されていました。この仕組みは今でも残っており、Linux では/etc/hosts がそのファイルです。

しかし、インターネットが広まるに従って、ホストファイルでは管理しきれなくなってきました。そこで登場したのが DNS(Domain Name System) です。

5.2.2 DNS コンテンツサーバーによるドメイン名の管理

ドメイン名を割り当てられた組織毎に DNS コンテンツサーバーを用意します。DNS コンテンツサーバーの管理者は、そのドメインに所属しているホスト名と割り当てられた IP アドレスを DNS コンテンツサーバー登録します。

あるドメインに所属しているホストにアクセスしたいユーザーは、そのドメインの DNS コンテンツサーバーに問い合わせを行うことで、IP アドレスを得ることができます。

DNS の仕組みでは、ドメイン（ゾーン）の管理権限がそれぞれの DNS 管理者に委譲されているので、ホストファイルのような一元管理ではなく分散管理となります。管理作業が分担されていて更新も頻繁に行われるので、リアルタイムにホスト名と IP アドレスの対応関係を調べることができる仕組みとなっています。

5.2.3 DNS キャッシュサーバーによる名前解決

ユーザーの端末が名前解決を必要とする度に、アクセス先の DNS コンテンツサーバーを探して問い合わせをするのは非常に煩雑になります。そこで端末は DNS キャッシュサーバーに名前解決を依頼し、DNS キャッシュサーバーが後述する再帰問い合わせという方法で DNS コンテンツサーバーに対して問い合わせを行います。名前解決が完了すると、その結果だけを端末に返します。

このような仕組みになっているので、PC やスマートフォンでは IP アドレス設定の際に検索に使用する DNS サーバーの設定が必要になります。この設定が、名前解決を依頼する DNS キャッシュサーバーの IP アドレスとなります。

また、名前解決の結果は一定期間キャッシュされています。同じホストの名前解決が依頼されるとキャッシュから返答するようにしているので、毎回調べる必要がありません。

5.3　ドメインの構造

DNS が取り扱うドメイン名は設計上、ルートドメインを頂点とした階層型のツリー構造となっています。ちょうどコンピュータのファイルシステムが、ルートディレクトリを頂点としたツリー構造になっているのと同じだと考えてよいでしょう。そして、その配下にあるドメインはサブドメインとよばれます。ドメインの階層型のツリーは、ルートドメインとたくさんのサブドメインから構成されています。

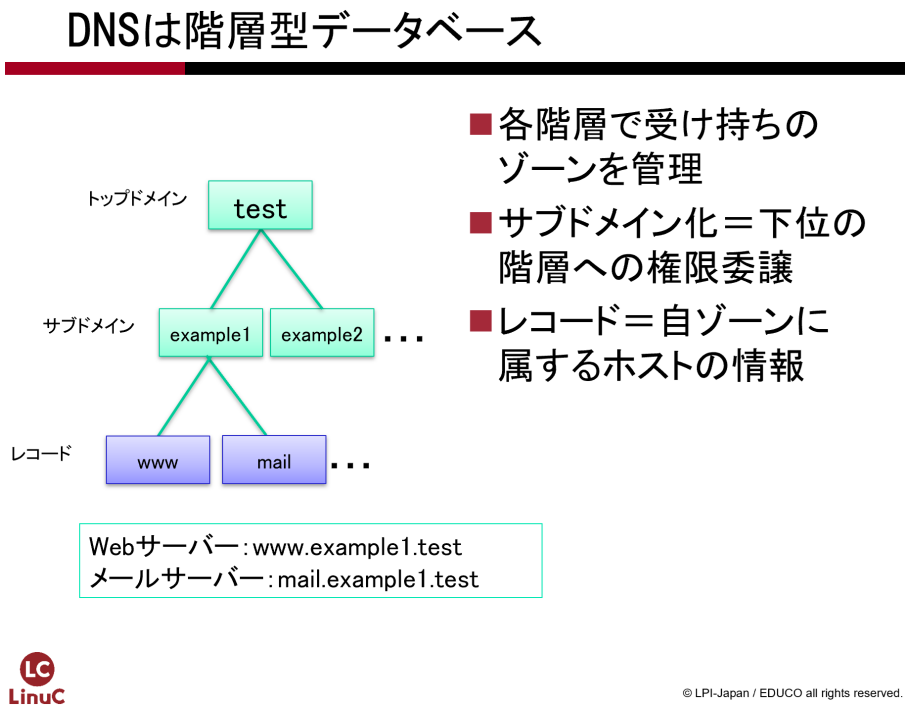

図 38: DNS は階層型データベース

5.3.1　ルートドメイン

ルートドメインは、ドメイン名の開始点です。通常は省略されますが、ドメイン名として記述する際には「.」（ドット）で表されます。

5.3.2　トップレベルドメイン

トップレベルドメインには、.com や.org のような組織別ドメインや、.jp のような国別ドメインがあります。また、日本の場合には example.co.jp のような組織種別型ドメインと、example.jp のような汎用 JP ドメインなどがあります。

5.3.3　ドメイン名の記述

ドメイン名の記述は、右側から記述していきます。FQDN(Fully Qualified Domain Name) であれば一番右にルートドメイン、そしてトップレベルドメインを記述し、さらに左側に各組織毎に割り当てられたドメイン名を記述していきます。各要素の間は「.」（ドット）で区切っていきます。

5.3.4　ドメイン名の記述例

- example.com.
- example.jp.
- example.co.jp.

トップレベルドメイン以降のドメイン名は、ドメイン取得者が独自にドメイン名を決めることができます。上記の例では example の部分が独自のドメイン名にあたります。

5.3.5 サブドメイン

記述例のようにドメイン名の左側にさらにドメイン名を記述していくことを「サブドメイン化」と呼びます。たとえば、example.co.jp ドメインをさらに東京と大阪の 2 つに分けて表記したいような場合には、以下の例のように記述します。

- tokyo.example.co.jp.
- osaka.example.co.jp.

サブドメイン化は、上位のドメイン（表記上の右側）を管理している管理者が行います。たとえば、tokyo.example.co.jp ドメインまでのサブドメインの階層は次のようになっています。

1. jp ドメインはルートドメインのサブドメイン
2. co.jp ドメインは jp ドメインのサブドメイン
3. example.co.jp ドメインは co.jp ドメインのサブドメイン
4. tokyo.example.co.jp ドメインは example.co.jp ドメインのサブドメイン

5.3.6 ドメイン名の取得

ドメイン名を取得するということは、上位のドメイン名の管理者にサブドメインを作ってもらい、管理権限を委譲してもらうということになります。

独自の短いドメイン名を取得したいのであればトップレベルドメインを管理している管理組織からサブドメイン化してもらうことになりますが、登録料などの費用がかかります。

短いドメイン名にこだわらない、費用をかけたくない場合には、既にドメイン名を取得している管理者からサブドメインの管理権限を委譲してもらうこともできます。

5.4　DNS を使った名前解決と再帰問い合わせ

DNS を使って名前を解決する、すなわち名前から IP アドレスを調べる時には、次のような手順で調査が行われます。

1. 端末は、自分の組織やプロバイダーの用意した DNS キャッシュサーバーへ問い合わせます。
2. DNS キャッシュサーバーは、ルートサーバーに「test」を管理する DNS コンテンツサーバーの IP アドレスを問い合わせます。ルートサーバーは、「test」を管理する DNS コンテンツサーバーの IP アドレスを DNS キャッシュサーバーへ返します。
3. DNS キャッシュサーバーは、「test」を管理する DNS コンテンツサーバーへ、サブドメイン「example1.test」を管理する DNS コンテンツサーバーの IP アドレスを問い合わせます。「test」を管理する DNS コンテンツサーバーは、サブドメイン「example1.test」を管理する DNS コンテンツサーバーの IP アドレスを DNS キャッシュサーバーへ返します。
4. DNS キャッシュサーバーは、「example1.test」を管理する DNS コンテンツサーバーへ、「www.example1.test」の IP アドレスを問い合わせます。「example1.test」を管理する DNS コンテンツサーバーは、「www.example1.test」の IP アドレスを DNS キャッシュサーバーへ返します。
5. DNS キャッシュサーバーは、クライアントに「www.example1.test」の IP アドレスを返します。

このように、DNS キャッシュサーバーがルートサーバーから順番に問い合わせを行い、最終的に目的のドメインを管理する DNS コンテンツサーバーまで問い合わせをしていくことを「再帰問い合わせ」と呼びます。

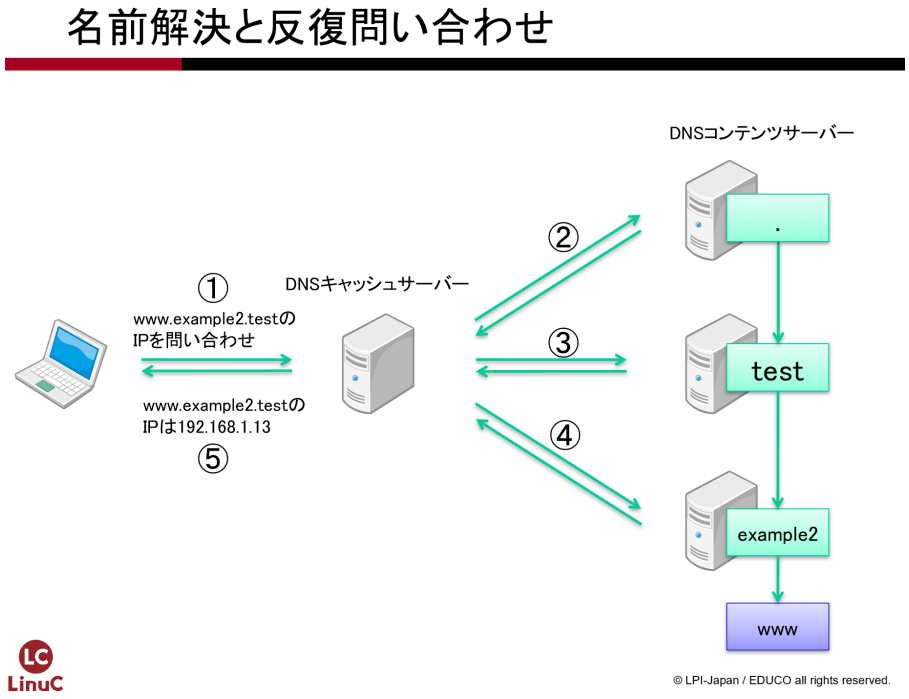

図 39: 名前解決と再帰問い合わせ

5.5 演習で構築する DNS の概略

2 つのドメイン名「example1.test」と「example2.test」を設定し、相互に名前解決ができるように DNS コンテンツサーバーを構築します。また、test ドメインからサブドメイン化してゾーンの管理権限を委譲する設定も行います。以下の 3 つの DNS サーバーを構築します。

5.5.1 test ゾーンのマシン

example1.test と example2.test をサブドメイン化し、ゾーン管理権限の委譲を設定します。また DNS キャッシュサーバーとしても動作します。

5.5.2 example1.test ゾーンのマシン

example1.test ゾーンを管理する DNS コンテンツサーバーとして設定します。

5.5.3 example2.test ゾーンのマシン

example2.test ゾーンを管理する DNS コンテンツサーバーとして設定します。

5.6 演習の手順

3 台のマシンを相互に接続し、相互に DNS を参照できるように設定します。追加でマシン 2 台分の仮想マシンの作成や OS のインストールなどが必要となります。以下の作業を行っていきます。

1. DNS サーバーソフトウェアである BIND を使って、example1.test ゾーンを管理する DNS コンテンツサーバーを設定します。
2. test ゾーンおよび example2.test ゾーンの仮想マシンを作成し、OS をインストールします。それぞれ IP アドレスなどの設定が異なる点に注意してください。
3. example2.test ゾーンを管理する DNS コンテンツサーバーを設定します。
4. test ゾーンを管理する DNS コンテンツサーバーを設定します。example1.test ゾーンおよび example2.test ゾーンに対する権限委譲を設定します。
5. example1.test ゾーンと example2.test ゾーンの DNS を相互に参照できることを確認します。

次章のメールサーバー構築では DNS サーバーが正しく設定されていることを前提としているので、この章の演習内容が完全に終わっている必要があります。

5.7 演習で使うアドレスとドメイン

演習で使用する 3 台のサーバーのドメイン名、ホスト名、IP アドレスは以下の通りです。

ドメイン名	ホスト名	IP アドレス
test	host0.test	192.168.1.11
example1.test	host1.example1.test	192.168.1.12
example2.test	host2.example2.test	192.168.1.13

5.8 アドレス解決の流れ

host1.example1.test のマシン (192.168.1.12) がホスト www.example2.test を解決するときの動きを追ってみましょう。

Web ブラウザで Web ページを表示させるとき、DNS キャッシュサーバーのことは特に意識せず Web サイトのアドレスを入力しています。ここでは Web アドレスを入力してリクエストしてページが表示されるまでの流れを例に、DNS がどのように動くのか簡単に説明します。

1. host1.example1.test マシンは、Web ブラウザにアドレスとして www.example2.test を入力します。
2. Web ブラウザは、Linux のリゾルバに問い合わせます。
3. リゾルバは、/etc/resolv.conf ファイルで指定されている DNS キャッシュサーバー（192.168.1.11）へ問い合わせます。
4. DNS キャッシュサーバーは、test ゾーンの DNS コンテンツサーバーを参照し、example2.test ゾーンの DNS コンテンツサーバーの IP アドレス（192.168.1.13）を返します。
5. DNS キャッシュサーバーは、example2.test ゾーンの DNS コンテンツサーバーに問い合わせします。

6. example2.test ゾーンの DNS コンテンツサーバーは、www.example2.test ホストの IP アドレス（192.168.1.13）を返します。
7. DNS キャッシュサーバーは、結果を example1.test マシンへ返します。
8. Web ブラウザは、www.example2.test に HTTP でアクセスし、Web ページを受け取って表示します。

実際のインターネットでの DNS の名前解決ではルートゾーンから順番に再帰問い合わせを行いますが、演習環境では DNS キャッシュサーバー自身が jp ゾーンの DNS コンテンツサーバーでもあるため、すぐに example2.test の DNS コンテンツサーバーに関する情報を返す点が異なります。

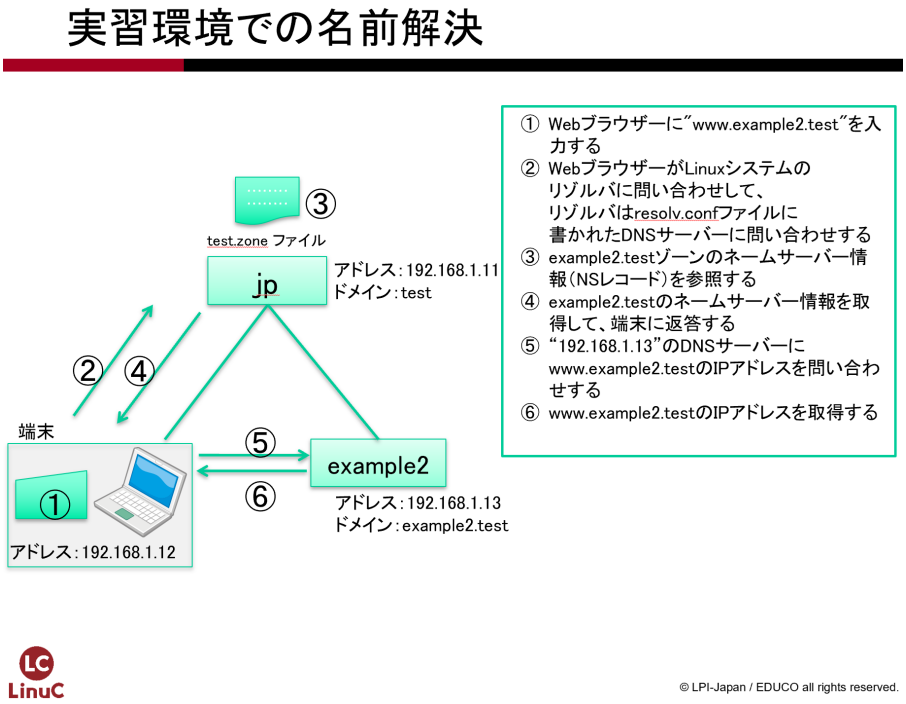

図 40: 実習環境での名前解決

5.9 DNS コンテンツサーバーの設定

DNS コンテンツサーバーのソフトウェアとして BIND をインストールして、各ゾーンの設定を行います。

5.9.1 ゾーンを設定する流れ

ゾーンを設定するために必要な作業は以下の手順となります。

1. BIND のインストールを行います。
2. named.conf ファイルにゾーンを追加します。
3. ゾーンファイルを作成し、レコードなどを記述します。
4. 設定ファイルに書式間違いが無いか確認します。
5. BIND の起動を行います。
6. BIND の自動起動の設定を行います。
7. ファイアウォールの設定を変更します。
8. /etc/resolv.conf の参照する DNS サーバーの設定を確認します。
9. 名前解決の確認を行います。

BIND の基本的な設定ファイルとして/etc/bind/named.conf ファイルがあります。/etc/bind/named.conf に BIND の基本的な設定とゾーンの定義を追加します。さらにゾーンの詳細を定義するゾーンファイルを/etc/bind ディレクトリに作ります。

5.10 BIND のインストール

BIND のインストールには、bind9 パッケージを使います。必要なパッケージがインストールされていないときには、apt コマンドでインストールします。

```
ubuntu@host1example1test:~$ sudo apt install bind9
[sudo] password for ubuntu:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bind9-utils dns-root-data
Suggested packages:
  bind-doc
The following NEW packages will be installed:
  bind9 bind9-utils dns-root-data
0 upgraded, 3 newly installed, 0 to remove and 73 not upgraded.
Need to get 419 kB of archives.
After this operation, 1,624 kB of additional disk space will be used.
Do you want to continue? [Y/n]
Get:1 http://jp.archive.ubuntu.com/ubuntu noble-updates/main amd64 bind9-utils
    amd64 1:9.18.30-0ubuntu0.24.04.2 [159 kB]
Get:2 http://jp.archive.ubuntu.com/ubuntu noble-updates/main amd64
    dns-root-data all 2024071801~ubuntu0.24.04.1 [5,918 B]
Get:3 http://jp.archive.ubuntu.com/ubuntu noble-updates/main amd64 bind9 amd64
    1:9.18.30-0ubuntu0.24.04.2 [254 kB]
Fetched 419 kB in 2s (180 kB/s)
Selecting previously unselected package bind9-utils.
(Reading database ... 86791 files and directories currently installed.)
Preparing to unpack .../bind9-utils_1%3a9.18.30-0ubuntu0.24.04.2_amd64.deb ...
Unpacking bind9-utils (1:9.18.30-0ubuntu0.24.04.2) ...
Selecting previously unselected package dns-root-data.
Preparing to unpack .../dns-root-data_2024071801~ubuntu0.24.04.1_all.deb ...
Unpacking dns-root-data (2024071801~ubuntu0.24.04.1) ...
Selecting previously unselected package bind9.
Preparing to unpack .../bind9_1%3a9.18.30-0ubuntu0.24.04.2_amd64.deb ...
Unpacking bind9 (1:9.18.30-0ubuntu0.24.04.2) ...
Setting up dns-root-data (2024071801~ubuntu0.24.04.1) ...
Setting up bind9-utils (1:9.18.30-0ubuntu0.24.04.2) ...
Setting up bind9 (1:9.18.30-0ubuntu0.24.04.2) ...
```

```
info: Selecting GID from range 100 to 999 ...
info: Adding group `bind' (GID 110) ...
info: Selecting UID from range 100 to 999 ...

info: Adding system user `bind' (UID 110) ...
info: Adding new user `bind' (UID 110) with group `bind' ...
info: Not creating home directory `/var/cache/bind'.
wrote key file "/etc/bind/rndc.key"
named-resolvconf.service is a disabled or a static unit, not starting it.
Created symlink /etc/systemd/system/bind9.service →
    /usr/lib/systemd/system/named.service.
Created symlink /etc/systemd/system/multi-user.target.wants/named.service →
    /usr/lib/systemd/system/named.service.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for ufw (0.36.2-6) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

5.11 /etc/named.conf の基本設定

/etc/bind/named.conf ファイルに BIND を DNS コンテンツサーバーとして動作させる基本設定を行います。

```
ubuntu@host1example1test:~$ sudo vi /etc/bind/named.conf

ubuntu@host1example1test:~$ sudo cat /etc/bind/named.conf
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
include "/etc/bind/named.conf.my-zones";  ← 追記設定を記載するファイルを指定
```

```
ubuntu@host1example1test:~$ sudo vi /etc/bind/named.conf.my-zones

ubuntu@host1example1test:~$ sudo cat /etc/bind/named.conf.my-zones
zone "example1.test" IN { ← example1.testゾーンを指定
        type master;
        file "/etc/bind/example1.test.zone"; ←
            example1.test用のゾーン定義ファイル名を指定
        allow-update { none; };
};
```

```
ubuntu@host1example1test:~$ sudo vi /etc/bind/example1.test.zone
```

```
ubuntu@host1example1test:~$ sudo cat /etc/bind/example1.test.zone
$TTL 3H
$ORIGIN example1.test.
@       IN SOA  host1 root (
                                        2025050501       ; serial
                                        1D      ; refresh
                                        1H      ; retry
                                        1W      ; expire
                                        3H )    ; minimum
        NS      host1.example1.test.
        MX 10   mail.example1.test.

host1   A       192.168.1.12
www     A       192.168.1.12
mail    A       192.168.1.12
```

```
ubuntu@host1example1test:~$ sudo vi /etc/bind/named.conf.options

ubuntu@host1example1test:~$ sudo cat /etc/bind/named.conf.options
options {
        directory "/var/cache/bind";
        //directory "/etc/bind";
        listen-on port 53 { 127.0.0.1; 192.168.1.12; };
        listen-on-v6 port 53 { ::1; };
        allow-query     { any; };

        recursion no;

        dnssec-validation auto;
};
```

設定の内容は以下の通りです。

5.11.1 問い合わせを受け付けるアドレスの設定

デフォルトの named.conf ファイルは、127.0.0.1(ローカルループバックインターフェース) への問い合わせにしか返答しない設定なので、外部からの問い合わせを受けられるようにサーバー自身の IP アドレスである「192.168.1.12;」を listen-on に追加します。後ほど設定する example2.test サーバーや local サーバーの場合にはそれぞれのサーバーの IP アドレスを記述します。

5.11.2 問い合わせを許可するアドレスの設定

allow-query にはデフォルトでは「localhost;」と設定されていて、ローカルからしか DNS 問い合わせができないようになっています。DNS コンテンツサーバーはインターネット上のすべての人から参照できなければならないので、この設定を「any」に変更します。

5.11.3 DNS コンテンツサーバーとしての設定

DNS コンテンツサーバーでは、recursion(再帰問合せ) を禁止にしておく必要があります。そのため、recursion に「no」を設定します。ただし、local サーバーの場合には DNS キャッシュサーバーとしても動作させるので yes のままにしておく必要があります。

5.12 ゾーンファイルの準備

5.13 ゾーンファイルの修正

5.13.1 $TTL

このゾーン定義ファイル内の記述の TTL（Time to Live・生存時間・有効期間）が 3 時間であることをデフォルト指定しています。

5.13.2 $ORIGIN

このゾーン定義が対象としているゾーン名を指定します。ゾーン定義内のホスト名はすべて FQDN で最後が「.」で終わる必要がありますが、省略された場合には$ORIGIN で指定されたゾーン名で補完されます。たとえば、「www」という記述は「www.example1.test.」と補完されて扱われます。

5.13.3 SOA レコード

@から始まるゾーンファイルの最初のレコードは SOA レコードです。このゾーンの管理ポリシーについて設定します。SOA レコードの先頭には@がありますが、これは$ORIGIN で指定したゾーン（ここでは example1.test.）に置き換えられます。host1 はこの DNS コンテンツサーバーのホスト名、root は管理者ユーザーです。どちらもゾーン名が補完されて、「host1.example1.test.」「root.example1.test.」となりますが、ユーザー名は最初の「.」を「@」にしてメールアドレスとして読替えます。これらは単なる文字列なので、BIND の動作には影響を与えません。シリアルナンバー (serial) は、ゾーン定義を変更する毎に必ず変更する必要があります。西暦 (4 桁の年) と月日 (2 桁ずつ) の後に 01 から 99 までの数字 (2 桁) が付いた 10 桁の数字で指定します。日が異なる場合には日付を、同じ日に変更が複数回あった場合には最後の 2 桁を変更するのを忘れないようにしてください。

5.13.4 NS レコード

NS レコードは、このゾーンの DNS コンテンツサーバーである自分自身を定義します。

5.13.5 MX レコード

MX レコードは受講生ドメインのメールサーバーを定義します。

NS レコードや MX レコードの定義では、右側に FQDN を入れるので、最後に必ず「.」を付けてください。また、先頭が空白になっていますが、これは前の行と同じ対象（この場合には@）が省略されていることを示しています。

5.13.6 A レコード

A レコードで名前と IP アドレスの対応を定義する箇所は、左側にホスト名、右側に IP アドレスが入ります。マシンのホスト名である host1 や、その他のサービスで使う名前である www や mai と IP アドレスへの対応を記述しました。最後に「.」が付かない名前には、$ORIGIN で定義しているゾーン名 (ここでは example1.test.) が自動的に追加されます。

5.14 設定ファイルとゾーンファイルの書式確認

設定ファイルとゾーンファイルの作成が終わったら、書式の確認を行います。設定が多岐に渡るため、間違いがないかを確認するためのコマンドを使って確認します。

5.14.1 設定ファイルの書式確認と注意点

/etc/bind/named.conf ファイルの編集時、括弧やセミコロンの不足などは良くある設定ミスです。named-checkconf コマンドで/etc/bind/named.conf に間違いがないか確認しましょう。

```
ubuntu@host1example1test:~$ named-checkconf /etc/bind/named.conf
```

この例のように、何も表示されなければ書式に問題がないということです。問題がある場合には、次のように問題がありそうな行番号が表示されます。

```
ubuntu@host1example1test:~$ named-checkconf /etc/bind/named.conf
/etc/named.conf:11: missing ';' before '}'
```

これは、listen-on port の{}にアドレスを追加するときに IP アドレスの後にセミコロンを記述するのを忘れたというエラーです。エラー表示を見ながら、こうした問題を取り除きましょう。

5.14.2 ゾーンファイルの書式確認

ゾーンファイルを編集時、よくあるミスとしては、FQDN で記述すべきところを最後の. が抜けているなどがあります。named-checkzone コマンドを使ってゾーンファイルに間違いがないか確認しましょう。引数は$ORIGIN に指定したゾーン名と、確認を行うゾーンファイル名です。

```
ubuntu@host1example1test:~$ named-checkzone example1.test
    /etc/bind/example1.test.zone
zone example1.ltest/IN: loaded serial 2025050501
OK
```

書式に問題がなければ、この例のように設定したシリアルナンバーが表示され、OK と表示されます。設定が間違っている場合には、次のように問題のある行番号が表示されます。

```
ubuntu@host1example1test:~$ named-checkzone example1.test
    /etc/bind/example1.test.zone
zone example1.test/IN: NS 'host1.example1.test.example1.test' has no address
    records (A or AAAA)
zone example1.test/IN: not loaded due to errors.
```

この例では、NS レコードの右側に書いたホスト名が FQDN になっていないため、「host1.example1.test.example1.test」となってしまい、対応する A レコードが見つからないというエラーが発生しています。

5.15 BIND の再起動と確認

BIND の設定を更新するため、再起動してみましょう。BIND の再起動は、systemctl コマンドで bind9 ユニットを使います。

```
ubuntu@host1example1test:~$ sudo systemctl restart bind9
```

起動できたら、状態を確認します。

```
ubuntu@host1example1test:~$ systemctl status bind9
● named.service - BIND Domain Name Server
     Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; preset:
         enabled)
     Active: active (running) since Mon 2025-05-05 06:09:08 UTC; 3s ago
       Docs: man:named(8)
   Main PID: 3591 (named)
     Status: "running"
      Tasks: 4 (limit: 2272)
     Memory: 4.5M (peak: 4.7M)
        CPU: 107ms
     CGroup: /system.slice/named.service
             mq3591 /usr/sbin/named -f -u bind

May 05 06:09:08 host1example1local named[3591]: command channel listening on
    ::1#953
May 05 06:09:08 host1example1local named[3591]: managed-keys-zone: loaded
    serial 2
May 05 06:09:08 host1example1local named[3591]: zone 0.in-addr.arpa/IN: loaded
    serial 1
May 05 06:09:08 host1example1local named[3591]: zone 127.in-addr.arpa/IN:
    loaded serial 1
May 05 06:09:08 host1example1local named[3591]: zone localhost/IN: loaded
    serial 2
```

```
May 05 06:09:08 host1example1local named[3591]: zone 255.in-addr.arpa/IN:
    loaded serial 1
May 05 06:09:08 host1example1local named[3591]: zone example1.local/IN: loaded
    serial 2025050501
May 05 06:09:08 host1example1local named[3591]: all zones loaded
May 05 06:09:08 host1example1local systemd[1]: Started named.service - BIND
    Domain Name Server.
May 05 06:09:08 host1example1local named[3591]: running
```

Active の欄に「active (running)」と表示されていることを確認します。

必要に応じて、表示を終了するために Q キーを押します。

5.16 自動起動の設定

システム起動時に BIND が自動的に起動されるようになっているか、systemctl is-enabled コマンドで確認します。

```
ubuntu@host1example1test:~$ sudo systemctl is-enabled bind9
alias
```

自動起動の設定がされている場合には、「enabled」または「alias」と表示されます。

5.17 参照する DNS サーバーの設定

DNS サーバーによる名前解決を確認する前に、参照する DNS サーバーの設定について確認、変更します。

5.17.1 /etc/resolv.conf による参照 DNS サーバーの設定

Linux で名前解決を行うために参照する DNS サーバーは/etc/resolv.conf に記述されます。

```
ubuntu@host1example1test:~$ sudo vi /etc/resolv.conf

ubuntu@host1example1test:~$ cat /etc/resolv.conf
nameserver 192.168.1.12
search example1.test
```

設定の内容は使用しているネットワーク環境によって異なりますが、参照する DNS サーバーを指定する nameserver の設定値として、インストール時に設定した「8.8.8.8」以外に設定されているのが NAT ネットワークの DHCP で設定された参照する DNS サーバーです。

5.17.2 エディタで/etc/resolv.conf を修正すると？

エディタで/etc/resolv.conf を修正すると、設定は即時有効となります。

ただし、システムの再起動などによって修正変更は破棄されます。一時的に動作を変更したいような場合にはよいですが、継続して設定を適用したい場合には大元となるネットワークの設定を変更する必要があります。本章の最後では、ネットワークインターフェースの設定を変更する方法を解説しています。

5.17.3 ネットワークインターフェースの名前を確認

まず、ネットワークインターフェースの名前を確認します。

```
ubuntu@host1example1test:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
    default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
```

```
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
    group default qlen 1000
    link/ether 08:00:27:ee:b7:1d brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.12/24 brd 192.168.1.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 240f:32:57b8:1:a00:27ff:feee:b71d/64 scope global dynamic mngtmpaddr
        noprefixroute
       valid_lft 291sec preferred_lft 291sec
    inet6 fe80::a00:27ff:feee:b71d/64 scope link
       valid_lft forever preferred_lft forever
```

この例では、IP アドレス「192.168.1.12」からネットワークインターフェースは「enp0s3」と分かります。

5.17.4 起動時に読み込まれる resolv.conf の設定の更新

OS 起動時に/etc/resolv.conf は、プログラム systemd-resolved により/run/systemd/resolve/stub-resolv.conf のリンクとなっています。その為、/run/systemd/resolve/resolv.conf を編集しリンクを貼り直します。

```
ubuntu@host1example1test:~$ sudo ls -l /etc/resolv.conf
lrwxrwxrwx 1 root root 39 Feb 16 20:58 /etc/resolv.conf ->
    ../run/systemd/resolve/stub-resolv.conf

ubuntu@host1example1test:~$ sudo ln -fs /run/systemd/resolve/stub-resolv.conf
    /etc/resolv.conf
```

```
ubuntu@host1example1test:~$ sudo vi /etc/systemd/resolved.conf
[Resolve]
DNSStubListener=no

ubuntu@host1example1test:~$ sudo systemctl restart systemd-resolved
```

```
ubuntu@host1example1test:~$ sudo cat /etc/netplan/50-cloud-init.yaml
[sudo] password for ubuntu:
network:
  version: 2
  ethernets:
    enp0s3:
      addresses:
      - "192.168.1.12/24"
      nameservers:
        addresses:
        - 8.8.8.8
        search: []
      routes:
      - to: "default"
        via: "192.168.1.1"

ubuntu@host1example1test:~$ sudo vi /etc/netplan/50-cloud-init.yaml

ubuntu@host1example1test:~$ sudo cat /etc/netplan/50-cloud-init.yaml
network:
  version: 2
  ethernets:
    enp0s3:
      addresses:
      - "192.168.1.12/24"
      nameservers:
        addresses:
        - 192.168.1.12
        search: [example1.test]
```

```
      routes:
      - to: "default"
        via: "192.168.1.1"
```

```
ubuntu@host1example1test:~$ sudo systemctl reboot
```

5.17.5 /etc/resolv.conf の変更の確認

起動後、/etc/resolv.conf が変更されたことを確認します。

```
ubuntu@host1example1test:~$ cat /etc/resolv.conf
nameserver 192.168.1.12
search example1.test
```

このように、参照する DNS サーバーの設定が自分自身だけになりました。

5.18 名前解決の確認

BIND を起動し、名前解決が正常に行われるかを確認します。名前解決の確認には dig コマンドが使用できます。

5.18.1 dig コマンドでホストの名前解決を確認

dig コマンドでホスト名から IP アドレスが解決されることを確認します。dig コマンドの引数に名前解決するホスト名を指定して実行します。

```
ubuntu@host1example1test:~$ dig host1.example1.test

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> host1.example1.test
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 8355
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 1f2f3acc082ceae301000000068186513f091803433fb6364 (good)
;; QUESTION SECTION:
;host1.example1.test.           IN      A

;; ANSWER SECTION:
host1.example1.test.    10800   IN      A       192.168.1.12

;; Query time: 0 msec
;; SERVER: 192.168.1.12#53(192.168.1.12) (UDP)
;; WHEN: Mon May 05 07:13:23 UTC 2025
;; MSG SIZE  rcvd: 93
```

host1.example1.test の A レコードが正しく設定されていることが確認できます。

5.18.2 正しく名前解決が行えない場合

正しく動作しない場合、結果の下から 3 行にある「SERVER」の結果が「192.168.1.12」になっているかを確認してください。

5.18.3 その他の A レコードを名前解決する

同様に、www.example1.test や mail.example1.test も確認してみます。

```
ubuntu@host1example1test:~$ dig www.example1.test

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> www.example1.test
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 50636
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: bfec6c91de160359010000006818656 8f20a05ed61bea5f8 (good)
;; QUESTION SECTION:
;www.example1.test.             IN      A

;; ANSWER SECTION:
www.example1.test.      10800   IN      A       192.168.1.12

;; Query time: 0 msec
;; SERVER: 192.168.1.12#53(192.168.1.12) (UDP)
;; WHEN: Mon May 05 07:14:48 UTC 2025
;; MSG SIZE  rcvd: 91
```

```
ubuntu@host1example1test:~$ dig mail.example1.test

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> mail.example1.test
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41210
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 6d30f32079eea33a010000006818656da02c664311bc9a76 (good)
;; QUESTION SECTION:
;mail.example1.test.            IN      A

;; ANSWER SECTION:
mail.example1.test.     10800   IN      A       192.168.1.12

;; Query time: 0 msec
;; SERVER: 192.168.1.12#53(192.168.1.12) (UDP)
;; WHEN: Mon May 05 07:14:53 UTC 2025
;; MSG SIZE  rcvd: 92
```

5.18.4 dig コマンドで NS レコードを確認

ドメイン名の後に ns を指定すると、ドメインに登録されている NS レコード (ネームサーバーの情報) と、NS レコードで返されたホストの A レコードが表示されます。

```
ubuntu@host1example1test:~$ dig example1.test ns

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> example1.test ns
;; global options: +cmd
;; Got answer:
```

```
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44674
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 2
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: c2a1daeba429a6ce01000000681865b53b764f0a7969a375 (good)
;; QUESTION SECTION:
;example1.test.                         IN      NS

;; ANSWER SECTION:
example1.test.          10800   IN      NS      host1.example1.test.

;; ADDITIONAL SECTION:
host1.example1.test.    10800   IN      A       192.168.1.12

;; Query time: 0 msec
;; SERVER: 192.168.1.12#53(192.168.1.12) (UDP)
;; WHEN: Mon May 05 07:16:05 UTC 2025
;; MSG SIZE  rcvd: 107
```

5.18.5 dig コマンドで MX レコードを確認

ドメイン名の後に mx を指定すると、ドメインに登録されている MX レコード (メールサーバーの情報) と、MX レコードで返されたホストの A レコードが表示されます。

```
ubuntu@host1example1test:~$ dig example1.test mx

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> example1.test mx
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 55253
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 2
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: aaec93e87a20012c01000000681865d3b615ea0d7704eb0a (good)
;; QUESTION SECTION:
;example1.test.                         IN      MX

;; ANSWER SECTION:
example1.test.          10800   IN      MX      10 mail.example1.test.

;; ADDITIONAL SECTION:
mail.example1.test.     10800   IN      A       192.168.1.12

;; Query time: 0 msec
;; SERVER: 192.168.1.12#53(192.168.1.12) (UDP)
;; WHEN: Mon May 05 07:16:35 UTC 2025
;; MSG SIZE  rcvd: 108
```

5.18.6 参照する DNS サーバーを指定した dig コマンドの実行

dig コマンドの引数に「[@IP アドレス]」を指定することで、一時的に参照する DNS サーバーを変更することができます。DNS サーバーの動作が正しくないような場合の原因究明に利用できます。

```
ubuntu@host1example1test:~$ dig www.example1.test @192.168.1.12
```

5.19 example2.test サーバーと test サーバーの追加

example1.test ドメインの設定ができたので、相互に名前解決ができるように仮想マシンを 2 台追加し、それぞれ example2.test ドメイン、test ドメインを管理する DNS サーバーとして設定します。

追加の手順は以下の情報を参考に、第 2 章および第 3 章の手順を繰り返し行ってください。

5.19.1 仮想マシン作成の留意事項

VirtualBox で仮想マシンを追加で 2 台作成します。

既に作成している仮想マシンと同じように作成しますが、区別が付くように仮想マシンの名前をホスト名に合わせてください。

名前
host2.example2.test
host0.test

5.19.2 OS のインストール

作成した仮想マシンに OS をインストールします。

ホスト名と IP アドレスをそれぞれのサーバーに合わせて設定します。

ホスト名	IP アドレス	DNS
host2.example2.test	192.168.1.13	192.168.1.13
host0.test	192.168.1.11	192.168.1.11

5.19.3 相互通信の確認

OS が起動したら、仮想マシン間で相互に通信ができることを ping コマンドを使って確認してください。

以下のように、各マシンから既に動作している host1.example1.test に向けて ping コマンドを実行してみます。

```
$ ping 192.168.1.12
```

5.20 example2.test ゾーンの設定

まず、example2.test ゾーンの設定を行います。手順は example1.test ゾーンを設定した以下の手順と同じです。

1. BIND のインストールを行います。
2. named.conf ファイルにゾーンを追加します。
3. ゾーンファイルを作成し、レコードなどを記述します。
4. 設定ファイルに書式間違いが無いか確認します。
5. BIND の起動を行います。
6. BIND の自動起動の設定を行います。
7. /etc/resolv.conf の参照する DNS サーバーの設定を確認します。
8. 名前解決の確認を行います。

以下、example1.test ゾーンの設定と異なるポイントです。

5.20.1 named.conf の設定

named.conf を設定します。listen-on に設定する IP アドレスが「192.168.1.13」になる点と、定義するゾーン名が「example2.test」になる点が異なります。

```
ubuntu@host2example2test:~$ sudo vi /etc/bind/named.conf
ubuntu@host2example2test:~$ cat /etc/bind/named.conf
// This is the primary configuration file for the BIND DNS server named.
//
```

```
// Please read /usr/share/doc/bind9/README.Debian for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
include "/etc/bind/named.conf.my-zones";  ← 追記設定を記載するファイルを指定
```

```
ubuntu@host2example2test:~$ sudo vi /etc/bind/named.conf.my-zones
ubuntu@host2example2test:~$ cat /etc/bind/named.conf.my-zones
zone "example2.test" IN { ← example1.testゾーンを指定
        type master;
        file "example2.test.zone"; ←
            example1.test用のゾーン定義ファイル名を指定
        allow-update { none; };
};
```

5.20.2 ゾーンファイルの作成

ゾーンファイルを作成します。

```
ubuntu@host2example2test:~$ sudo vi /etc/bind/example2.test.zone
ubuntu@host2example2test:~$ cat /etc/bind/example2.test.zone
$TTL 3H
$ORIGIN example2.test.
@       IN SOA  host2 root (
                                        2025050501      ; serial
                                        1D      ; refresh
                                        1H      ; retry
                                        1W      ; expire
                                        3H )    ; minimum
        NS      host2.example2.test.
        MX 10   mail.example2.test.

host2   A       192.168.1.13
www     A       192.168.1.13
mail    A       192.168.1.13
```

5.20.3 設定ファイルの確認

設定ファイルの書式に間違いがないか、確認します。

```
ubuntu@host2example2test:~$ named-checkconf /etc/bind/named.conf
```

```
ubuntu@host2example2test:~$ named-checkzone example2.test
    /etc/bind/example2.test.zone
zone example2.test/IN: loaded serial 2025050501
OK
```

5.20.4 Bind の再起動

bind9 を再起動し、自動起動設定を確認します。

```
ubuntu@host2example2test:~$ sudo systemctl restart bind9
ubuntu@host2example2test:~$ systemctl status bind9
● named.service - BIND Domain Name Server
     Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; preset:
         enabled)
     Active: active (running) since Mon 2025-05-05 08:44:44 UTC; 8s ago
       Docs: man:named(8)
   Main PID: 2159 (named)
     Status: "running"
      Tasks: 4 (limit: 2272)
     Memory: 4.5M (peak: 4.7M)
        CPU: 108ms
     CGroup: /system.slice/named.service
             mq2159 /usr/sbin/named -f -u bind

May 05 08:44:44 host2example2local named[2159]: command channel listening on
    ::1#953
May 05 08:44:44 host2example2local named[2159]: managed-keys-zone: loaded
    serial 2
May 05 08:44:44 host2example2local named[2159]: zone 0.in-addr.arpa/IN: loaded
    serial>
May 05 08:44:44 host2example2local named[2159]: zone 127.in-addr.arpa/IN:
    loaded seri>
May 05 08:44:44 host2example2local named[2159]: zone 255.in-addr.arpa/IN:
    loaded seri>
May 05 08:44:44 host2example2local named[2159]: zone example2.local/IN: loaded
    serial>
May 05 08:44:44 host2example2local named[2159]: zone localhost/IN: loaded
    serial 2
May 05 08:44:44 host2example2local named[2159]: all zones loaded
May 05 08:44:44 host2example2local named[2159]: running
May 05 08:44:44 host2example2local systemd[1]: Started named.service - BIND
    Domain Na>
```

```
ubuntu@host2example2test:~$ sudo systemctl is-enabled bind9
alias
```

5.20.5 ネットワーク設定の変更

Ubuntu のネットワーク設定に関連する、/etc/resolv.conf(resolved プログラム) と netplan の設定を変更します。

```
ubuntu@host2example2test:~$ sudo ln -fs /run/systemd/resolve/stub-resolv.conf
    /etc/resolv.conf
```

```
ubuntu@host1example1test:~$ sudo vi /etc/systemd/resolved.conf
[Resolve]
DNSStubListener=no
```

```
ubuntu@host2example2test:~$ sudo vi /etc/netplan/50-cloud-init.yaml

ubuntu@host2example2test:~$ sudo cat /etc/netplan/50-cloud-init.yaml
network:
  version: 2
  ethernets:
    enp0s3:
      addresses:
      - "192.168.1.13/24"
      nameservers:
```

```
        addresses:
        - 192.168.1.13
        search: [example2.test]
      routes:
      - to: "default"
        via: "192.168.1.1"
```

5.20.6 システムの再起動

ネットワーク設定を反映する為、Ubuntu を再起動します。

```
ubuntu@host2example2test:~$ sudo systemctl reboot
```

5.20.7 DNS レコードの確認

再起動完了後、DNS サーバで正常に回答が得られるか確認します。

```
ubuntu@host2example2test:~$ dig host2.example2.test

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> host2.example2.test
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25440
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: c3e461a7154b7e370100000068187e456ea67ac7c750d713 (good)
;; QUESTION SECTION:
;host2.example2.test.           IN      A

;; ANSWER SECTION:
host2.example2.test.    10800   IN      A       192.168.1.13

;; Query time: 0 msec
;; SERVER: 192.168.1.13#53(192.168.1.13) (UDP)
;; WHEN: Mon May 05 09:00:53 UTC 2025
;; MSG SIZE  rcvd: 93

ubuntu@host2example2test:~$ dig www.example2.test

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> www.example2.test
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24422
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: ce1045269f0ded900100000068187e4a0e6c1acef196fa87 (good)
;; QUESTION SECTION:
;www.example2.test.             IN      A

;; ANSWER SECTION:
www.example2.test.      10800   IN      A       192.168.1.13

;; Query time: 0 msec
```

```
;; SERVER: 192.168.1.13#53(192.168.1.13) (UDP)
;; WHEN: Mon May 05 09:00:58 UTC 2025
;; MSG SIZE  rcvd: 91

ubuntu@host2example2test:~$ dig mail.example2.test

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> mail.example2.test
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21839
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 5bb70f42b0d26cb70100000068187e4f81181f7ae37bf7ea (good)
;; QUESTION SECTION:
;mail.example2.test.            IN      A

;; ANSWER SECTION:
mail.example2.test.     10800   IN      A       192.168.1.13

;; Query time: 0 msec
;; SERVER: 192.168.1.13#53(192.168.1.13) (UDP)
;; WHEN: Mon May 05 09:01:03 UTC 2025
;; MSG SIZE  rcvd: 92

ubuntu@host2example2test:~$ dig example2.test ns

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> example2.test ns
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46180
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 2
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 1d882f4fe9dac0e50100000068187e7ba7cf1023cd3734e8 (good)
;; QUESTION SECTION:
;example2.test.                 IN      NS

;; ANSWER SECTION:
example2.test.          10800   IN      NS      host2.example2.test.

;; ADDITIONAL SECTION:
host2.example2.test.    10800   IN      A       192.168.1.13

;; Query time: 0 msec
;; SERVER: 192.168.1.13#53(192.168.1.13) (UDP)
;; WHEN: Mon May 05 09:01:47 UTC 2025
;; MSG SIZE  rcvd: 107

ubuntu@host2example2test:~$ dig example2.test mx

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> example2.test mx
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23307
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 2
;; WARNING: recursion requested but not available
```

```
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 32ca3ae6331aa84a0100000068187e80fb23bc833ff7dfe2 (good)
;; QUESTION SECTION:
;example2.test.                         IN      MX

;; ANSWER SECTION:
example2.test.          10800   IN      MX      10 mail.example2.test.

;; ADDITIONAL SECTION:
mail.example2.test.     10800   IN      A       192.168.1.13

;; Query time: 0 msec
;; SERVER: 192.168.1.13#53(192.168.1.13) (UDP)
;; WHEN: Mon May 05 09:01:52 UTC 2025
;; MSG SIZE  rcvd: 108

ubuntu@host2example2test:~$ dig www.example2.test @192.168.1.13

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> www.example2.test @192.168.1.13
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16932
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: c852859fb4f1368b0100000068187e8ad7718da8b1b2098f (good)
;; QUESTION SECTION:
;www.example2.test.             IN      A

;; ANSWER SECTION:
www.example2.test.      10800   IN      A       192.168.1.13

;; Query time: 0 msec
;; SERVER: 192.168.1.13#53(192.168.1.13) (UDP)
;; WHEN: Mon May 05 09:02:02 UTC 2025
;; MSG SIZE  rcvd: 91
```

5.21 上位 test ゾーンの DNS コンテンツサーバーの設定

example1.test ゾーン、example2.test ゾーンの設定が完了したら、上位ゾーンにあたる test ゾーンの設定を行います。手順は example1.test ゾーンや example2.test ゾーンを設定した以下の手順と同じです。

1. BIND のインストールを行います。
2. named.conf ファイルにゾーンを追加します。
3. ゾーンファイルを作成し、レコードなどを記述します。
4. 設定ファイルに書式間違いが無いか確認します。
5. BIND の起動を行います。
6. BIND の自動起動の設定を行います。
7. /etc/resolv.conf の参照する DNS サーバーの設定を確認します。
8. 名前解決の確認を行います。

以下、example1.test ゾーンの設定と異なるポイントです。

5.21.1 named.conf の設定

named.conf を設定します。listen-on に設定する IP アドレスが 192.168.1.11 になる点と、定義するゾーン名が.test になる点が異なります。また、この DNS サーバーは DNS キャッシュサーバーとしても動作させるので、recursion の設定を「yes;」のままにしておきます。

```
ubuntu@host0test:~$ sudo vi /etc/bind/named.conf
ubuntu@host0test:~$ cat /etc/bind/named.conf
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
include "/etc/bind/named.conf.my-zones";
```

```
ubuntu@host0test:~$ sudo vi /etc/bind/named.conf.my-zones
ubuntu@host0test:~$ cat /etc/bind/named.conf.my-zones
zone "test" IN {
        type master;
        file "test.zone";
        allow-update { none; };
};
```

```
ubuntu@host0test:~$ sudo vi /etc/bind/named.conf.options
ubuntu@host0test:~$ cat /etc/bind/named.conf.options
options {
        directory "/var/cache/bind";
        //directory "/etc/bind";
        listen-on port 53 { 127.0.0.1; 192.168.1.11; };
        listen-on-v6 port 53 { ::1; };
        allow-query     { any; };

        recursion yes;

        dnssec-validation auto;
};
```

5.21.2 ゾーンファイルの作成

ゾーンファイルを作成します。example1.test と example2.test を委譲する NS レコードも記載します。

```
ubuntu@host0test:~$ sudo vi /etc/bind/test.zone
ubuntu@host0test:~$ cat /etc/bind/test.zone
$TTL 3H
$ORIGIN test.
@       IN SOA  host0 root (
                                        2025050501      ; serial
                                        1D      ; refresh
                                        1H      ; retry
                                        1W      ; expire
                                        3H )    ; minimum
        NS      host0.test.
example1.test. NS      host1.example1.test.
example2.test. NS      host2.example2.test.
```

```
host0   A       192.168.1.11
host1.example1.test.     A       192.168.1.12
host2.example2.test.     A       192.168.1.13
```

5.21.3 設定ファイルの確認

設定ファイルの書式に間違いがないか、確認します。

```
ubuntu@host0test:~$ named-checkconf /etc/bind/named.conf
```

```
ubuntu@host0test:~$ named-checkzone test /var/cache/bind/test.zone
zone test/IN: loaded serial 2025050501
OK
```

5.21.4 Bind の再起動

bind9 を再起動し、自動起動設定を確認します。

```
ubuntu@host0test:~$ sudo systemctl restart bind9
[sudo] password for ubuntu:
ubuntu@host0test:~$ systemctl status bind9
● named.service - BIND Domain Name Server
     Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; preset: en>
     Active: active (running) since Mon 2025-05-05 15:18:57 UTC; 7s ago
       Docs: man:named(8)
   Main PID: 1956 (named)
     Status: "running"
      Tasks: 4 (limit: 2272)
     Memory: 5.1M (peak: 5.4M)
        CPU: 241ms
     CGroup: /system.slice/named.service
             mq1956 /usr/sbin/named -f -u bind

May 05 15:18:57 host0test named[1956]: command channel listening on ::1#953
May 05 15:18:57 host0test named[1956]: managed-keys-zone: loaded serial 16
May 05 15:18:57 host0test named[1956]: zone test/IN: loaded serial 2025050501
May 05 15:18:57 host0test named[1956]: zone 0.in-addr.arpa/IN: loaded serial 1
May 05 15:18:57 host0test named[1956]: zone 127.in-addr.arpa/IN: loaded serial 1
May 05 15:18:57 host0test named[1956]: zone 255.in-addr.arpa/IN: loaded serial 1
May 05 15:18:57 host0test named[1956]: zone localhost/IN: loaded serial 2
May 05 15:18:57 host0test named[1956]: all zones loaded
May 05 15:18:57 host0test named[1956]: running
May 05 15:18:57 host0test systemd[1]: Started named.service - BIND Domain Name
```

```
ubuntu@host0test:~$ sudo systemctl is-enabled bind9
alias
```

5.21.5 ネットワーク設定の変更

Ubuntu のネットワーク設定に関連する、/etc/resolv.conf(resolved プログラム) と netplan の設定を変更します。

```
ubuntu@host0test:~$ sudo ln -fs /run/systemd/resolve/stub-resolv.conf
    /etc/resolv.conf
```

```
ubuntu@host0test:~$ sudo vi /etc/systemd/resolved.conf
[Resolve]
DNSStubListener=no
```

```
ubuntu@host0test:~$ sudo vi /etc/netplan/50-cloud-init.yaml

ubuntu@host0test:~$ sudo cat /etc/netplan/50-cloud-init.yaml
network:
  version: 2
  ethernets:
    enp0s3:
      addresses:
      - "192.168.1.11/24"
      nameservers:
        addresses:
        - 192.168.1.11
        search: [test]
      routes:
      - to: "default"
        via: "192.168.1.1"
```

5.21.6 システムの再起動

ネットワーク設定を反映する為、Ubuntu を再起動します。

```
ubuntu@host0test:~$ sudo systemctl reboot
```

5.21.7 DNS レコードの確認

再起動完了後、DNS サーバで正常に回答が得られるか確認します。

```
ubuntu@host0test:~$ dig host0.test

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> host0.test
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 51412
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: cd4484ef3fccd0ce010000006818d88f87036ee71c3ae627 (good)
;; QUESTION SECTION:
;host0.test.                    IN      A

;; ANSWER SECTION:
host0.test.             10800   IN      A       192.168.1.11

;; Query time: 0 msec
;; SERVER: 192.168.1.11#53(192.168.1.11) (UDP)
;; WHEN: Mon May 05 15:26:07 UTC 2025
;; MSG SIZE  rcvd: 83
```

```
ubuntu@host0test:~$ dig example1.test ns

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> example1.test ns
;; global options: +cmd
```

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 34417
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: a0db2494cbd2f000010000006818d8a5b6107ddb44449d5d (good)
;; QUESTION SECTION:
;example1.test.                 IN      NS

;; ANSWER SECTION:
example1.test.          10800   IN      NS      host1.example1.test.

;; Query time: 11 msec
;; SERVER: 192.168.1.11#53(192.168.1.11) (UDP)
;; WHEN: Mon May 05 15:26:29 UTC 2025
;; MSG SIZE  rcvd: 90
```

```
ubuntu@host0test:~$ dig example2.test ns

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> example2.test ns
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31608
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: b3d43799c24d59db010000006818d8b693cc76881afc95af (good)
;; QUESTION SECTION:
;example2.test.                 IN      NS

;; ANSWER SECTION:
example2.test.          10800   IN      NS      host2.example2.test.

;; Query time: 40 msec
;; SERVER: 192.168.1.11#53(192.168.1.11) (UDP)
;; WHEN: Mon May 05 15:26:46 UTC 2025
;; MSG SIZE  rcvd: 90
```

5.21.8 example1.test と exapmle2.test の DNS サーバの名前解決先の変更

example1.test・example2.test を管理する DNS サーバのネットワーク設定のうち、名前解決を行う DNS サーバを上位 DNS にあたる test ゾーンを管理する「192.168.1.11」に変更し、設定を反映させる為、Ubuntu を再起動します。

```
ubuntu@host1example1test:~$ sudo vi /etc/netplan/50-cloud-init.yaml
```

```
ubuntu@host1example1test:~$ sudo cat /etc/netplan/50-cloud-init.yaml
network:
  version: 2
  ethernets:
    enp0s3:
      addresses:
      - "192.168.1.12/24"
      nameservers:
        addresses:
        - 192.168.1.11
        search: [example1.test]
```

```
      routes:
      - to: "default"
        via: "192.168.1.1"
```

```
ubuntu@host1example1test:~$ sudo systemctl reboot
```

```
ubuntu@host2example2test:~$ sudo vi /etc/netplan/50-cloud-init.yaml
```

```
ubuntu@host2example2test:~$ sudo cat /etc/netplan/50-cloud-init.yaml
network:
  version: 2
  ethernets:
    enp0s3:
      addresses:
      - "192.168.1.13/24"
      nameservers:
        addresses:
        - 192.168.1.11
        search: [example2.test]
      routes:
      - to: "default"
        via: "192.168.1.1"
```

```
ubuntu@host2example2test:~$ sudo systemctl reboot
```

5.22 次章のメールサーバ構築に向けて

MX レコードの名前解決は、次の章で行うメールサーバーを動かす上での前提条件となります。きちんと名前解決が行えていることを確認した上で次の章に進んでください。

5.23 うまく名前解決が行えない場合には

サーバー相互の名前解決がうまく動作しない場合には、以下の点を確認してください。

- DNS サーバーが起動している
- DNS サーバーが名前解決に応答している
- ping コマンドでお互いに IP 通信が行えている
- ファイアウォールの設定が変更されている
- 参照する DNS サーバーの設定が「192.168.56.100」に変更されている

6 メールサーバーの構築

第 6 章では、メールのやり取りが行えるようメールサーバーを設定します。まずは Postfix を使って、メールサーバー同士でメールのやり取りが行えるように設定します。さらに IMAP サーバーの Dovecot とメールクライアントソフトウェアを使って、より実践的なメール環境を構築します。

6.1 用語集

メールサーバー

電子メールのサービスを行います。クライアントよりメールを受け取り、相手先のメールサーバーまで送ります。また、受信用のメールサーバーでは、送ってきたメールを蓄積しておいて、クライアントの要求に応じて応答します。

MTA（Mail Transfer Agent）

メールの転送を行うプログラムです。Sendmail や Postfix などが代表例です。

SMTP（Simple Mail Transfer Protocol）

電子メールの送信、転送のときに利用されるプロトコルのことです。

SMTP 認証

SMTP でのメールを送信する際に認証を行う機構です。迷惑メール対策としてのメール中継の制限を、この認証機能で許可する、といった利用方法があります。

Postfix

MTA として動作するサーバープログラムです。Linux や Unix のシステムで古くから使われてきた Sendmail よりもセキュリティが高く、高速に動作すると言われています。

POP3（Post Office Protocol version3）

クライアントが電子メールを取り寄せるときに利用されるプロトコルです。シンプルな設計で、IMAP4 と比べて機能が少ないです。

IMAP4（Internet Message Access Protocol 4）

クライアントが電子メールを取り寄せるときに利用されるプロトコルです。メールのフォルダ機能サポート等、多機能です。

Dovecot

POP3 や IMAP4 のサーバー機能を提供するプログラムです。

6.2 メールのやり取りの仕組み

インターネット上で沢山の人が電子メールを利用していますが、電子メールは以下の手順でやり取りされています。括弧内はそれぞれの手順に関わる動作やプロトコルです。

1. 送信者のメールクライアントから送信用メールサーバーにメールを送信します（SMTP 認証）
2. 送信用メールサーバーは受信用メールサーバーにメールを転送します（SMTP）
3. 受信用メールサーバーは宛先アドレスのメールボックスにメールを配送します（メール配送）
4. 受信者のメールクライアントでメールボックスのあるメールサーバーに接続します（POP3 や IMAP4）
5. 受信者のメールクライアントがメールを受信して、メールを見ることができます

6.3 メールの送信

メールの送信は、メールサーバーを介してやり取りが行われます。メールサーバーがメールを受け取ると、宛先のメールアドレスを担当しているメールサーバーに転送され、最終的に宛先のメールボックスに入れられます。Thunderbird や Outlook のようなメールクライアントから送信したメールでも、Gmail のような Web メールから送信したメールでも、動作原理は同じです。

メールの送信に関わるプログラムやプロトコルについて解説します。

6.3.1 MTA（Mail Transfer Agent）

メールクライアントから送信されたメールは、送信用メールサーバーから宛先の受信用メールサーバーに転送されます。このメールの転送を行うプログラムを MTA と呼びます。本教科書では Postfix という MTA を利用します。他に有名な MTA として Sendmail があります。

6.3.2 MDA（Mail Delivery Agent）

受信用メールサーバーが受信したメールを宛先アドレスのメールボックスに配送するプログラムを MDA と呼びます。MTA である Postfix が MDA の機能も受け持ちます。他に MDA として Procmail があります。

6.3.3 MUA（Mail User Agent）

メールの利用者が、メールの送信や受信を行うプログラム (メールクライアントソフトウェア) を MUA と呼びます。Thunderbird などがあります。Web メールも MUA の一種となります。

6.3.4 SMTP（Simple Mail Transfer Protocol）の拡張

メールクライアントからのメール送信や、メールサーバー間のメールの転送は、SMTP というプロトコルでやり取りされています。SMTP はかなり昔に設計、定義されたプロトコルのため、認証やアクセス制限などが無く、勝手にメールサーバーを利用されて迷惑メールを送られてしまう問題がありました。このような問題を解決するために ESMTP（拡張 SMTP）が定義されました。SMTP と呼ぶ場合、この ESMTP で定義された機能も含んでいることがあります。

6.3.5 SMTP 認証（SMTP Authentication）とリレー

SMTP 認証は ESMTP の機能のうちの 1 つです。通常の SMTP には認証機能が無いため、送信元の IP アドレスを制限するなど適切に設定されていないと迷惑メール送信の踏み台とされてしまう問題があります。SMTP 認証は、メールの送信時に認証を行い、認証が行われた場合のみメールの送信を受け付けます。受け付けたメールを宛先の受信用メールサーバーに転送することをリレーと呼びます。

6.4 メールの受信

送信されたメールが宛先のメールボックスに配送されると、受信者はメールを受信して読むことができます。メールの受信に関係するいくつかの事項について解説します。

6.4.1 メールの配送

受信用メールサーバーがメールを受け取り、宛先アドレスのメールボックスにメールを届けることを配送と呼びます。メールボックスが無い場合、宛先不明として送信用メールサーバーにエラーを返します。メールの配送を行うソフトウェアを MDA（Mail Delivery Agent）と呼びます。

6.4.2 ローカル配送

送信側と受信側が同じメールサーバーを使用している場合、メールは外部のメールサーバーに転送する必要がなく、すぐに宛先アドレスのメールボックスに配送されます。これをローカル配送と呼びます。

6.4.3 POP3（Post Office Protocol version 3）

POP3 は電子メールを受信するときに利用するプロトコルです。非常にシンプルなプロトコルで、ユーザー名、パスワードを利用して接続し、メールの内容を受信します。

6.4.4　IMAP4（Internet Message Access Protocol 4）

IMAP4 も POP3 同様、メールを受信するときに利用するプロトコルです。IMAP4 は POP3 に比べて機能が豊富で、大きな特徴としてフォルダ機能をサポートしていることが挙げられます。メールを IMAP サーバーに残しておくことができるので、メールクライアントと Web メールを併用したりすることもできますが、その分だけメールサーバーのストレージを消費するので容量管理が必要になります。

6.5　メールサーバーの構築

メールサーバーの構築は、送信用メールサーバーと受信用メールサーバーの 2 台を構築します。それぞれのサーバーを以下のように設定します。

- 1 台のマシンで、メールサーバーとメールクライアントの 1 台 2 役とします。
- MTA として Postfix をインストールし、送信用メールサーバーおよび受信用メールサーバーとして設定します。
- IMAP サーバーとして Dovecot をインストールし、設定します。
- それぞれのメールサーバーにメールアカウントを作成します。
- メールクライアントとして Thunderbird をインストールし、サーバー自身を送受信用サーバーとして設定します。

通常、メールサーバーとメールクライアントは別々のマシンを用意し、その間を SMTP や POP3、IMAP4 などで接続しますが、実習では同じマシンで行います。

構築は、Postfix の設定と mail コマンドによるメールの送受信、Dovecot と Thunderbird の設定とメールの送受信の 2 段階に分けて行います。

6.5.1　mail コマンドを利用したメールの送受信

Postfix の設定が完了したら、動作確認として mail コマンドを使ってメールのやり取りを行います。メールクライアントの設定を必要としないので、MTA が正しく設定されていることを確認するのに適しています。

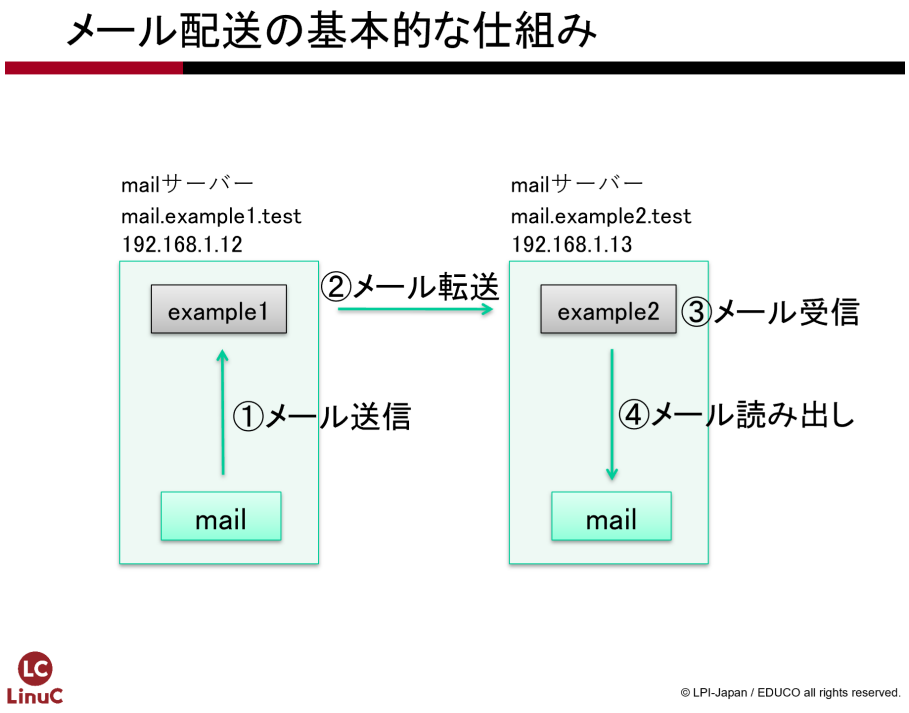

図 41: メール配送の基本的な仕組み

6.5.2　Thunderbird を利用したメールの送受信

Thunderbird をメールクライアントとして設定し、SMTP 認証によるメール送信や、Dovecot による IMAP サーバーからのメール受信を行います。

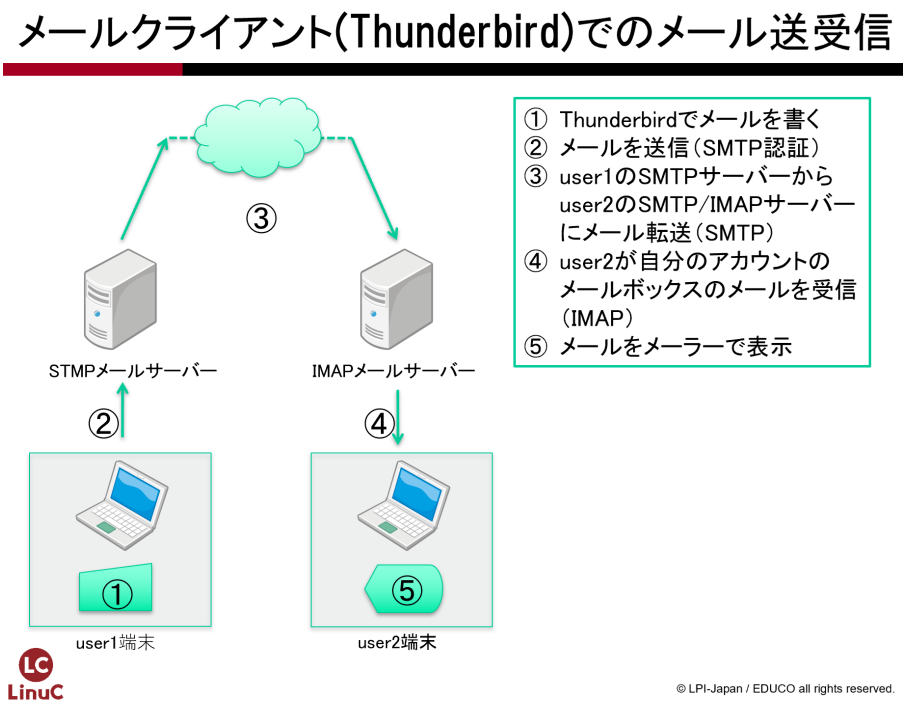

図 42: Thunderbird でのメール送受信

6.6　Postfix のインストール

Postfix を apt コマンドでインストールをします。また、IMAP サーバーとして使用する dovecot パッケージも同時にインストールしておきます。

6.6.1　パッケージのインストール

apt コマンドで必要なパッケージをインストールします。Postfix のインストールでは設定のタイプの選択をしますが、「Internet Site」としておけば大丈夫です。また、それぞれのドメインを入力します。

```
$ sudo apt install postfix dovecot-pop3d dovecot-imapd
```

6.7　Postfix の設定ファイル main.cf の設定

Postfix の設定ファイルは/etc/postfix/main.cf です。次のパラメータを探して設定します。

```
$ sudo vi /etc/postfix/main.cf
```

パラメータによっては、デフォルト値が設定されているので、その場合には書き換えます。コメントアウトされた形で記述されている場合には、コメントアウトを外して設定を有効にした上で値を記述します。

「smtpd_sasl_auth_enable」と「smtpd_recipient_restrictions」は、main.cf に記述されていないので、ファイルの最後に追加しておきます。

host1 と host2 に、それぞれ以下のように設定します。

host1 の設定

項目	設定値
myhostname	mail.example1.test

項目	設定値
mydomain	example1.test
inet_interfaces	localhost, 192.168.1.12
mydestination	$mydomain

host2 の設定

項目	設定値
myhostname	mail.example2.test
mydomain	example2.test
inet_interfaces	localhost, 192.168.1.13
mydestination	$mydomain

それぞれのパラメータの意味と設定値は以下の通りです。

6.7.1 myhostname

メールサーバーのホスト名を設定します。

host1 の設定

```
myhostname = mail.example1.test
```

host2 の設定

```
myhostname = mail.example2.test
```

6.7.2 mydomain

メールサーバーのドメイン名を設定します。

host1 の設定

```
mydomain = example1.test
```

host2 の設定

```
mydomain = example2.test
```

6.7.3 inet_interfaces

メールを受け付けるネットワークインターフェースの IP アドレスを設定します。localhost の記述を忘れると、自分自身からのメールを受け付けなくなるので注意が必要です。

host1 の設定

```
inet_interfaces = localhost, 192.168.1.12
```

host2 の設定

```
inet_interfaces = localhost, 192.168.1.13
```

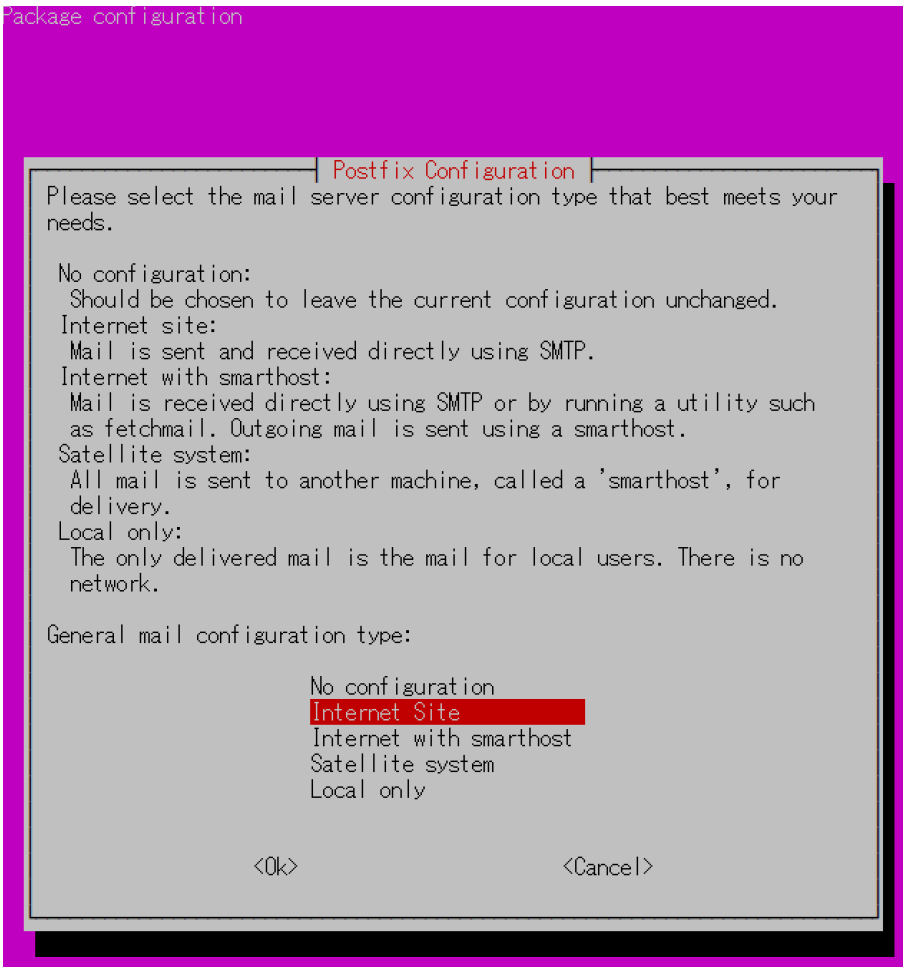

図 43: Postfix Configuration のタイプ

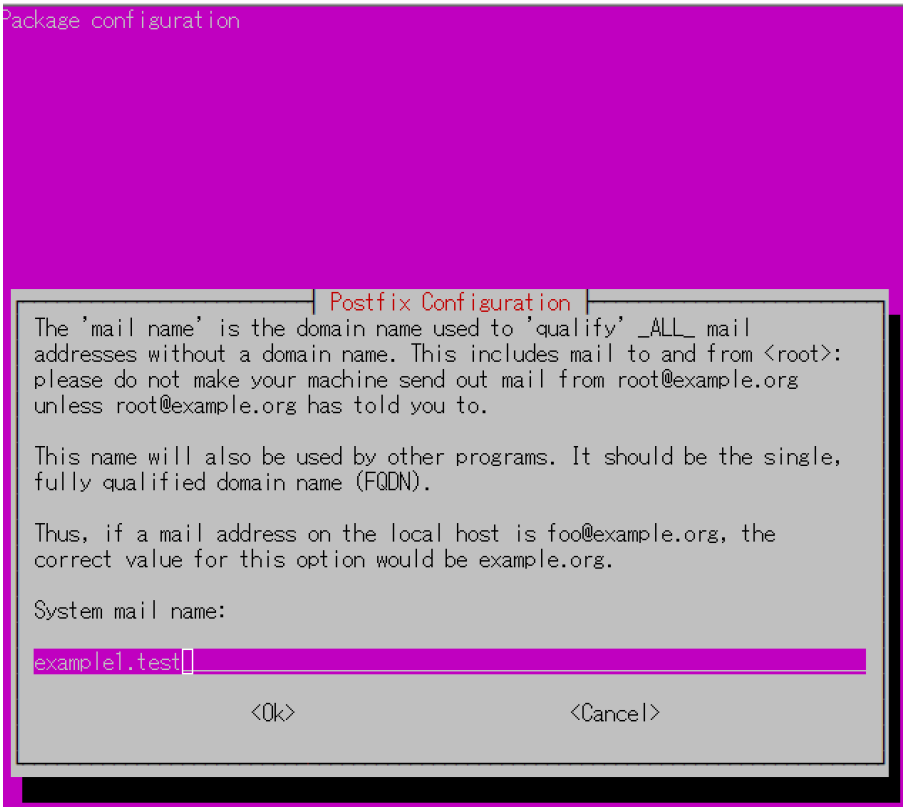

図 44: Postfix Configuration(ドメイン設定)

6.7.4 mydestination

メールを受信するドメイン名を設定します。宛先メールアドレス（アカウント名@ドメイン名）がこのドメイン名になっているメールのみ受け取ります。

host1 と host2 共通の設定

```
mydestination = $mydomain
```

6.7.5 mynetworks

Postfix が信頼するネットワークの範囲を指定します。今回の構成で使用している「192.168.1.0/24」と localhost のネットワークである「127.0.0.0/8」を記載します。

host1 と host2 共通の設定

```
mynetworks = 192.168.1.0/24 127.0.0.0/8
```

6.8 書式のチェック

/etc/postfix/main.cf の修正ができたら、書式チェックを行っておきます。

```
$ sudo postfix check
```

書式が正しい場合には、何も表示されません。エラーが表示された場合には、エラー内容をよく見て修正します。例えば、パラメータ名を記載ミスした場合には、以下のように指摘されます。

```
/usr/sbin/postconf: warning: /etc/postfix/main.cf: unused parameter:
    mynetworkss=192.168.1.0/24 127.0.0.0/8
```

6.9 Postfix の起動

設定の変更内容を反映させるため、postfix サービスを再起動します。

```
ubuntu@host1example1test:~$ sudo systemctl restart postfix
```

6.10 Postfix の自動起動

Postfix の自動起動の設定を確認しておきます。

```
ubuntu@host1example1test:~$ systemctl is-enabled postfix
enabled
```

これで Postfix の設定は完了です。

6.11 アカウントの作成

メールのやり取りを行うためのアカウントを作成します。アカウントは host1 と host2 の双方で行います。

6.11.1 host1 に user1 を作成

host1 で user1 というアカウントを作成します。このアカウントは user1@example1.test というメールアドレスになります。

```
ubuntu@host1example1test:~$ sudo adduser user1
[sudo] password for ubuntu:
info: Adding user `user1' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `user1' (1001) ...
info: Adding new user `user1' (1001) with group `user1 (1001)' ...
info: Creating home directory `/home/user1' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for user1
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
info: Adding new user `user1' to supplemental / extra groups `users' ...
info: Adding user `user1' to group `users' ...
```

6.11.2 host2 に user2 を作成

host2 で user2 というアカウントを作成します。このアカウントは user2@example2.test というメールアドレスになります。

```
ubuntu@host2example2test:~$ sudo adduser user2
[sudo] password for ubuntu:
info: Adding user `user2' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `user2' (1001) ...
info: Adding new user `user2' (1001) with group `user2 (1001)' ...
info: Creating home directory `/home/user2' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for user2
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
info: Adding new user `user2' to supplemental / extra groups `users' ...
info: Adding user `user2' to group `users' ...
```

6.11.3 メールエイリアスのデータベース構築

newaliases コマンドを実施し、メールエイリアスデータベースを構築します。

```
ubuntu@host1example1test:~$ sudo newaliases
```

6.12 mail コマンドを使ったメール送受信のテスト

メールの送受信が行えるかテストを行います。メールの送受信は作成したユーザー user1 と user2 で行います。ユーザーで操作できるよう user1/user2 でリモートアクセス (ssh) を行い、メールの送受信には mail コマンドを使用します。mail コマンドを利用できるようにパッケージ「mailutils」をインストールします。

```
ubuntu@host1example1test:~$ sudo apt install mailutils
```

6.12.1 ログの確認用端末の設定（任意）

メールサーバーはバックグラウンドで動作するため、どのように動いているのか確認するためにはログを参照する必要があります。

tail コマンドに-f オプションを付けて実行すると、ログが書き込まれる毎に再読み込みされて最新のログを閲覧できます。

1. メールサーバへリモートアクセスします。
2. tail コマンドを実行して/var/log/maillog を表示します。

```
ubuntu@host1example1test:~$ sudo tail -f /var/log/mail.log
```

ログの確認後は、「ctrl+c」で終了できます

6.12.2 user1@example1.test から user2@example2.test へメール送信

mail コマンドを使って、host1 の user1 から user2@example2.test へメールを送信します。

```
user1@host1example1test:~$ mail user2@example2.test ←
    mailコマンドの引数に宛先のアドレスを指定
Cc:
Subject: Test mail from user1 ← Subjectを入力
This is test mail from user1 ← メッセージ本文を入力
^D ← メッセージ本文の入力が終わったらCtrl+dを入力
```

6.12.3 user2 のメール着信確認

mail コマンドを使って、host2.example2.test の user2 にメールが届いているかを確認します。

```
user2@host2example2test:~$ mail
"/var/mail/user2": 1 message 1 new
>N   1 user1@host1example Tue May  6 00:09  16/655   Test mail from user1
& 1 ← 1を入力
Return-Path: <user1@host1example1test>
X-Original-To: user2@example2.test
Delivered-To: user2@example2.test
Received: from mail.example1.test (unknown [192.168.1.12])
        by mail.example2.test (Postfix) with ESMTPS id 8B71822224
        for <user2@example2.test>; Tue,  6 May 2025 00:09:08 +0000 (UTC)
Received: by mail.example1.test (Postfix, from userid 1001)
        id 351E8401B; Tue,  6 May 2025 00:09:08 +0000 (UTC)
To: <user2@example2.test>
Subject: Test mail from user1
User-Agent: mail (GNU Mailutils 3.17)
Date: Tue,  6 May 2025 00:09:08 +0000
Message-Id: <20250506000908.351E8401B@mail.example1.test>
From: user1@host1example1test

This is test mail from user1

& q ← qを入力
```

このように、host1.example1.test から host2.example2.test にメールが送られていることがわかります。

反対に user2@example2.test から user1@example1.test へのメールも送信できることを確認してみましょう。

6.13 メールクライアントソフトでのメールの送受信

通常のメールサーバーの運用では、メールの利用者はメールクライアントを使用してメールの送受信を行います。送信は SMTP、受信は IMAP4 や POP3 をプロトコルとして使用します。

IMAP サーバーを利用してメールを受信できるよう、POP3/IMAP サーバーである Dovecot を設定し、メールを送受信してみます。また、メールクライアントには Virtualbox を動作させているホスト PC 上に Thunderbird などのソフトウェアをインストールしメールの送受信を行います。

以下の作業は host1 で行いますが、host2 でも設定して双方向でメールのやり取りができるようにしてもよいでしょう。

6.14 Dovecot の設定

IMAP サーバーとして Dovecot の設定を行います。

今回設定するファイルは/etc/dovecot/dovecot.conf と、/etc/dovecot/conf.d ディレクトリ以下に分かれている以下のファイルです。

- /etc/dovecot/dovecot.conf（変更不要）

- /etc/dovecot/conf.d/10-mail.conf
- /etc/dovecot/conf.d/10-auth.conf
- /etc/dovecot/conf.d/10-ssl.conf

6.14.1 /etc/dovecot/dovecot.conf（変更不要）

全体的な設定ファイルです。デフォルトの設定がコメントアウトで記述されています。特に変更は必要ありません。

```
$ sudo vi /etc/dovecot/dovecot.conf
（略）
# Protocols we want to be serving.
#protocols = imap pop3 lmtp submission  ← IMAP/POP3/LMTP/SMTP
    submissionが使用可能

# A comma separated list of IPs or hosts where to listen in for connections.
# "*" listens in all IPv4 interfaces, "::" listens in all IPv6 interfaces.
# If you want to specify non-default ports or anything more complex,
# edit conf.d/master.conf.
#listen = *, :: ← ホストのすべてのIPアドレスで接続を受け付ける
```

6.14.2 /etc/dovecot/conf.d/10-mail.conf

メールボックスの位置などを設定するファイルです。

今回は mbox 形式のメールボックスを指定します。また、メールボックスへのアクセス権限を設定します。

```
ubuntu@host1example1test:~$ sudo vi /etc/dovecot/conf.d/10-mail.conf
```

```
（略）
#   mail_location = maildir:~/Maildir
#   mail_location = mbox:~/mail:INBOX=/var/mail/%u
#   mail_location = mbox:/var/mail/%d/%1n/%n:INDEX=/var/indexes/%d/%1n/%n
#
# <doc/wiki/MailLocation.txt>
#
#mail_location =
mail_location = mbox:~/mail:INBOX=/var/mail/%u  ←
    postfixでも使われるmbox形式によるメールの保存を行う

（略）

# Group to enable temporarily for privileged operations. Currently this is
# used only with INBOX when either its initial creation or dotlocking fails.
# Typically this is set to "mail" to give access to /var/mail.
#mail_privileged_group =
mail_privileged_group = mail ← 権限が必要な動作はmailグループとして行う

# Grant access to these supplementary groups for mail processes. Typically
# these are used to set up access to shared mailboxes. Note that it may be
# dangerous to set these if users can create symlinks (e.g. if "mail" group is
# set here, ln -s /var/mail ~/mail/var could allow a user to delete others'
# mailboxes, or ln -s /secret/shared/box ~/mail/mybox would allow reading it).
#mail_access_groups =
mail_access_groups = mail ← mailグループにアクセス権限を与えるように追加
（略）
```

6.14.3 /etc/dovecot/conf.d/10-auth.conf

認証を設定するファイルです。

今回は暗号化していない平文での認証を許可します。

```
ubuntu@host1example1test:~$ sudo vi /etc/dovecot/conf.d/10-auth.conf
```

```
##
## Authentication processes
##

# Disable LOGIN command and all other plaintext authentications unless
# SSL/TLS is used (LOGINDISABLED capability). Note that if the remote IP
# matches the local IP (ie. you're connecting from the same computer), the
# connection is considered secure and plaintext authentication is allowed.
# See also ssl=required setting.
#disable_plaintext_auth = yes
disable_plaintext_auth = no ← noに変更
（略）
```

6.14.4 /etc/dovecot/conf.d/10-ssl.conf

SSL/TLS を設定するファイルです。

今回は SSL/TLS 暗号化をしませんので、SSL/TLS 暗号化の利用を停止しておきます。

```
ubuntu@host1example1test:~$ sudo vi /etc/dovecot/conf.d/10-ssl.conf
```

```
##
## SSL settings
##

# SSL/TLS support: yes, no, required. <doc/wiki/SSL.txt>
# disable plain pop3 and imap, allowed are only pop3+TLS, pop3s, imap+TLS and
    imaps
# plain imap and pop3 are still allowed for local connections
ssl = no    ← yesをnoに変更
```

6.15 Dovecot の再起動

dovecot サービスを再起動します。

```
ubuntu@host1example1test:~$ sudo systemctl restart dovecot
```

6.15.1 Dovecot の自動起動

自動起動の設定を確認します。

```
ubuntu@host1example1test:~$ sudo systemctl is-enabled dovecot
enabled
```

6.16 メールクライアントの設定

次に、ホスト PC にてメールクライアントソフトウェアの Thunderbird の設定を行いますが、まずホスト PC が利用する DNS サーバの情報を、構築済みの DNS サーバ「192.168.1.11」に変更します。

6.16.1 Thunderbird の起動

Thunderbird を起動します。

Thunderbird が起動すると別途 Web ブラウザが開いて Thunderbird の Web ページが表示されますが、Web ブラウザごと閉じて構いません。

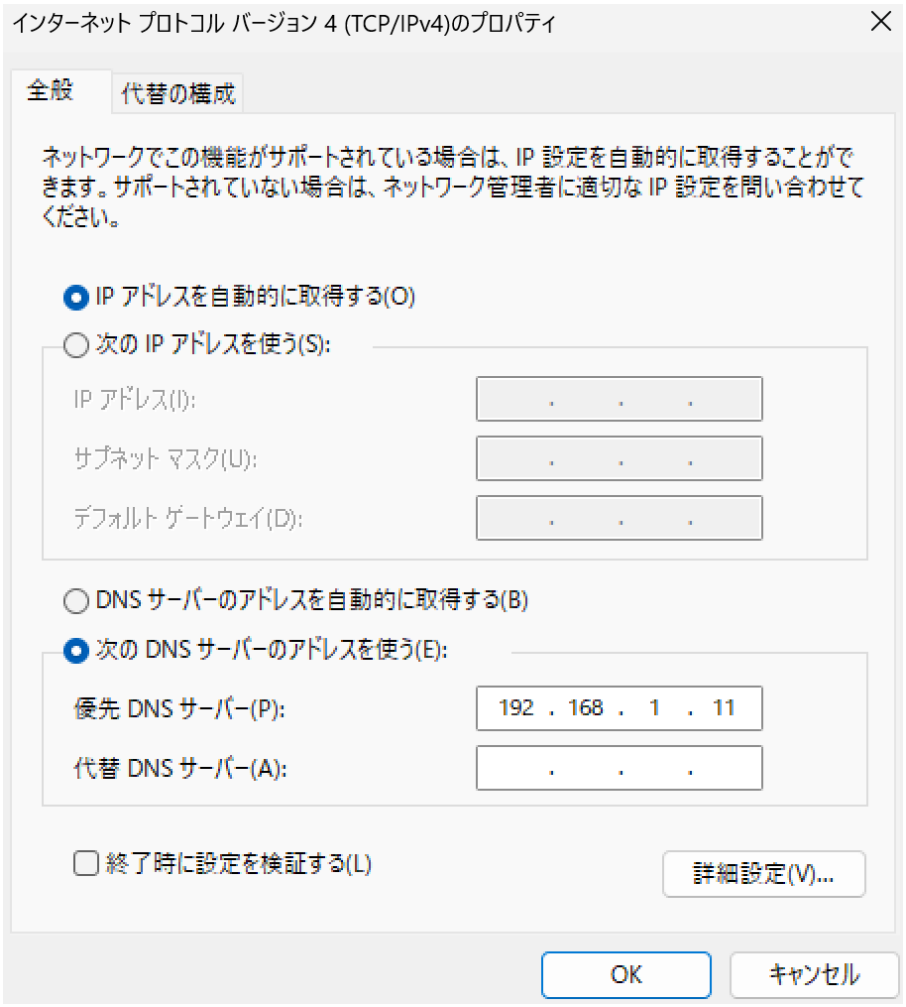

図 45: ホスト PC の DNS 設定変更

6.16.2　Thunderbird の基本設定

Thunderbird のアプリケーションウインドウを表示し、「既存のメールアドレスのセットアップ」タブが表示されていることを確認します。もしくは、「メールアカウント操作」の中にある「メールアカウントを追加」を選択します。

各設定項目の値を以下のように入力します。

設定項目	設定値
あなたの名前	user1
メールアドレス	user1@example1.test
パスワード	アカウント作成時に指定したもの
パスワードを記憶する	チェックしておく

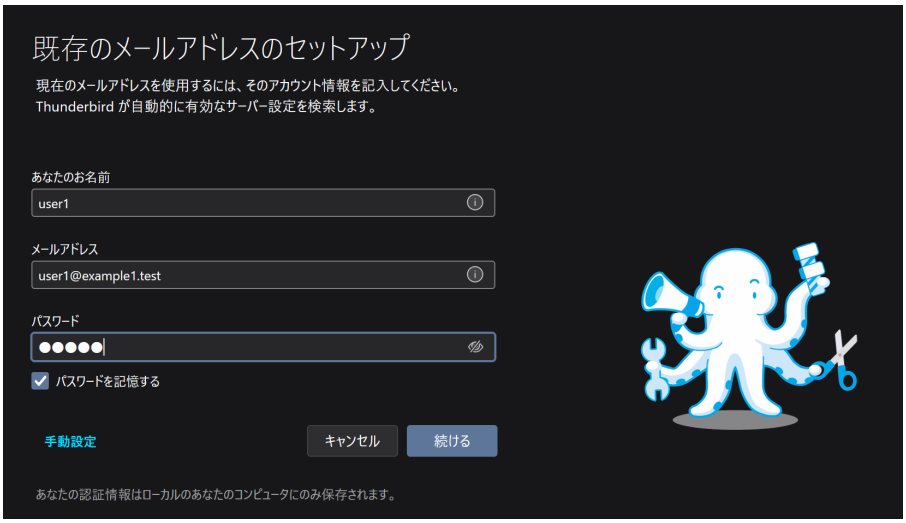

図 46: メールアドレスとパスワードの設定

6.16.3 Thunderbird の送受信メールサーバーの設定

「手動設定」をクリックして、Thunderbird の送受信メールサーバーの設定を行います。

「受信サーバー」と「送信サーバー」が表示されるので、各設定項目を以下のように入力します。

設定項目	設定値
プロトコル	IMAP
ホスト名	mail.example1.test
ポート番号	143
接続の保護	なし
認証方式	通常のパスワード認証
ユーザ名	user1

設定項目	設定値
ホスト名	mail.example1.test
ポート番号	25
接続の保護	なし
認証方式	通常のパスワード認証
ユーザ名	user1

入力を終えたら「完了」ボタンをクリックします。

図 47: 受信サーバーと送信サーバーの設定

「警告！」ダイアログが表示されますが、左下の「接続する上での危険性を理解しました」をチェックし、「確認」ボタンをクリックします。

図 48: 警告ダイアログ

最後に「リンクしたサービスへの接続」の「完了」ボタンをクリックします。正常に完了出来た場合、

図 49: 正常完了ダイアログ 1

図 50: 正常完了ダイアログ 2

6.17 メールの送信

メールを送信するには、「作成」ボタンをクリックしてメール作成ウインドウを呼び出します。

6.17.1 自分宛のメール送信

1. 「作成」ボタンをクリック
2. 宛先に自分のメールアドレス（user1@example1.test）を指定して、メールを作成、送信してみます。
3. 「受信」ボタンをクリックして、メールが受信できることを確認します。

図 51: テストメール 1

6.17.2 別サーバー宛のメール送信

1. 「作成」ボタンをクリック
2. 宛先に他の受講生のメールアドレス（user2@example2.test）を指定して、メールを作成、送信してみます。
3. host2 で mail コマンドを使ってメールを受信できたことを確認します。
4. mail コマンドで user1@example1.test 宛にメールを送信し、host1 で受信できることを確認します。

```
user2@host2example2test:~$ mail
"/var/mail/user2": 1 message 1 new
>N   1 user1              Tue May  6 01:05  22/890   テストメール from u
? 1
Return-Path: <user1@example1.test>
X-Original-To: user2@example2.test
Delivered-To: user2@example2.test
Received: from mail.example1.test (unknown [192.168.1.12])
        by mail.example2.test (Postfix) with ESMTPS id 9B1E4220B3
        for <user2@example2.test>; Tue,  6 May 2025 01:05:40 +0000 (UTC)
Received: from [192.168.1.122] (unknown [192.168.1.122])
        by mail.example1.test (Postfix) with ESMTP id 3DF257A3D
        for <user2@example2.test>; Tue,  6 May 2025 01:05:39 +0000 (UTC)
Message-ID: <15e595f5-6ea5-4b67-9242-e69458cce8fb@example1.test>
Date: Tue, 6 May 2025 10:05:37 +0900
MIME-Version: 1.0
User-Agent: Mozilla Thunderbird
Content-Language: en-US
To: user2@example2.test
From: user1 <user1@example1.test>
Subject: テストメール from user1
Content-Type: text/plain; charset=UTF-8; format=flowed
Content-Transfer-Encoding: 8bit

テストメール(from user1)です

? q
```

6.18 うまく動作しない場合には

本章では、電子メールに関する学習を行いました。また、実際にメールサーバーを設定し、mail コマンドや Thunderbird を利用してメールの送受信の確認を行いました。

メールサーバーの設定は、メールサーバーが正しく設定され起動していたとしても、DNS サーバーが正しく動いていなければ利用できない場合があります。設定ファイルの記述に問題がないのに、メールがどうしても送れない、受信できない場合は、まず DNS が正しく動いているか dig コマンドを実行して確認します。また、ログ（/var/log/maillog）を見て、エラーが出ていないかを確認してみてください。

7　ネットワークとセキュリティの設定

第 7 章では、ネットワークとセキュリティの設定についてあらためて確認します。Ubuntu では、基本的なネットワークやセキュリティの設定はインストール時に行われます。ここでは、これらを OS インストール後に設定する方法について説明します。

7.1　用語集

ネットワークインターフェース

LAN ケーブルを接続して、外部のマシンとの間でデータをやり取りするための物理的なインターフェースです。

ループバックインターフェース

マシン内部でデータをやり取りするための仮想的なインターフェースです。

IP（Internet Protocol）

IP は、ネットワークに接続したコンピュータ間でデータをやり取りするためのプロトコルです。

IP アドレス

IP アドレスは、IP 通信で各コンピュータに割り当てられる値です。データの送り先として IP アドレスを指定すると、その IP アドレスが割り当てられたコンピュータに送信されたデータが届きます。

IPv4（Internet Protocol version 4）

現在のインターネットで利用されている通信プロトコルです。IPv4 では、IP アドレスを 4 バイト（32 ビット）で表します。本来は 32 個の 2 進数（0 と 1）の羅列ですが、人間に分かりやすくするために 1 バイトごとに 10 進数に変換して.（ドット）で区切って「192.168.1.1」の様に表記します。次世代の IP である IPv6 では、IP アドレスを 128 ビットで表します。

ネットワークアドレス

ホストが属しているネットワーク自体を指し示す IP アドレスです。

ブロードキャストアドレス

ホストが属しているネットワーク全体を指し示す IP アドレスです。このアドレスに対して通信を行うことで、ネットワークに属しているホストすべてに対して通信が行えます。

ネットマスク

IP アドレスのうち、どこまでがネットワーク部で、どこまでがホスト部かを示すための値です。IP アドレスとネットマスクの 2 つの値から、ネットワークアドレス、ブロードキャストアドレスを割り出すことができます。

デフォルトゲートウェイ

インターネットは、小さなネットワークが相互に接続したネットワークです。小さなネットワーク間を接続する機器としてルーター（ゲートウェイ）が使われます。ゲートウェイは 1 つのネットワークに複数設置することができますが、特に指定が無い場合にはデフォルトゲートウェイを使って外部のネットワークとの通信を行います。

DHCP（Dynamic Host Configuration Protocol）

IP アドレスなどのネットワーク設定を自動的に割り当てるプロトコルです。

TCP（Transmission Control Protocol）

TCP は、コネクション方式で通信するプロトコルです。IP と組み合わせた TCP/IP がインターネットの標準的な通信プロトコルです。TCP の特長として、届かなかった通信パケットを再送信して確実に通信を行う仕組みがあります。

UDP（User Datagram Protocol）

UDP は、コネクションレス方式で通信するプロトコルです。TCP とは異なり、データの再送信が行われないので通信の確実性は劣りますが、セッションを確立するための「3 ウェイハンドシェイク」の手間が不要なためシンプルな通信に適しています。たとえば、大量に問い合わせが行われる DNS への名前解決の問い合わせは UDP となっています。

ポート番号

ポート番号は、TCP と UDP が通信する際に使用する値です。たとえば Web サーバーはポート番号 80 番を使用して動作しているので、Web ブラウザは目的の Web サーバーのポート番号 80 番に接続します。ポート番号は 0 番から 65535 番まで使用できますが、0〜1023 番は WELL KNOWN PORT、1024〜49151 番は REGISTERED PORT として予約されています。

ICMP（Internet Control Message Protocol）

データの転送エラーやデータ転送量などの情報を通知するためのプロトコルです。

ping コマンド

ping コマンドは、ICMP を使って宛先に指定したホストに到達することができるかどうかを確認するコマンドです。

7.2　ネットワーク管理

ネットワークが上手く使えない場合には、確認のためにネットワークインターフェースが正しく設定されているかを調べる必要があります。また、設定が間違っている場合には、インターフェースの設定を変更する必要があります。ここでは、ネットワークインターフェースの確認と設定の方法について解説します。

7.2.1　ネットワークインターフェースの確認

Linux をインストールしたマシンが正常にネットワークに接続できるかどうか、設定を確認します。ip コマンドで確認します。

```
ubuntu@host1example1test:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
    default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
    group default qlen 1000
    link/ether 08:00:27:ee:b7:1d brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.12/24 brd 192.168.1.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 240f:32:57b8:1:a00:27ff:feee:b71d/64 scope global dynamic mngtmpaddr
        noprefixroute
       valid_lft 285sec preferred_lft 285sec
    inet6 fe80::a00:27ff:feee:b71d/64 scope link
       valid_lft forever preferred_lft forever
```

ip コマンドで表示された lo は仮想的なループバックインターフェースです。また、この例では enp0s3 に「192.168.1.12」が割り当てられていることが分かります。なお、この名称は、enoXX、ensXX、ethX、enxXX などの名称になる場合もあります。

7.2.2 ネットワークインターフェースの再設定

/etc/netplan/50-cloud-init.yaml を編集します。

```
ubuntu@host1example1test:~$ sudo cat /etc/netplan/50-cloud-init.yaml
[sudo] password for ubuntu:
network:
  version: 2
  ethernets:
    enp0s3:
      addresses:
      - "192.168.1.12/24"
      nameservers:
        addresses:
        - 192.168.1.11
        search: [example1.test]
      routes:
      - to: "default"
        via: "192.168.1.1"
```

設定を適用します。

```
ubuntu@host1example1test:~$ sudo netplan apply
```

7.2.3 ネットワークインターフェースの動作確認

ネットワークインターフェースが動作しているかは ping コマンドで確認します。ping コマンドで確認する IP アドレスとして自分の物理ネットワークインターフェースの IP アドレス、もう一台のサーバの IP アドレス（192.168.1.13）やその他のマシンの IP アドレスなどを指定します。ping コマンドは Ctrl+C で中止できます。

```
ubuntu@host1example1test:~$ ping 192.168.1.13
PING 192.168.1.13 (192.168.1.13) 56(84) bytes of data.
64 bytes from 192.168.1.13: icmp_seq=1 ttl=64 time=5.96 ms
64 bytes from 192.168.1.13: icmp_seq=2 ttl=64 time=2.98 ms
64 bytes from 192.168.1.13: icmp_seq=3 ttl=64 time=1.17 ms
64 bytes from 192.168.1.13: icmp_seq=4 ttl=64 time=3.15 ms
^C
--- 192.168.1.13 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 1.166/3.314/5.959/1.713 ms
```

7.3 サービスのポート番号を確認

どんなネットワークサービスが自分の PC で動いているかを、ss コマンドで確認します。ss -at コマンドを実行すると、現在の TCP 通信の状態をすべて表示します。

7.3.1 ss コマンドを使ったポート使用状況の確認

ss コマンドは-a オプションでサービスの状態をすべて表示、-t オプションで TCP（Transmission Control Protocol）のサービスが使うポートなどの情報のみを表示します。

```
ubuntu@host1example1test:~$ ss -ta
State       Recv-Q    Send-Q                                  Local Address:Port
                                                 Peer Address:Port          Process
```

```
LISTEN      0         100                                          127.0.0.1:smtp
                                                       0.0.0.0:*
LISTEN      0         10                                           127.0.0.1:domain
                                                     0.0.0.0:*
LISTEN      0         5                                            127.0.0.1:953
                                                        0.0.0.0:*
LISTEN      0         4096                                   127.0.0.53%lo:domain
                                                     0.0.0.0:*
LISTEN      0         100                                     192.168.1.12:smtp
                                                       0.0.0.0:*
LISTEN      0         100                                          0.0.0.0:imap2
                                                      0.0.0.0:*
LISTEN      0         10                                      192.168.1.12:domain
                                                     0.0.0.0:*
LISTEN      0         4096                                    127.0.0.54:domain
                                                     0.0.0.0:*
LISTEN      0         511                                          0.0.0.0:http
                                                       0.0.0.0:*
LISTEN      0         100                                          0.0.0.0:pop3
                                                       0.0.0.0:*
LISTEN      0         10                                             [::1]:domain
                                                        [::]:*
LISTEN      0         100                                             [::]:imap2
                                                         [::]:*
LISTEN      0         5                                              [::1]:953
                                                           [::]:*
LISTEN      0         4096                                              *:ssh
                                                              *:*
LISTEN      0         511                                             [::]:http
                                                          [::]:*
LISTEN      0         100                                             [::]:pop3
                                                          [::]:*
ESTAB       0         0                                [::ffff:192.168.1.12]:ssh
                                    [::ffff:192.168.1.122]:56119
ESTAB       0         52                               [::ffff:192.168.1.12]:ssh
                                    [::ffff:192.168.1.122]:50112
SYN-SENT    0         1                  [240f:32:57b8:1:a00:27ff:feee:b71d]:33848
                 [2a05:d018:91c:3200:2846:99fb:81b6:1e11]:https
```

7.3.2 services ファイルによるポート番号の確認

ポート番号とサービスの対応（WELL KNOWN PORT NUMBERS:0～1023 や REGISTERED PORT NUMBERS:1024～49151）が定義されている/etc/services ファイルも確認してみてください。

```
ubuntu@host1example1test:~$ cat /etc/services
（略）
ssh             22/tcp                          # SSH Remote Login Protocol
telnet          23/tcp
smtp            25/tcp          mail
time            37/tcp          timserver
time            37/udp          timserver
whois           43/tcp          nicname
tacacs          49/tcp                          # Login Host Protocol (TACACS)
tacacs          49/udp
domain          53/tcp                          # Domain Name Server
domain          53/udp
bootps          67/udp
bootpc          68/udp
tftp            69/udp
gopher          70/tcp                          # Internet Gopher
```

```
finger          79/tcp
http            80/tcp          www             # WorldWideWeb HTTP
kerberos        88/tcp          kerberos5 krb5 kerberos-sec     # Kerberos v5
kerberos        88/udp          kerberos5 krb5 kerberos-sec     # Kerberos v5
iso-tsap        102/tcp         tsap            # part of ISODE
acr-nema        104/tcp         dicom           # Digital Imag. & Comm. 300
pop3            110/tcp         pop-3           # POP version 3
sunrpc          111/tcp         portmapper      # RPC 4.0 portmapper
sunrpc          111/udp         portmapper
auth            113/tcp         authentication tap ident
nntp            119/tcp         readnews untp   # USENET News Transfer Protocol
ntp             123/udp                         # Network Time Protocol
epmap           135/tcp         loc-srv         # DCE endpoint resolution
netbios-ns      137/udp                         # NETBIOS Name Service
netbios-dgm     138/udp                         # NETBIOS Datagram Service
netbios-ssn     139/tcp                         # NETBIOS session service
imap2           143/tcp         imap            # Interim Mail Access P 2 and 4
（略）
```

7.4 SSH によるリモートログイン

SSH はネットワーク経由でリモートにある Linux サーバーにログインするために使用するプロトコルです。通信が暗号化されているため、覗き見されてもパスワードや作業内容が分からない他、公開鍵を使った認証を行うことでパスワードをネットワークに流すことなくログインすることができます。

Linux では、OpenSSH のサーバーとクライアントが用意されています。

7.4.1 パスワードによる認証

ssh コマンドは特別な設定を行わなくても、パスワード認証でリモートログインすることができます。

以下のようにして、自分自身に SSH で接続してみます。初めての接続の場合には、SSH サーバーの公開鍵が送られてきて接続してもよいか確認されるので「yes」と入力します。パスワード認証が可能だと、パスワードの入力が要求されます。

```
ubuntu@host1example1test:~$ ssh user1@localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ED25519 key fingerprint is SHA256:c68i9zv/K4gfef+MMMxm0e6WMJ09ufi9dQwA8vt8n4A.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ED25519) to the list of known hosts.
user1@localhost's password:
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-58-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Tue May  6 01:17:31 AM UTC 2025

  System load:             0.0
  Usage of /:              46.8% of 9.75GB
  Memory usage:            13%
  Swap usage:              0%
  Processes:               124
  Users logged in:         2
  IPv4 address for enp0s3: 192.168.1.12
  IPv6 address for enp0s3: 240f:32:57b8:1:a00:27ff:feee:b71d

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.
```

```
   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

75 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


You have mail.
Last login: Tue May  6 00:02:42 2025 from 192.168.1.122

user1@host1example1test:~$ exit ← リモートログインを終了
logout
Connection to localhost closed.
ubuntu@host1example1test:~$ ← 元のユーザーに復帰
```

7.4.2 公開鍵による認証

パスワード認証は、通信経路が SSH で暗号化されているといっても、パスワードがネットワークを流れていること、またパスワードを自動的に生成して順番に試していく「総当たり攻撃」を受けた場合不正にログインされてしまう可能性があるので、インターネット上に公開されているサーバーで使用するには相応しくありません。

公開鍵認証は、あらかじめサーバーに設置した公開鍵と対になっている秘密鍵を持っているユーザーしかリモートログインできない認証方法です。

以下の手順で公開鍵認証を設定します。

1. 公開鍵と秘密鍵を生成する ssh-keygen コマンドを使用して一対の公開鍵（id_ed25519.pub）と秘密鍵（id_ed25519）を生成します。鍵のファイルはホームディレクトリに作られた.ssh ディレクトリに保存されます。

秘密鍵には不正利用を防止するためのパスフレーズを設定します。接続時にパスフレーズを正しく入力できないと、秘密鍵は利用できないので、公開鍵認証による接続はできません。このパスフレーズは SSH クライアント側で秘密鍵に対して処理されるので、ネットワーク上には情報は流れません。

```
ubuntu@host1example1test:~$ su - user1 ← user1ユーザーに切り替え
Password:
user1@host1example1test:~$

user1@host1example1test:~$ ssh-keygen ←
    鍵形式を省略したのでED25519形式で鍵を生成
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/user1/.ssh/id_ed25519):
Created directory '/home/user1/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user1/.ssh/id_ed25519
Your public key has been saved in /home/user1/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:r05cuOLisPskAOeApHjAco7++UQ5sVmtYXS7iWqFzW4 user1@host1example1test
The key's randomart image is:
+--[ED25519 256]--+
|o.     . .       |
|*o.   . o .      |
|B+o  . + o       |
|o=.   @ +.o      |
|o .  B *So.      |
| o  . =. +       |
```

```
|  + o+.E+ .      |
|   B+..o .       |
|  o++o..o        |
+----[SHA256]-----+
```

1. 接続先に authorized_keys を作成するユーザーに公開鍵認証による SSH での接続を許可するには、ユーザーアカウントを作成し、そのユーザーのホームディレクトリに.ssh/authorized_keys ファイルを作成しておきます。.ssh ディレクトリのパーミッションは 700（drwx——）、authorized_keys ファイルのパーミッションは 600（-rwx——）に設定する必要があります。

```
user1@host1example1test:~$ ls -ld .ssh
drwx------ 2 user1 user1 4096 May  6 01:24 .ssh

user1@host1example1test:~$ cd .ssh

user1@host1example1test:~/.ssh$ touch authorized_keys

user1@host1example1test:~/.ssh$ chmod 600 authorized_keys

user1@host1example1test:~/.ssh$ cat id_ed25519.pub >> authorized_keys

user1@host1example1test:~/.ssh$ ls -l authorized_keys
-rw------- 1 user1 user1 105 May  6 01:27 authorized_keys
```

1. 公開鍵認証で接続する公開鍵認証で接続します。ssh コマンドの使用法自体はパスワード認証と同じですが、パスワードの代わりに秘密鍵に設定したパスフレーズの入力が必要です。

```
ubuntu@host1example1test:~$ ssh user1@192.168.1.12
The authenticity of host '192.168.1.12 (192.168.1.12)' can't be established.
ED25519 key fingerprint is SHA256:c68i9zv/K4gfef+MMMxm0e6WMJ09ufi9dQwA8vt8n4A.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.12' (ED25519) to the list of known hosts.
user1@192.168.1.12's password:
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-58-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Tue May  6 01:42:03 AM UTC 2025

  System load:             0.03
  Usage of /:              46.8% of 9.75GB
  Memory usage:            13%
  Swap usage:              0%
  Processes:               124
  Users logged in:         2
  IPv4 address for enp0s3: 192.168.1.12
  IPv6 address for enp0s3: 240f:32:57b8:1:a00:27ff:feee:b71d

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

75 updates can be applied immediately.
```

```
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


You have mail.
Last login: Tue May  6 01:35:50 2025 from 127.0.0.1
user1@host1example1test:~$
```

7.4.3 パスワード認証の禁止

パスワード認証が有効になっていると、パスワードの総当たり攻撃により不正にリモートログインできてしまいます。公開鍵認証で接続できるようになった後には、SSH サーバーの設定を変更してパスワード認証を禁止しておきます。

1. ubuntu ユーザーでパスワード認証で接続できることを確認する user1 ユーザーになっている場合には exit して ubuntu ユーザーに戻ります。

```
user1@host1example1test:~$ exit
logout
ubuntu@host1example1test:~$ ssh ubuntu@localhost
ubuntu@localhost's password:
ubuntu@host1example1test:~$ exit
logout
Connection to localhost closed.
```

1. 設定ファイル/etc/ssh/sshd_config.d/50-cloud-init.conf を修正する

```
ubuntu@host1example1test:~$ sudo cat /etc/ssh/sshd_config.d/50-cloud-init.conf
```

```
（略）
PasswordAuthentication no ← noに設定した行を追加
#PasswordAuthentication yes
（略）
```

1. 設定を変更後、ssh 設定を再読み込みする

```
ubuntu@host1example1test:~$ sudo systemctl reload ssh
```

1. 公開鍵認証を設定していないユーザーで SSH サーバーに接続して、接続できないことを確認する

```
ubuntu@host2example2test:~$ ssh ubuntu@192.168.1.12
ubuntu@192.168.1.12: Permission denied (publickey).
```

1. 公開鍵認証を設定済みのユーザーで SSH サーバーに接続して、接続できることを確認する

```
user1@host1example1test:~$ ssh user1@localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ED25519 key fingerprint is SHA256:c68i9zv/K4gfef+MMMxm0e6WMJ09ufi9dQwA8vt8n4A.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ED25519) to the list of known hosts.
Enter passphrase for key '/home/user1/.ssh/id_ed25519':
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-58-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro
```

```
 System information as of Tue May  6 01:54:51 AM UTC 2025

  System load:             0.2
  Usage of /:              46.8% of 9.75GB
  Memory usage:            13%
  Swap usage:              0%
  Processes:               125
  Users logged in:         2
  IPv4 address for enp0s3: 192.168.1.12
  IPv6 address for enp0s3: 240f:32:57b8:1:a00:27ff:feee:b71d

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

75 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


You have mail.
Last login: Tue May  6 01:42:05 2025 from 192.168.1.12
user1@host1example1test:~$
```

この設定を行った後はパスワード認証でリモートログインできなくなるので、かならず事前に管理権限を持ったユーザーが公開鍵認証でリモートログインできるようにしておく必要があります。

7.5 ファイアウォールの設定

ファイアウォールはネットワークにおいて様々なアクセス制限を行い、ネットワークからの攻撃や不正なアクセス等を防ぐ機能です。

Ubuntu のファイアウォール機能は ufw によって管理されています。ufw では、ネットワークインターフェースへのパケットの受信の許可、拒否のルールを管理しています。ufw の設定は、ufw コマンドで行います。

7.5.1 ファイアウォール設定の確認

まず、ufw の状態を確認すると、ufw が無効となっていることが分かります。

```
ubuntu@host1example1test:~$ sudo ufw status
sudo: unable to resolve host host1example1test: Name or service not known
Status: inactive
```

ufw を有効化する前に、まず ssh 通信が出来るようにしておきます。

```
ubuntu@host1example1test:~$ sudo ufw allow 22/tcp
Rules updated
Rules updated (v6)
```

ufw を有効化し、状態を確認します。

```
ubuntu@host1example1test:~$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```

```
ubuntu@host1example1test:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
```

7.5.2 許可サービスの追加

サービスの許可を追加します。以下の例では、今回の演習で構築した各サービスを許可しています。

```
ubuntu@host1example1test:~$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)

ubuntu@host1example1test:~$ sudo ufw allow 25/tcp
Rule added
Rule added (v6)

ubuntu@host1example1test:~$ sudo ufw allow 110/tcp
Rule added
Rule added (v6)

ubuntu@host1example1test:~$ sudo ufw allow 143/tcp
Rule added
Rule added (v6)

ubuntu@host1example1test:~$ sudo ufw allow 53/tcp
Rule added
Rule added (v6)

ubuntu@host1example1test:~$ sudo ufw allow 53/udp
Rule added
Rule added (v6)

ubuntu@host1example1test:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
80/tcp                     ALLOW       Anywhere
25/tcp                     ALLOW       Anywhere
110/tcp                    ALLOW       Anywhere
143/tcp                    ALLOW       Anywhere
53/tcp                     ALLOW       Anywhere
53/udp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
80/tcp (v6)                ALLOW       Anywhere (v6)
25/tcp (v6)                ALLOW       Anywhere (v6)
110/tcp (v6)               ALLOW       Anywhere (v6)
143/tcp (v6)               ALLOW       Anywhere (v6)
53/tcp (v6)                ALLOW       Anywhere (v6)
53/udp (v6)                ALLOW       Anywhere (v6)
```

この設定は即座に有効になります。ufw が OS 起動時に自動起動されるか確認しておきましょう。

```
ubuntu@host1example1test:~$ sudo systemctl is-enabled ufw
```

```
enabled
```

7.5.3 許可サービスの取り消し

許可されているサービスを取り消しすることもできます。

```
ubuntu@host1example1test:~$ sudo ufw deny 53/tcp
Rule updated
Rule updated (v6)

ubuntu@host1example1test:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
80/tcp                     ALLOW       Anywhere
25/tcp                     ALLOW       Anywhere
110/tcp                    ALLOW       Anywhere
143/tcp                    ALLOW       Anywhere
53/tcp                     DENY        Anywhere
53/udp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
80/tcp (v6)                ALLOW       Anywhere (v6)
25/tcp (v6)                ALLOW       Anywhere (v6)
110/tcp (v6)               ALLOW       Anywhere (v6)
143/tcp (v6)               ALLOW       Anywhere (v6)
53/tcp (v6)                DENY        Anywhere (v6)
53/udp (v6)                ALLOW       Anywhere (v6)
```

Linux サーバー構築標準教科書 (Ubuntu 版)

2026 年 7 月 1 日　Ver.1.0.0

発行元　LPI-Japan