目次

9. ネットワーク入門.....5
予習編.....6
9.1 ネットワークの仕組み.....10
9.1.1 インターネット上のサービス.....10
9.1.2 クライアント・サーバモデル.....11
9.1.3 コンピュータ間の通信.....11
9.2 IP アドレス.....13
9.2.1 2 進数と IP アドレス.....13
9.2.2 ネットワークセグメント.....14
9.2.3 IP アドレスのクラス.....16
9.2.4 グローバルアドレスとプライベートアドレス.....17
9.2.5 サブネット.....18
9.2.6 異なるネットワークセグメント間での通信.....19
9.2.7 名前解決.....20
9.3 Linux におけるネットワーク設定.....21
9.3.1 基本的なコマンド.....21
9.3.2 IP アドレスの設定.....22
9.4 ネットワーク診断ツール.....24
9.4.1 ping.....24
9.4.2 traceroute.....26
9.4.3 nslookup.....27
9.5 確認問題.....28
column.....29
10. Linux のインストール.....33
予習編.....34
10.1 ハードディスクのパーティション.....36
10.1.1 ハードディスク.....36
10.1.2 マスターブートレコード.....37
10.1.3 パーティションとファイルシステム...38
10.2 CentOS のインストール準備.....40
10.2.1 メディアからの起動.....40
10.2.2 言語の選択.....41
10.3 インストールの詳細設定.....42
10.3.1 地域設定.....42
10.3.2 ソフトウエア.....43
10.3.3 システム.....44
10.4 CentOS のインストール.....49
10.4.1 root パスワード.....50
10.4.2 ユーザの作成.....50
10.4.3 完了.....51
10.5 インストール後処理.....52
10.5.1 初期セットアップ.....52
10.5.2 ログイン.....54
10.6 システムの起動.....58
10.6.1 ターゲットとランレベル.....58
10.6.2 ターゲットの切り替え.....59
10.6.3 自動起動設定.....61
11. ApacheHTTP サーバ(1).....63
予習編.....64
11.1 Web の仕組みと Web サーバ.....66
11.1.1 Web の仕組み.....66
11.1.2 デーモンプログラム.....67
11.1.3 サーバと TCP ポート.....68
11.1.4 Web サーバ.....68
11.1.5 HTTP.....69
11.2 Apache の起動と基本的な利用.....71
11.2.1 Apache のインストール.....71
11.2.2 Apache の起動.....71
11.2.3 Apache の実行プロセス.....72
11.2.4 ドキュメントの配置.....73
column.....74
12. ApacheHTTP サーバ(2).....77
予習編.....78
12.1 Apache の設定.....79
12.1.1 Apache 設定ファイル.....79
12.1.2 Apache の基本設定.....80
12.1.3 <Directory>タグ.....81
12.2 基本認証.....83
12.2.1 基本認証の概要.....83
12.2.2 基本認証の設定.....83
12.2.3 パスワードファイルの作成.....84
column.....85
13. ApacheHTTP サーバ(3).....91
予習編.....92
13.1 Apache の機能拡張.....93
13.1.1 モジュールとは.....93
13.1.2 Apache のモジュール.....94
13.1.3 Apache モジュールの追加・削除の設定.....95
13.2 CGI.....96
13.2.1 CGI の仕組み.....96
13.2.2 CGI の利用設定.....97
13.3 確認問題.....99
column.....100
14. BIND DNS サーバ(1).....103
予習編.....104
14.1 インターネット上の名前解決.....106
14.1.1 FQDN とドメイン.....106
14.1.2 DNS サーバの役割.....107
14.1.3 リゾルバと DNS サーバ.....107
14.1.4 名前解決とレコード.....108
14.2 名前解決ツール.....109
14.2.1 nslookup コマンド.....109
14.2.2 dig コマンド.....111
14.3 ネームサーバの構成.....113
14.3.1 ネームサーバの機能.....113
14.3.2 ゾーンファイルとゾーン転送.....114
14.4 確認問題.....115
15. BIND DNS サーバ(2).....117
予習編.....118
15.1 ゾーンの定義.....119
15.1.1 ゾーンの委譲とネームサーバの設置.....119
15.1.2 named.conf.....120
15.1.3 ローカルゾーンファイルの準備....121
15.1.4 ゾーンの定義.....122
15.2 正引きの設定.....124
15.2.1 正引きゾーンファイルの作成.....124
15.2.2 BIND の起動.....127
15.2.3 ゾーンの省略.....128
column.....129
付録 確認問題 解答例.....135
第 9 章.....136
第 10 章.....142
第 11 章.....144
第 12 章.....145
第 13 章.....147
第 14 章.....148
補足.....153

9

ネットワーク入門

本章のねらい

- ネットワークの仕組みを学ぶ
- IP アドレスについて学ぶ
- ネットワークセグメントとネットマスクについて学ぶ
- Linux におけるネットワークの設定について学ぶ
- ネットワーク診断ツールについて学ぶ

予習編

Linux は強力なネットワーク機能を持つ OS であり、ネットワークサーバとして必要な機能・ソフトウェアが充実しています。本章では、「インターネット」や「LAN」などネットワークの知識を学びます。

コンピュータネットワーク

普通、パソコンというと、自分のマシンにワープロソフトや音楽再生ソフトなどを導入して、単独で利用するものを連想する方が多いでしょう。しかし、現在ではインターネットに代表されるように、Web ブラウジングによる情報収集や電子メールによる他人との情報交換など、複数のコンピュータ間で互いに通信し合うような利用法が多く用いられています。

インターネットでは、どのような仕組みによって他のコンピュータと通信をするのでしょうか？どのようにして、世界中の Web サイトを見たり、世界中の人と電子メールをやりとりするのでしょうか？

正解を簡単に述べますと、世界中のマシンを接続する網の目のように張り巡らされたデータの通信路がそれらを可能にしています。この通信路で結ばれ、互いに通信しあう世界中のコンピュータの総体を「インターネット」と呼びます。ケーブルや無線などで構成されるデータの通信路がインターネットの根幹であり、この通信路に接続することによって、自分のコンピュータもインターネットの一員となることができるのです。

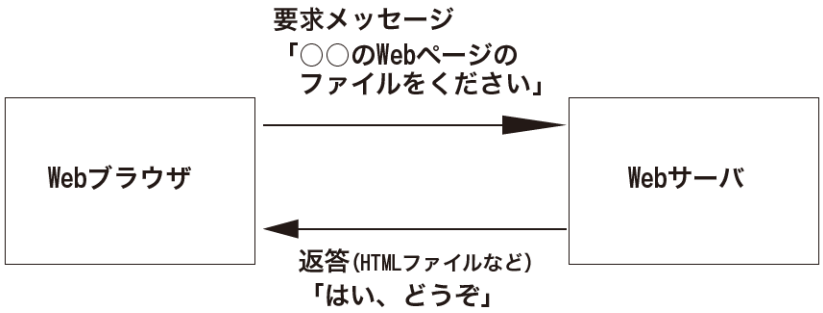
ここまででは「インターネット（The Internet）」という単語を使用しましたが、コンピュータ同士が通信しあうのは、インターネット上においてのみというわけではありません。例えば、オフィスや自宅に置いてある複数台のコンピュータの間で、ファイルやプリンタを共有することもできます。このように、互いにデータをやりとりできる複数のコンピュータの集まりのことを、一般に「コンピュータネットワーク」または、単に「ネットワーク」と呼びます。「インターネット」は、世界規模の巨大なコンピュータネットワークです。

あるコンピュータがネットワークの一員として、他のコンピュータとデータをやりとりするためには、単にケーブルをつなげば良いというものではありません。ハードウェアだけでなく、ソフトウェア的にもネットワークへの接続機能が備わっていなければ、信頼できる通信はできません。ソフトウェアの中でも、とりわけ OS にネットワーク機能への対応がなされていることが非常に重要です。近年の Windows や Mac OS、そして Linux などの OS は全てネットワーク機能を備えています。

サーバとクライアント

Web ページは、Microsoft Internet Explorer や Mozilla などの Web ブラウザと呼ばれるソフトを用いて見ることができます。文字や画像などから成る Web ページは、実際には HTML ファイルと呼ばれる種類のファイルをインターネット上から自分のパソコンにダウンロードし、このデータを Web ブラウザ側が解釈することによって表示されています。ファイルをダウンロードするということは、ダウンロードされるファイルを提供する側も存在するはずです。この役割を担っているのが Web サーバと呼ばれるコンピュータです。

Web ページ閲覧のイメージ図



Web サーバ

Web ページの公開に限らず、インターネット上では電子メールの配送やソフトウェアのダウンロードなど、様々なサービスが実現されています。一般にこれらのサービスは、「サービスを提供する側」と「サービスを受ける側」に分かれています。前者の方を「サーバ（server）」、後者を「クライアント（client）」と呼びます。

Web ページの公開の例では、Web サーバが「サーバ」、Web ブラウザが「クライアント」の役割を担っています。今後の講義の中では、Web サーバだけでなく様々なサービスを提供するサーバの構築・管理について解説していきます。

IP アドレス

IP アドレスとは、ネットワーク上にあるコンピュータを識別するためのもので、「電話番号」のようなものです。前述の Web サーバなどに通信するには、相手の IP アドレスを指定しなければなりません。

IP アドレスはピリオドで区切られた、0〜 255 の 4 組の整数から成り立っています。

```
192.168.255.1
```

コンピュータ内部では IP アドレスは、次の様な 2 進数の形で解釈されています。

```
192 . 168 . 255 . 1
    ↓
1000000 10101000 11111111 00000001
```

コンピュータの内部では、全てのデータは 0 と 1 の数字の列で表されます。数字を 0〜9 までの 10 種類の数字で表す方法を 10 進数と呼び、0 と 1 の 2 種類で表す方法を 2 進数と呼びます。コンピュータは、文字や数値、画像など全てのデータは 2 進数で表されます。IP アドレスも同様に 32 桁の 2 進数で表されていますが、通常は人が読みやすいように、4 組の 10 進数で表現します。

インターネットに限らず、オフィスのような小規模のコンピュータネットワークにでも、、互いを識別するために IP アドレスが用いられます。ネットワークの管理を行うためには、各ネットワーク機器に IP アドレスを割り振る必要があります。IP アドレスは、重複してはいけない、予約済みアドレスなど、割り振り方にはいくつかのルールが存在します。この章では、ネットワークの基本について解説していきます。

ネットワークセグメント

インターネットなどの巨大なネットワークは、小さなネットワーク同士を、互いに接続することで大きな規模になってきました。この小さなネットワークの単位を「ネットワークセグメント」と呼びます。基本的に、ネットワークセグメントが異なるコンピューターは直接通信できません。ネットワークセグメント間の通信を行なうためには、「ゲートウェイ」と呼ばれる、橋渡しする装置が必要になります。

IP アドレスは、ネットワークセグメントを表す「ネットワーク部」と、各々の機器を表す「ホスト部」から成り立っています。ネットワーク部は電話番号の「局番」に相当し、ホスト部はそれ以下の部分に相当する、と考えるとよいでしょう。ネットワーク部とホスト部の境界を決める値は「ネットマスク」と言います。

Linux におけるネットワーク設定

IP アドレスは、個々のマシンを区別するための物であることを紹介しました。しかし、コンピュータは、あらかじめ決まった IP アドレスを持っているわけではありません。「自分の IP アドレスは〇.〇.〇.〇である」と設定する必要があります。

Windows であれば、「コントロールパネル」などから呼び出されるプログラムのテキストボックスに IP アドレスを入力して設定を行います。Linux では設定ファイルと総称されるテキストファイルに、書式に従って設定を記述していきます。vi の操作にはもう慣れましたか？今回の講義では vi を用いて、ネットワークに関する設定ファイルを編集し、Linux におけるネットワークの設定を行います。

ネットワーク診断ツール

ネットワークに障害が起きたとき、その原因究明の手がかりを得るためのコマンドで、次のようなツールがあります。

・ping

通信相手にパケットが正しく届くかどうかをチェックするツールで、相手にエコー（こだま）要求パケットを送信します。これを受信した側がエコー返信パケットを返すと、ping コマンドを実行した側は相手側に対してパケットが到着していることを確認することができます。

・traceroute

ネットワークにおける通信は、多くの場合途中で多くの中継点を通過します。traceroute コマンドは、通信が行われるときのホスト間の経路（route）を追跡（trace）し、通信相手に到達するまでの途中経路を表示します。

・nslookup

nslookup コマンドは、ネームサーバに問い合わせて、ホストIPアドレスの対応を検索します。

9.1 ネットワークの仕組み

コンピュータネットワーク（単にネットワークともいう）は、互いに通信回線を使って接続され、通信しあうことができるコンピュータの集まりを指します。いわゆる「インターネット」や会社内などで使われる「LAN（ローカルエリアネットワーク）」も、このコンピュータネットワークの 1 つです。インターネット上において、Web ページの閲覧や電子メールのやりとりが行われていますが、これらはすべて複数のコンピュータ同士が互いにデータを通信しあうことによって実現されています。インターネットや LAN の普及に伴い OS やソフトウェアもネットワークに対応したものとなりつつあり、情報の共有・交換なくしてはもはやコンピュータは語れない、という時代になりつつあります。

9.1.1 インターネット上のサービス

コンピュータネットワークの最も身近な例としては「インターネット」が挙げられます。Web ブラウザを用いた Web ページへのアクセスや電子メールのやりとり以外にも様々なサービスがインターネット上で提供されています。

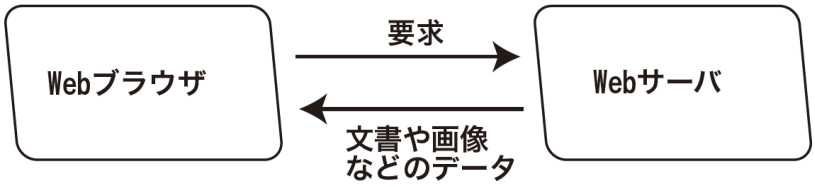
サービス	説　明
WWW	Internet Explorer や Firefox などの Web ブラウザソフトを用いて、Web サーバにアクセスすることにより、様々な情報を入手できる
電子メール	メーラ（メールソフト）を用いて、遠く離れた人と素早くメッセージ交換する
FTP	通信インターネット上のコンピュータと手元のコンピュータとの間でファイルのやりとりをする
TELNET	ログイン別のコンピュータにインターネットを経由して、ログインし、利用することができる
名前解決	IP アドレスとホスト名を対応させる（詳しくは後述）

9.1.2 クライアント・サーバモデル

インターネット上では、様々なサービスが提供されていますが、私たちは普段はこれらのサービスを享受する側にいます。サービスを受ける側に対して、サービスを提供する側が当然存在します。サービスを提供する側のコンピュータのことをサーバ（server）※1といいます。これに対し、サービスを受ける側のコンピュータのことをクライアント（client）といいます。また、ネットワークは、サービスの要求や応答の通信経路として機能します。

サーバー	サービスを提供する側（例えばコーヒーサーバ）
クライアント	サービスを消費する側（コーヒーを飲みたい人）

Web ブラウジングの場合では、クライアントとして Web ブラウザを用いるコンピュータがいて、このクライアントの要求に応じて、Web サーバが情報を提供する仕組みになっています。



様々なサービスは、1 種類のサーバのみで提供されているのではありません。Web サービスには Web サーバ、電子メールの送受信にはメールサーバというように、サービスに応じて様々な種類のサーバが動いています。一般的にこれらのサービスの機能を提供しているのはデーモン（daemon）と呼ばれるプログラムです。

サーバの種類	説　明	代表的なプログラム
Web サーバ	Web ブラウザに HTML 文書や画像などのデータを送信する	Apache,Nginx,llS(internet information services)
SMTP サーバ	電子メールサービスのうち、メール送信を受け持つサーバ	sendmail,Postfix,qmail
POP サーバ	電子メールサービスのうち、ユーザーからのメール受信要求に応える	ipop3d,qpopper
FTP サーバ	ftp コマンドのクライアントプログラムからの要求に応じてファイルのやりとりを行うサーバ	ProFTPD,wu-ftpd
TELNET サーバ	離れたコンピュータからのリモートログインを許可する	telnetd
ネームサーバ	名前解決を行うサーバ	BIND,djbdns

9.1.3 コンピュータ間の通信

クライアント・サーバモデルを見ても分かるようにインターネットは、コンピュータ同士が通信しあい、サービスが実現されることによって、成り立っています。コンピュータ同士が通信するためには、「物理的につながっている」だけでなく、「論理的（ソフトウェア的）にもつながっている」必要があります。これは、電話をかけるときには、電話線でつながっているだけではなく、電話番号を押して相手の電話機との接続を

※1「サーバ」というとき、単に提供する機能を指すだけでなく、ハードウェアそのものをサーバというときもあることに注意してください。

確立しなくてはいけないのと同じことだと捉えるとよいでしょう。ケーブルを介して電気信号が通ったとしても、その信号を両者が解釈できなければ意味のある通信は行えません。

またWebサーバなどのインターネット上に散在するサーバに対して、どのようにして接続するのか、接続先のサーバの場所をどのようにして特定するのかなどの決まりがないと、信頼できる通信手段とはなり得ません。特にネットワークにはさまざまなOSやソフトウェアが混在しており、接続方法や通信方法などについて共通の決まりごとが必要になります。そのため、コンピュータの通信にはプロトコル（protocol）という決まりが設けられており、これに基づいて様々な通信が行われています。

9.2 IP アドレス

IP アドレスは、ネットワーク上にあるコンピュータを識別するためのアドレス（住所）と解釈されますが、「コンピュータの電話番号である」と例えたほうがわかりやすいでしょう。この IP アドレスを用いることで、アクセスしたい Web サーバを指定することができます。電話番号が重複しないように割り当てられているのと同様に、個々のコンピュータには重複しない IP アドレスが割り当てられます。

IP アドレスはインターネットに限らず、ローカルエリアネットワーク（LAN）などでもコンピュータを特定するために用いられます。一般にあるコンピュータをネットワークにつなぐということは、IP アドレスが割り当てられるということを意味します。プロバイダへのダイヤルアップなども IP アドレスを取得するための手続きです。

9.2.1 2 進数と IP アドレス

IP アドレスは以下のように、0～255 の整数4組みを小数点で区切った形式で表されます。

123.45.67.89

コンピュータ内部では、全ての情報を 0 と 1 で全表す2進数を採用しています。先の IP アドレスもコンピュータ内部では次のように扱います。

123.45.67.89　→ 01111011001011010100001101011001（2進数）

このように IP アドレスは 32 桁の 2 進数で表されています。桁が長く扱いにくいので、8bit（通信業界では 1 オクテット、情報業界では1バイトと呼びます）ずつ区切り、4 組の 10 進数で表現します。

2 進数	01111011	00101101	01000011	01011001
	↓	↓	↓	↓
10 進数	123	45	67	89

IP アドレスは 32 ビット固定長なので、全部で 2^{32}（=4,294,967,296）約 43 億個のアドレスを表現できます。一見十分に思えるこの上限も、情報機器が普及が普及した現代では、とても足りません。そこで 128bit 長さに拡張した IPv6（IP バージョン 6）の導入が進められています。IPv 6のアドレス上限は約 340 澗（10^{36}）にもなり、当面不足しそうにありません。

[練習]

1. 「192.168.0.1」を 2 進数表現に直します。
2. 「100.150.200.250」を 2 進数表現に直します。
3. 「10101100 00010100 00000000 00000001」を IP アドレスの記法に直します。

9.2.2 ネットワークセグメント

インターネットのように巨大なネットワークも、元来は別々の小さなネットワークでしたが、少しずつ結びつくことで現在の規模にまで発展しました。そのため、現在でも大規模なネットワークは「小さなネットワーク」の集合体になっています。この小さなネットワークの単位のことを「ネットワークセグメント」と呼びます。個々のネットワークセグメントは独立しており、異なるネットワークセグメントに属するコンピュータ同士は直接通信することはできません。

IP アドレスは、ネットワークセグメントを表す「ネットワーク部」とネットワークセグメント中における自分自身を表す「ホスト部」から成り立っています。ネットワーク部はネットワークセグメントに固有の値となるので、あるネットワークセグメントに属するコンピュータは、ネットワーク部が共通でホスト部の異なる IP アドレスを持ちます。

電話番号であれば、-（ハイフン）で区切ることで、どこまでが局番であるかということを一目で認識できます。しかし、IP アドレスは数字の集まりであるため、それだけではどこまでがネットワーク部でどこからがホスト部であるのかということを判別できません。そこで IP アドレスの中でどこまでがネットワーク部であるのかを表すために「ネットマスク」という値が用いられます。

	ネットワーク部	ホスト部
一般表記	172 . 20 .	0 . 1 .
2 進数での表記	10101100 00010100	00000000 00000001
ネットマスク	11111111 11111111 255 . 255 .	00000000 00000000 0 . 0 .

ネットワーク部に相当する部分に 1 を埋め、ホスト部に相当する部分に 0 を埋めることにより、ネットマスクという値を定義します。

ネットマスクも IP アドレスと同様に 1 オクテットごとに区切って 10 進数に換えることにより、人が読みやすい形で表されています。各ネットワークセグメントに属する IP アドレスは、ネットワーク部は共通で、ホスト部が（2 進数表現で）全て 0 の IP アドレス（172.20.0.0）から、ホスト部が全て 1 の IP アドレス（172.20.255.255）の値を取ります。このうち、

ネットワークアドレス	ホスト部が全て 0 の IP アドレス
ブロードキャストアドレス	ホスト部が全て 1 の IP アドレス

は特別な IP アドレスでこれらを実際にコンピュータに割り当てることはできないことになっています。ネットワークアドレスはネットワークセグメント自体を表し、ブロードキャストアドレスはネットワークセグメント内の全てのコンピュータに対し同時に発信するときに用いられるアドレスです。

ネットマスクが 255.255.0.0 の 172.20.0.1 という IP アドレスを持つホストの属するネットワークセグメントは下の表のように 2^{16}（=65,536）個の IP アドレスを持つことができます。

ネットワーク部	ホスト部
10101100 00010100	00000000 00000000
10101100 00010100	00000000 00000001
10101100 00010100	00000000 00000010
10101100 00010100	00000000 00000011
10101100 00010100	00000000 00000100
：	： ：
：	： ：
10101100 00010100	11111111 11111100
10101100 00010100	11111111 11111101
10101100 00010100	11111111 11111110
10101100 00010100	11111111 11111111

しかし、ホスト部が全て0と1は、それぞれネットワークアドレス、ブロードキャストアドレスとして予約されています。

	IP アドレス表記	ネットワーク部	ホスト部
ネットワークアドレス	172.20.0.0	10101100 00010100	00000000 00000000
ブロードキャストアドレス	172.20.255.255	10101100 00010100	11111111 11111111

よって実際の機器に割り当てられる IP アドレスは 65,534（=65,536-2）個となります。

ネットワークセグメントを表すにはネットマスクを用いる方法と、

```
172.20.0.0/255.255.0.0
```

ネットワーク部をプリフィクスとして表現する方法があります。

```
172.20.0.0/16
```

[練習]

1. 192.168.0.0/24 というネットワークセグメントのネットワーク部は、何 bit ですか？
2. 192.168.0.0/24 というネットワークセグメントのホスト部は、何 bit ですか？
3. 192.168.0.0/24 というネットワークセグメントのネットワークアドレスとブロードキャストアドレスは？

9.2.3 IP アドレスのクラス

IP アドレスはその登場当初より、ネットワークの規模に応じて A～E の 5 つのクラスに分けられています。このうち通常用いられているのはクラス A～クラス C だけです。

クラス	説　明	IP アドレスの範囲
クラス A	ネットワーク部が 8bit 長 最上位 1bit が 0	0.0.0.0～127.255.255.255
クラス B	ネットワーク部が 16bit 長 最上位 2bit が 10	128.0.0.0～191.255.255.255
クラス C	ネットワーク部が 24bit 長 最上位 3bit が 110	192.0.0.0～223.255.255.255

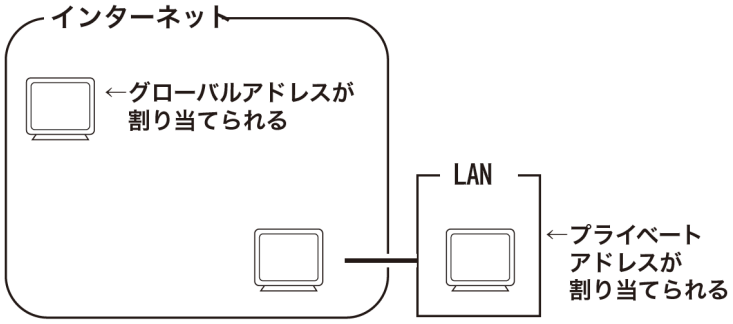
それぞれのクラスに属することのできる IP アドレスの数は以下のようになっています。

クラス A : 16,777,216 個（$=2^{24}$）
クラス B : 65,536 個（$=2^{16}$）
クラス C : 256 個（$=2^{8}$）

クラス A の中で「127.0.0.0」というネットワークはローカルループバックネットワークと呼ばれ、ホスト自身を表すネットワークセグメントを指します。その中のローカルホスト（自分自身）を表すために「127.0.0.1」が使われます。

9.2.4 グローバルアドレスとプライベートアドレス

IP アドレスには大きく分けて 2 つの種類があります。グローバルアドレスとプライベートアドレスです。グローバルアドレスはインターネット上に存在するホストに割り当てられるアドレスで、プライベートアドレスは LAN などのプライベートネットワーク内のみで使われるアドレスです。プライベートネットワークはインターネットから直接参照することができないネットワークセグメントを指します。



プライベートアドレスを利用することには

・外部と切り離すことによって、セキュリティを高める ・インターネット上における IP アドレスの枯渇を軽減する

という利点があります。

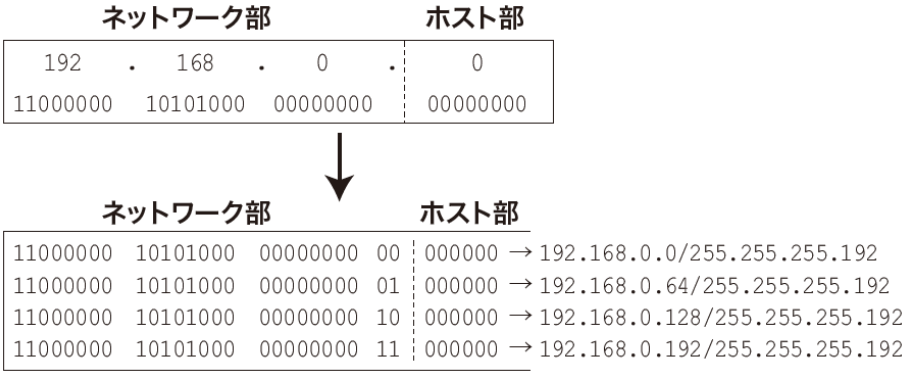
プライベートアドレスはグローバルアドレスと異なり、プライベートネットワーク上で重複することがなければどのようなアドレスを使っても良いことになっています。しかし、基本的にはグローバルアドレスとは異なるアドレスを使うことが推奨されていて、使用できるアドレスはクラスごとに決まっています。

クラス A	10.0.0.0～10.255.255.255
クラス B	172.16.0.0～172.31.255.255
クラス C	192.168.0.0～192.168.255.255

9.2.5 サブネット

IP アドレスの有効利用を目的として、ネットワークセグメントをネットマスクを用いてサブネットに分割するというものがあります。分割されたネットワークセグメントにおいても、ホスト部が全て 0 のものはネットワークアドレス、全て 1 のものはブロードキャストアドレスになります。

例えば、192.168.0.0/24 というネットワークセグメントを 4 つに分割するときには、ネットワーク部をクラス C の 24bit から 26bit に変更することにより可能となります。



今までホスト部であった下位 8bit のうち上位 2bit をネットワーク部として、

```
00      01      10      11
```

の 4 通りに固定したサブネットに分割しています。

それぞれのサブネットに属する IP アドレスは以下のようになります。

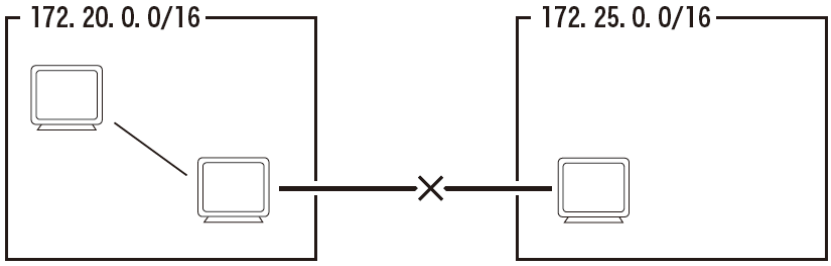
サブネット	IP アドレスの範囲	IP アドレスの数
192.168.0.0/26	192.168.0.0 ～ 192.168.0.63	64 個
192.168.0.64/26	192.168.0.64 ～ 192.168.0.127	64 個
192.168.0.128/26	192.168.0.128 ～ 192.168.0.191	64 個
192.168.0.192/26	192.168.0.192 ～ 192.168.0.255	64 個

[練習]

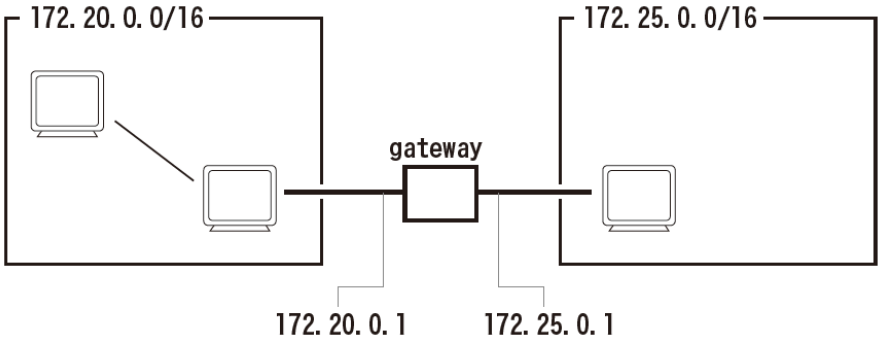
1. 192.168.0.64/26 のネットワークアドレスとブロードキャストアドレスは？
2. 192.168.0.64/26 で実際にマシンに割り振ることができる IP アドレスの数は？
3. 172.25.0.0/16 のネットワークセグメントを 1 つのサブネットに属する IP アドレスが 8 個になるよう分割するにはネットマスクは何 bit ですか？

9.2.6 異なるネットワークセグメント間での通信

各ネットワークセグメントは互いに独立していて、直接通信することはできません。



異なるセグメント間で通信を行うにはゲートウェイと呼ばれる装置（ルータなど）が必要になります。ゲートウェイは通信を行うネットワークセグメント双方の IP アドレスを持つことにより、互いの橋渡しをすることができます。



上の図のように 172.20.0.0/16 内の IP アドレスと 172.25.0.0/16 内の IP アドレスの双方を持つことによって、ゲートウェイは互いのネットワークセグメントの橋渡しをしています。

ゲートウェイは NIC（ネットワークインターフェースカード）※2を複数持っており、このような機能を果たすことができます。

[練習]

ゲートウェイを介さずに、通信可能な IP アドレスの組み合わせを以下から選んでください。

1. 172.20.0.32/16 ⇔ 172.16.0.31/16
2. 172.16.0.54/16 ⇔ 172.16.31.28/16
3. 192.168.1.56/24 ⇔ 192.168.2.33/24
4. 192.168.1.50/26 ⇔ 192.168.1.200/26

※2 NIC（ネットワークインターフェースカード）とは、ネットワークと接続を行うためにコンピュータに装着されるカードのことです。

9.2.7 名前解決

IP アドレスはネットワーク上のホストを表すために用いられますが、単なる数値の列で記述されているため非常に覚えにくくなっています。通常、Web ページなどを閲覧する際には www.linuxacademy.ne.jp などのようなホスト名を用います。

ホスト名と IP アドレスの間には対応関係があります。あるホスト名がどの IP アドレスを指しているのかを調べることを名前解決といいます。このサービスを提供するサーバを DNS サーバまたはネームサーバ（name server）と呼び、参照する DNS サーバは/etc/resolv.conf ファイルにおいて事前に設定されています。

現在では DNS サーバを用いた名前の管理が行われていますが、インターネットの規模がまだ小さかった頃は各マシンの IP アドレスとホスト名の対応を記述したファイルを準備していました。このファイルは今でも残っていて、/etc/hosts ファイルがそれに当たります。

/etc/hosts ファイルの例

```
127.0.0.1       localhost       localhost.localdomain
   ↑               ↑                    ↑
 IP アドレス      FQDN                 別名
```

9.3 Linux におけるネットワーク設定

9.3.1 基本的なコマンド

ip コマンド

ip コマンドは、ネットワークの設定および、現在の設定内容の表示を行います。同様のコマンドに ifconfig がありますが、IPv6 化に伴い現在は非推奨となっています。

例

```
$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
     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
           valid_lft forever perferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen
1000
     link/ether 00:15:5d:03:0d:04 brd ff:ff:ff:ff:ff:ff
     inet 172.30.1.137/16 brd 172.30.255.255 scope global enp3s0
           valid_lft forever preferred_lft forever
     inet6 fe80::215:5dff:fe03:d04/64 scope link
           valid_lft forever preferred_lft forever
（省略）
```

主な値

- デバイス名
  lo や enp0s3 は NIC の名称です。
- link（リンク層の値）
  Ethernet の MAC アドレス、brd に続きブロードキャストアドレスが表示されます。
- inet（IPv4 の値）
  IP アドレス/プレフィクス、brd に続きブロードキャストアドレスが表示されます。
- inet6（IPvP の値）
  IPv6 のアドレスが表示されます。

9.3.2 IP アドレスの設定

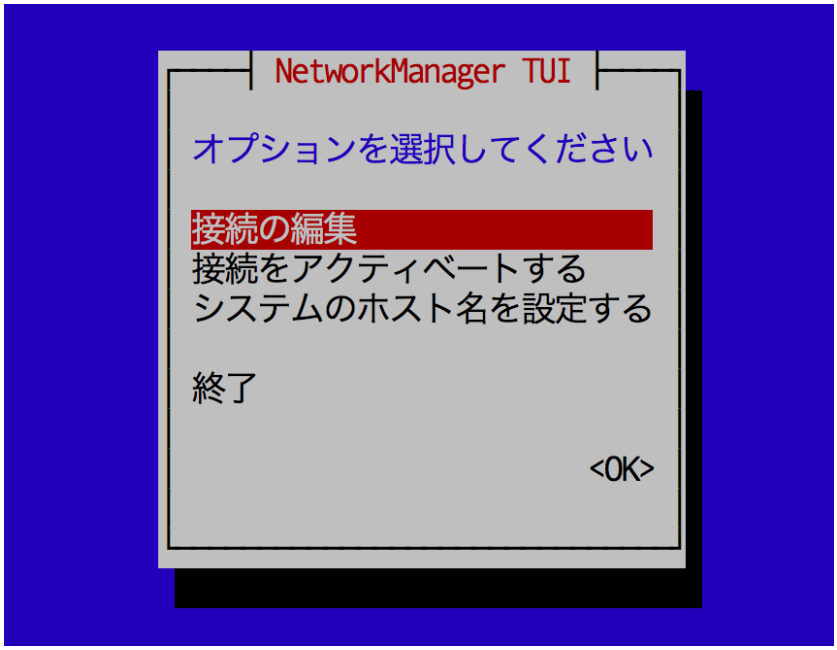
NIC の設定情報は/etc/sysconfig/network-scripts/ifcfg-<NIC 論理名>に記述され、恒久的な設定をおこないます。

NIC 個別設定：ifcfg-enp0s3 の例

```
DEVICE=enp0s3          ← NIC の名称
TYPE=Ethernet          ← ネットワーク種類（有線、無線など）
BOOTPROTO=static               ← 割り当て方法
（省略）
NAME=enp3s0
ONBOOT=yes
IPADDR=10.20.142.6
PREFIX=16
GATEWAY=10.20.0.1
DNS1=10.20.250.1
DOMAIN=s142.la.net
```

CentOS では、NetworkManager が自動的にネットワークの設定を行います。設定の変更は、nmtui で行います。

```
# nmtui
```



ネットワーク共通設定

NIC 個々ではなく、全体を通して共通の項目は/etc/sysconfig/network に記述します。設定内容が NIC 個別と重複した場合は、NIC の方が優先されます。

ホスト名

ホスト名はファイル/etc/hostname に記述します。古いタイプの CentOS では、先の network ファイルに HOSTNAME として記述しています。

```
$ cat /etc/hostname
h006.s143.la.net
```

設定ファイルを変更した後、その内容を反映するには systemctl コマンドを使います。

```
systemctl　サブコマンド　サービス
```

例

```
# systemctl restart network
```

他のサービスも、同様に systemctl のサブコマンドで起動・停止・再起動・状態確認が行えます。

サブコマンド	動作
start	サービス起動
stop	サービス停止
restart	サービス再起動
status	サービスの状況確認

9.4 ネットワーク診断ツール

ネットワークに障害が起きたとき、その原因の追求するためのコマンドを学びます。

9.4.1 ping

ping コマンドは指定したコンピュータに正しく届くかどうかをチェックするツールです。ping コマンドを実行すると、相手にエコー要求パケットを送信します。エコー要求パケットを受け取った相手側はエコー返信パケットを ping コマンドを実行した側に返します。これを受信することで、ping コマンドを実行した側は相手側に対して（少なくともネットワーク層のレベルまでは）パケットが到達することを確かめることができます。

ping コマンドのエコー要求パケットや返信パケットには ICMP（Internet Control Message Protocol）というプロトコルに基づいたメッセージが格納されています。ICMP はエコーを要求して、相手の応答を確かめるだけでなく、パケットが届くまでの時間やエラーの種類を通知したりする機能も持っています。

```
ping  [ オプション ]  相手先の IP アドレスまたはホスト名
```

例

```
$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.031 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.043 ms
^C
--- 127.0.0.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.031/0.037/0.043/0.006 ms
```

ping を終了するには、^C を押します。直後に統計情報が表示されます。ping コマンドの結果は以下の書式で表示されます。

64 bytes from	from	127.0.0.1:	icmp seq=2	ttl=64	time=0.043 m
↑		↑	↑	↑	↑
データサイズ		宛先	パケット連番	生存時間	かかった時間

データサイズは ECHO パケットの大きさで、規定値は 56 バイト＋IP ヘッダ8バイトの計 64 バイトです。ping コマンドは繰り返しパケットを送信し続けます。その連番が「icmp_seq=」の値で、連続していない場合、通信に欠落があることを示します。

パケットには目的地に到達するまで経由するルータの上限が指定され、これを TTL（Time to live）と呼びます。デフォルトでは 64 に設定されてルータを経由するたびに、この値は1つ減じられます。

time はパケットが返ってくるまでの時間をマイクロ秒やミリ秒で表示します。この時間が長すぎる場合、ネットワークのどこかに何らかの問題があることになります。

また、一定時間を過ぎても返答が受信できなかった場合は、「DestinationHost Unreachable」（目的のホストに到達できません）や「Timed out」（時間切れ）などのメッセージが表示されます。このようなメッセージが出た場合、ネットワークのどこかに重大な障害があることになります。

統計情報は rtt（Round-Trip Time、往復遅延時間）で示され、最短/平均/最長/標準偏差（ばらつき）を表します。

ping の繰り返し回数を指定するためには、-c（回数）オプションを付けます。

例

```
$ ping -c 2 127.0.0.1
```

［練習］

1. デフォルトゲートウェイのマシンに対して、ping コマンドを実行します。
2. 自分のコンピュータに ping を実行します。
3. 隣の人のマシンに対して、ping パケットを 5 回だけ送るように実行します。
4. www.example.net に対して、ping パケットを 10 回送信するように ping コマンドを実行します。

9.4.2 traceroute

ネットワークにおける通信は幾つかの中継点（ルータ）を介して行われます。
traceroute は通信相手に到達するまでに経由するルータの情報を画面に表示します。

```
traceroute ［ IP アドレス　又は　FQDN ］
```

単純な例

```
$ traceroute www.yahoo.co.jp
traceroute to www.yahoo.co.jp (183.79.198.79), 30 hops max, 60 byte packets
  1 172.30.0.1 (172.30.0.1) 1.371 ms 1.329 ms 1.312 ms
  2 c65-4a-R-v302.data-hotel.net (203.174.70.254) 1.576 ms 1.562 ms 1.553 ms
  3 480-1-ae14.data-hotel.net (203.174.66.29) 5.123 ms 5.111 ms 5.099 ms
（省略）
```

パケットに欠落がある例

```
10 po2.mcs02.net.tnz.yahoo.co.jp (203.216.238.194) 5.205 ms 6.377 ms 8.425 ms
11 * * *
12 * * *
13 * * *
14 * po2.mcs02.net.tnz.yahoo.co.jp (203.216.238.194) 13.933 ms !X *
```

「*」が出ているところは、なんらかの原因でパケットが到着しなかったことを表します。

時間の後に表示される「!」は何らかの問題が検出されたことを表します。「!H」はホスト、「!N」はネットワーク、「!P」はプロトコルに問題があることを表しています。「!X」はセキュリティの観点から情報提供を拒絶していることを表しています。

［練習］

1. 自分自身に対し、traceroute を実行します。
2. デフォルトゲートウェイに対して、traceroute を実行します。
3. www.linuxacademy.ne.jp に対して、traceroute を実行します。

9.4.3 nslookup

FQDN と IP アドレスの相互変換を行うコマンドです。nslookup は、インターネット上のネームサーバに問い合わせて、「FQDN から IP アドレス」または、「IP アドレスから FQDN」を検索します。また利用にあたっては、対話モードと非対話モードの 2 種類の方法があります。なお、問い合わせ先のネームサーバは /etc/resolv.conf ファイルで設定します。

```
nslookup [ FQDN/IP アドレス ]
```

非対話モードの例

```
$ nslookup localhost
Server: 192.168.0.1        ← 問合せ先ネームサーバ
Address: 192.168.0.1#53    ← ネームサーバのアドレスとポート番号

Non-authoritative answer:
Name: localhost
Address: 127.0.0.1
```

FQDN と IP アドレスの組合せが 2 対表示されますが、前者の方は問合せ先のネームサーバの情報で、後者の方が問い合わせた結果としての FQDN と IP アドレスです。

対話モードの例

```
$ nslookup
> localhost
Server:    192.168.0.1
Address:   192.168.0.1#53

Non-authoritative answer:
Name:      localhost
Address:   127.0.0.1
> exit
```

引数を指定せずに nslookup コマンドを実行すると、対話モードで起動されます。プロンプトが不等号「>」になり、そこへ問合せたいホスト名や IP アドレスを入力するたびに結果が表示されます。終了するには exit と入力します。

9.5 確認問題

1. 202.224.128.100 を 2 進数表記します。
2. 202.224.128.0/24 のネットワークのネットワークアドレス、ブロードキャストアドレスを求めます。
3. 2. のネットワークセグメントで実際にマシンに割り振ることができる IP アドレスの数を求めます。
4. 自分のマシンの IP アドレスを調べます。(ip コマンドを用いて)
5. nmtui コマンドを使って、以下の要件を設定します。
   「接続の編集」を選択し、enp で始まる NIC を編集します。
   ・IPv4 設定<手作業>
   　　アドレスは 10.20.xxx.yyy / 16
   　　ゲートウェイは 10.20.0.1
   　　DNS は 10.20.250.1
   ・IPv6 設定＜無視＞
6. 設定を保存したら、ネットワークを再起動し内容を確認します。

```
# systemctl restart network
```

7. www.linuxacademy.ne.jp に対して ping コマンドを実行し、通信が可能であることを確認します。
8. www.linuxacademy.ne.jp の IP アドレスを nslookup コマンドで調べます。
9. ブラウザを開いて、http://www.linuxacademy.ne.jp/に接続します。
10. 9.で調べた www.linuxacademy.ne.jp の IP アドレスをブラウザに入力します。http://[IP アドレス]/としても、同じページが表示されていることを確かめます。
11. traceroute コマンドで www.kernel.com への経路を調べます。

```
$ traceroute www.kernel.org
```

column

IPv4 と IPv6

IP アドレスの IP は、Internet Protocol（インターネットプロトコル）の略です。異なるコンピュータ間の通信はプロトコル（protocol, 規約）に基づいて行われています。IP もこのプロトコルの 1 つで、例えば IP アドレスを x.y.z.w と 4 つの値に分けて表すということもこの IP の中で決められています。

プロトコルは時と共にそれ自体発展していくものです。現在用いている IP は正式には、Internet Protocol version 4（IPv4）と呼ばれています。

しかし、この IPv4 について、現在いくつかの問題点が挙げられています。セキュリティや設計に関する問題など様々なものがありますが、その最たるものは「IP アドレスの枯渇問題」です。

IP アドレスは、0～255 の 256 個の整数を 4 つ並べたものです。つまり、理論的には 256^4 個、実に 4,294,967,296（約 43 億）個の IP アドレスを利用することができます。しかし、世界ではすでに大量のコンピュータが稼働していますし、今後も増加傾向にあることは間違いありません。また、いわゆるパソコン以外に、電話やテレビ、自動車などにもコンピュータは組み込まれています。そもそも世界人口の 68 億にも満たない容量の IP アドレスでは、絶対数が不足しているといえるでしょう。

そこで、現行の IPv4 を拡張・改善したものとして、現在、IPv6（Internet Protocol version 6）が提唱されています。この IPv6 では、前述の IP アドレスの枯渇の解決など様々な設計変更が行われ、大幅な改良が進められています。

目に見える変化としては、その IP アドレスの情報量が IPv4 の 32bit から、128bit へと拡張され、表記法も変わるということが挙げられます。IPv4 では 8bit ずつ「．」で区切り、10 進数に変換して表記していましたが、IPv6 では 128bit を 16bit ずつ「：」で区切って、それぞれを 16 進数で表記します。

例

```
0200:ff00:0000:0000:0000:0005:00AE:3141
```

16 進数では、1 桁で 10～15 までを表すために A～F という文字を代わりに用います。16 進数 1 桁は 4bit を表しますから、4 桁の 16 進数の数を 8 個並べることにより、128bit の情報を表現しています。

32bit から 128bit に変化したことで、IP アドレスの個数は今までの約 40 億個の 4 乗※4である、約 340 潤（1 潤は 10^{36}）個となります。この数は、世界人口を 60 億人として、一斉に 1 秒間に 1 ずつ数を数えたとしても、1031 年もかかってしまうほどの大きな数※5なのです。

※4 IPv4 の「4 倍の個数」と書かれている文献もありますが、これは誤りです。4 倍になったのは、bit 数です。
※5 ちなみに宇宙の年齢は現在の推定で、約 150 億年（おおざっぱにいって、10 の 10 乗年）とされています。つまり、宇宙の年齢の 3 乗もの時間がかかるというわけです。

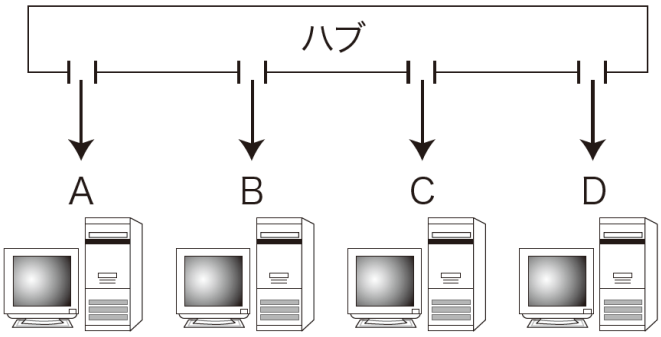
これだけの莫大な数の IP アドレスを利用することができれば、組織ごとに十分な数の IP アドレスを割り振ることができます。もしかしたら、1 家族に 1000 個の IP アドレスという時代も来るかも知れません。現在の IPv4 では、グローバル IP アドレスの取得に通常よりもコストがかさむということがありますが、IPv6 ではまるで当たり前のように様々なものに IP アドレスが割り振られていくこととなるでしょう。クラス A, B, C などの IP アドレスのクラスも IPv6 では廃止されることになり、より柔軟なネットワーク設計が可能となります。

IP アドレスに対する変更以外にも、様々な改良点があります。IPv4 では、IP spoofing と呼ばれる、外部のクラッカーのコンピュータが IP アドレスを偽装することで、あたかも組織内のコンピュータのように振る舞う攻撃がなされる場合があります。これに対し、IPsec という仕組みを用いれば、受信元から送られてくるデータが正当なものであることを確かめることができます。しかし、現状の IPv4 の設計で IPsec を用いるには、少々の困難が伴います。これに対し、IPv6 では IPsec を利用することを想定した設計がなされているので、安全性の高い通信を容易に行うことができるようになります。

既にいくつかのプロバイダでは、IPv6 を用いた試験サービスを行っていますし、IPv6 に対応したハードウェアや、OS や個々のアプリケーションなど続々と IPv6 に対応したソフトウェアが登場しています。現状の IPv4 からの移行は容易なものではありませんが、これからの発展が期待できる技術と言えるでしょう。

ネットワークを流れるパケットを見る

イーサネットを用いたネットワークでは、同じネットワークセグメントに流れるパケットは、宛先にかかわらず全てのマシンに届いています。



例えば、あるハブに A、B、C、D の4台のマシンが接続されていたとして、マシン A に届くべきパケットは、ハブを経由してマシン B、C、D にも届いて※6しまいます。その上で、それぞれのマシンはパケットのヘッダに書いてある宛先を見、自分が宛先でないと判断するとパケットを破棄するのです。

つまり、本来であれば破棄すべき他人にあてたパケットもマシン自体は受信しているということになります。この破棄すべき内容も含む、受信した全てのパケットを画面に表示するのが tcpdump コマンドです。

```
# tcpdump
16:14:07.119729 IP6 fe80::4d58:ecd1:cd7d:e8c8.61664 > ff02::1:3.hostmon: UDP, length 21
16:14:07.122486 IP6 fe80::4d58:ecd1:cd7d:e8c8.57438 > ff02::1:3.hostmon: UDP, length 21
16:14:07.221256 IP 192.168.0.53.56058 > 239.255.255.250.ssdp: UDP, length 174
16:14:07.222407 IP h240.46811 > anysec.apnic.net.domain: 53260% [1au] PTR?
250.255.255.239.in-addr.arpa. (57)
16:14:07.227151 IP anysec.apnic.net.domain > h240.46811: 53260- 0/9/1 (490)
16:14:07.227917 IP h240.55591 > b.iana-servers.net.domain: 23414% [1au] PTR?
250.255.255.239.in-addr.arpa. (57)
16:14:07.342023 IP b.iana-servers.net.domain > h240.55591: 23414 NXDomain*- 0/4/1 (511)
16:14:07.346187 IP h240.58423 > y.arin.net.domain: 868% [1au] PTR? 53.133.43.199.in-
addr.arpa. (55)
16:14:07.351083 IP y.arin.net.domain > h240.58423: 868- 0/6/1 (369)
（省略）
```

※6 スイッチングハグを用いている場合は、マシン A 宛のパケットはマシン A にしかハブから発せられないのでこの限りではありません。（スイッチングハブは交換機の役割をして、宛先のマシンにしかパケットを渡しません）

上の例では、NIC が受け取ったパケットを全て表示しています。ネットワーク上を流れるパケットの種類と送信元、送信先のホスト名などが記述されています。tcpdump コマンドは他にも色々なオプションをつけることにより、様々な用途に応じたパケットの情報を手に入れることができます。詳細は「man tcpdump」と入力することにより、tcpdump のオンラインマニュアルから調べることができます。
このようにネットワーク上を流れるパケットを見るためのツールを一般的にパケットスニッファ（packet sniffer,snif: 嗅ぎ回る）と呼びます。tcpdump は元々、ネットワーク管理者がトラブルの解決や不正なパケットを監視するためのツールとして開発されました。しかし、パケットスニッファの中には悪用され、ネットワーク上のデータの盗聴に用いられてしまうものもあります。例えば、TELNET によるリモート接続はパスワードが暗号化されない平文で流されるという欠点があります。悪意のあるユーザーが他人の経路の途中でパケットスニッファを用いてデータを盗聴し、簡単にパスワードを入手することが可能となっています。このため TELNET を用いたリモート接続は最近ではほとんど使われてはいません。

インターネット上で Web サイトを閲覧したり、メールを送受信したりすることの裏には、パケットが流れ、そしてそのパケットは簡単に見ることができるということを覚えておいてください。このことは、セキュリティ対策のキーポイントでもあります。

10. Linux のインストール

10

Linux のインストール

本章のねらい

- コンピュータの仕組みを学ぶ
- ハードディスクのパーティションについて学ぶ
- Linux のインストールについて学ぶ

予習編

コンピュータとは

コンピュータ内部で行われる処理は全て数値計算（詳しくは2進数やブール代数という）によって行われます。たとえば、ブラウザで Web ページを見るというだけで、Web サーバとのデータのやりとり、映像や楽音を数値に変換、それらのコンテンツをどこへどのように配置するかといったことまで、膨大な計算の結果なのです。

コンピュータは、その計算を司る CPU（Central Processing Unit）、途中の計算結果を蓄える主記憶装置（メモリ）、ハードディスクなどデータを永続的に記録する二次憶装置、利用者からの指示を与える入力装置、計算結果を知らせる出力装置からなります。

入力装置：	データを CPU に送る装置：キーボード、マウスなど
演算装置：	データを処理する部分（CPU が中心）
出力装置：	CPU がデータを送り出す装置：モニタ、プリンタ

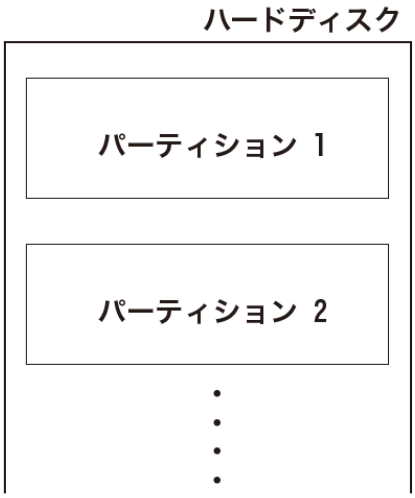
演算装置（CPU とメモリ）

CPU は計算を行うコンピュータの中枢装置で、コンピュータの性能を決定するほど重要な装置です。そのため用途により数多くの製品があります。たとえば PC では Intel/AMD の製品が多く使われていますが、スマートフォンやタブレット PC は省電力の ARM、スーパーコンピュータのような高速計算が求められる物には IBM/Oracle といった会社の製品が存在します。各 CPU は日本語、英語、フランスといったように、話す言葉が全く異なるります。そのため作業を記述したプログラムも各翻訳本のように CPU ごとに異なります。Linux ではこの CPU の違いをアーキテクチャと呼んでいます。

ハードディスクとその構造

ハードディスクは現在主流の大容量二次記憶装置です。処理速度はメモリとは比べものになりませんが、扱うデータ量が大きく、電源を切っても内容が失われません（不揮発性）。そのため OS やアプリケーションはハードディスクを中心とした二次記憶装置に保存されます。

ハードディスクはメモリに比べ膨大な容量を持つため、扱いやすい大きさに分割することができます。この分割する単位を「パーティション」と呼びます。またパーティション単位に書込み禁止設定を行ったり、特定のアプリ以外からはアクセスできないようにすることで、セキュリティを向上させることができます。



インストールとは

ソフトウェアを使えるようにするには、多くの場合、ハードディスクにプログラムをコピーする「インストール」（導入）作業を行なう必要があります。Linux 上で動作するソフトウェアも例外ではありません。また、インストールを行なうときには、事前にコンピュータの使用目的やコンピュータの状態に合わせた設定も行なう必要があります。最近のソフトウェアは、インストールの際に面倒な設定を自動で行なってくれるものが増えています。本章では、Linux ディストリビューションである CentOS7 をインストールします。

インストールの開始

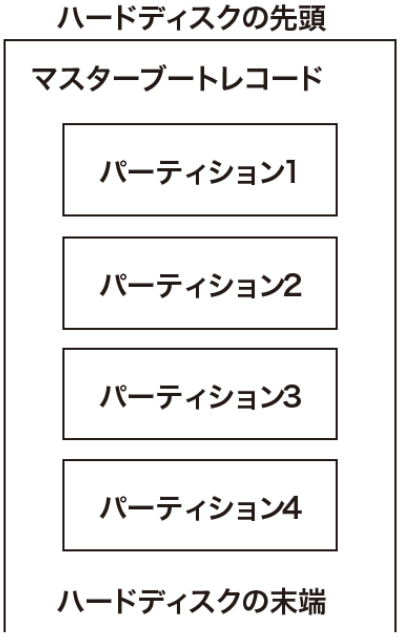
インストールを開始するときは、通常の起動とは違い、DVD-ROM に書き込まれているプログラムを使ってコンピュータを起動（ブート）します。なお、このとき DVD-ROM からブートできるように BIOS の設定変更が必要となる場合があります。DVD-ROM からコンピュータをブートするには、電源を入れた後、すぐに DVD-ROM を入れます。すると、DVD-ROM 内のプログラムに従ってインストールプログラムが起動し、インストールするプログラムがはじまります。

10.1 ハードディスクのパーティション

10.1.1 ハードディスク

ハードディスク（HDD: HardDisk Drive）は、記憶装置の一つです。処理速度はメモリなどに劣りますが、大容量で不揮発性※1なので、アプリケーションや OS などのソフトウェアを格納する装置として、用いられています。現在では半導体を使った SSD（Solid State Drive）も増えていますが、論理構造は HDD に似せています。

ハードディスクはパーティションと呼ばれる領域に分けることにより、複数の独立した記憶装置のように扱えるのです。



上図はパーティション分けされたハードディスクの模式図で、ハードディスクの先頭にマスターブートレコード（MBR）と呼ばれる領域があることを示しています。MBR にはパーティションの位置情報と OS を起動するためのプログラム（ブートローダ）が収められています。

※1 記憶装置では電源を切っても、記憶を維持していることを指す。

10.1.2 マスターブートレコード

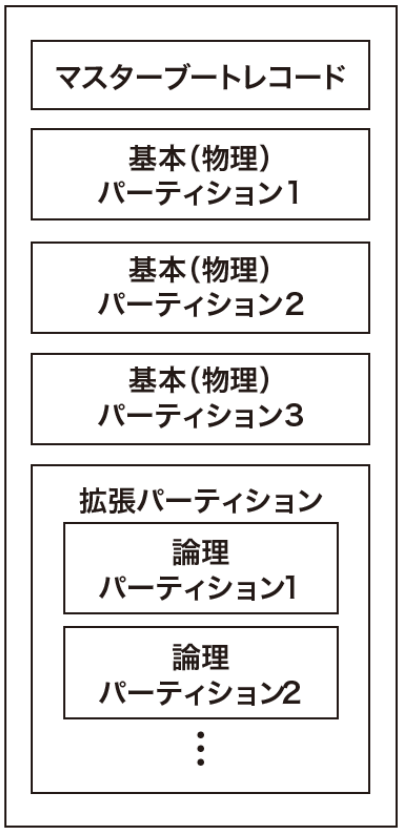
マスターブートレコード（MBR)はハードディスクの各パーティションに関する情報が格納されていますこれをパーティションテーブルと呼びます。

パーティションテーブルには

- 各パーティションの開始部分と終端部分
- パーティションのファイルシステムタイプ
- ブートデバイスの有無

などが記述されいてす。マシン起動時には BIOS がこのパーティションテーブルを読み込んで、ブートデバイスから OS を起動します。パーティションテーブルには最大 4 つのパーティションを登録することがで、これを物理パーティションと呼びます。

しかし、ハードディスクの大容量化により、4つ以上のパーティションが必要になりました。そこでパーティション4をさらに分割する拡張パーティションが導入されました。



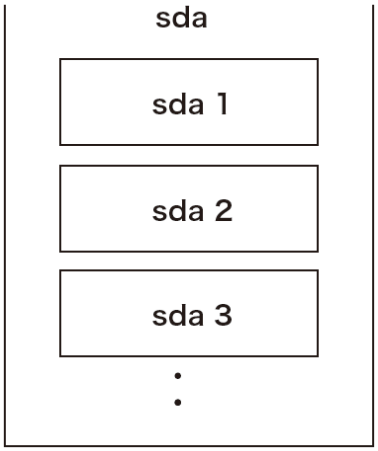
上の図のように拡張パーティションの中には複数の論理パーティションを持つことができます。拡張パーティションの中にパーティションテーブル作り、より多くのパーティションを管理します。

［練習］

1.fdisk コマンドを使って/dev/sda のパーティションテーブルを参照します。
サブコマンド p で内容を確認し、q で終了します。

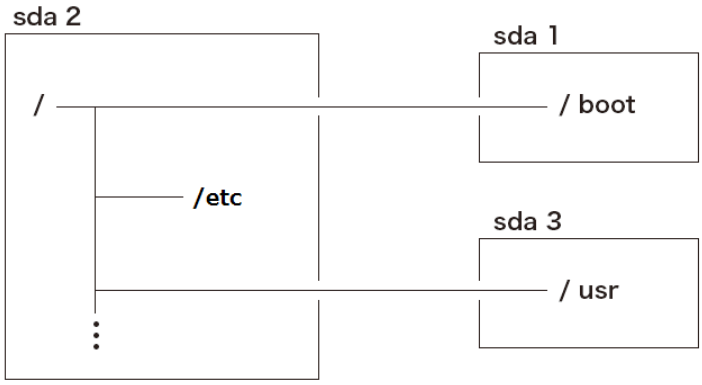
10.1.3 パーティションとファイルシステム

Windows などでは各パーティションは「C:」や「D:」といった「ドライブ」と呼ばれる名前が付けられ、識別されます。Linux では 1 台目の SATA 接続のハードディスクを sda、2 台目以降は順に sdb、sdc、…という名前※2が付けられます。個々のパーティションは sda1, sda2, …という様にハードディスク名に 1 から順に番号付けされます。



1 つのファイルシステムを作るためには 1 つのパーティションが利用されます。1 つのパーティション内に Windows 用のファイルシステムと Linux 用のファイルシステムを混在したりすることはできません。スワップ領域は独立した 1 つのパーティション（スワップパーティション）として作成されます。

スワップパーティションも特別な形式でフォーマットされています。多くの場合、Linux では複数のパーティションに分けてファイルシステムを格納します。/boot ディレクトリは sda1 に、/（ルートディレクトリ）は sda2、/usr ディレクトリは sda3 に、というように用途によってファイルシステムのツリー構造を複数のパーティションに分けて配置します。



Linux が起動しているときにはこれらのディレクトリは 1 つの大きなツリー構造になっていますが、実際

※2 IDE と呼ばれる古いタイプのハードディスクは hd、現在主流の SATA や SCSI は sd が用いられます。

には別々のパーティションに分けて格納されています。起動時に個々のパーティションを各ディレクトリに「マウント」することで、論理的にはルート（/）を頂点とする 1 つのファイルシステムを形成します。

特に/boot は OS が起動する前にブートローダが参照するため、他のファイルシステムと異なる形式を割り当てることがあります。

10.2 CentOS のインストール準備

CentOS のインストール・キット（DVD やファイルで提供されるインストール用ファイル群）には、必要なファイルが全て収録された DVD-Everything 版や、ネットワークを経由しインストールするために必要なプログラムだけを収録した NetInstall 版、ハードディスクへは書き込まずメモリだけで起動する Live 版など複数の種類があります。このような情報はリリースノートに記載されているので、まずはそちらを参照してください。

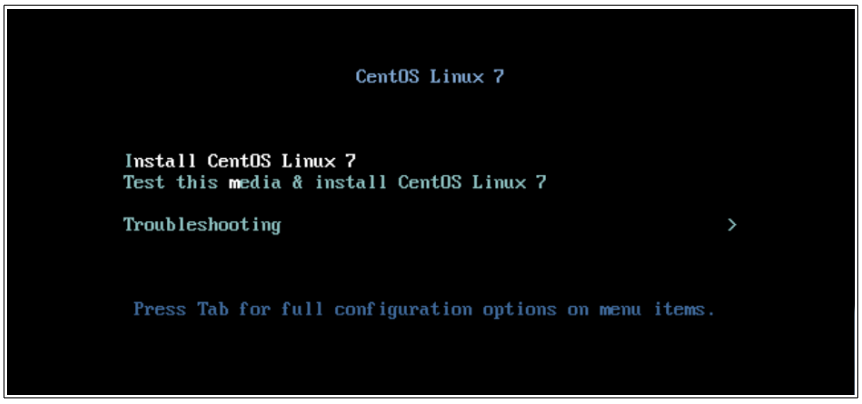
https://wiki.centos.org/Manuals/ReleaseNotes/CentOS7/Japanese

インストール・キットはインターネット上で公開されており、日本国内では以下のサイトが有名です。

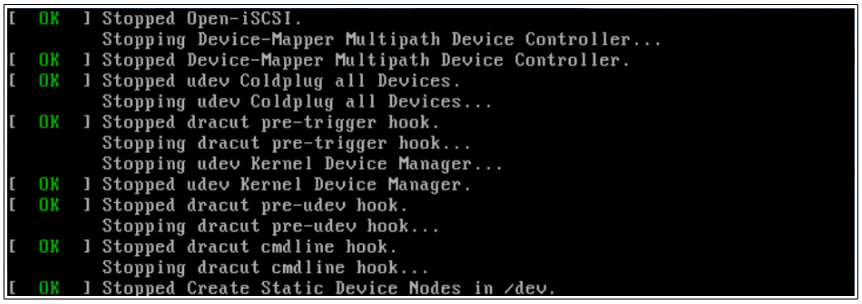
理化学研究所	ftp.riken.jp：Linux/centos/7/isos
KDDI 研究所	ftp.kddilabs.jp：Linux/packages/CentOS/7/isos/
株式会社 IIJ	ftp.iij.ad.jp：pub/linux/centos/7/isos/

10.2.1 メディアからの起動

BIOS のブート順序（boot sequence）で、DVD が先頭になるよう調整[1]した後、インストール・キットをセットし PC を起動します。インストールプログラム（以下、インストーラ）が起動され以下の画面が表示されます。



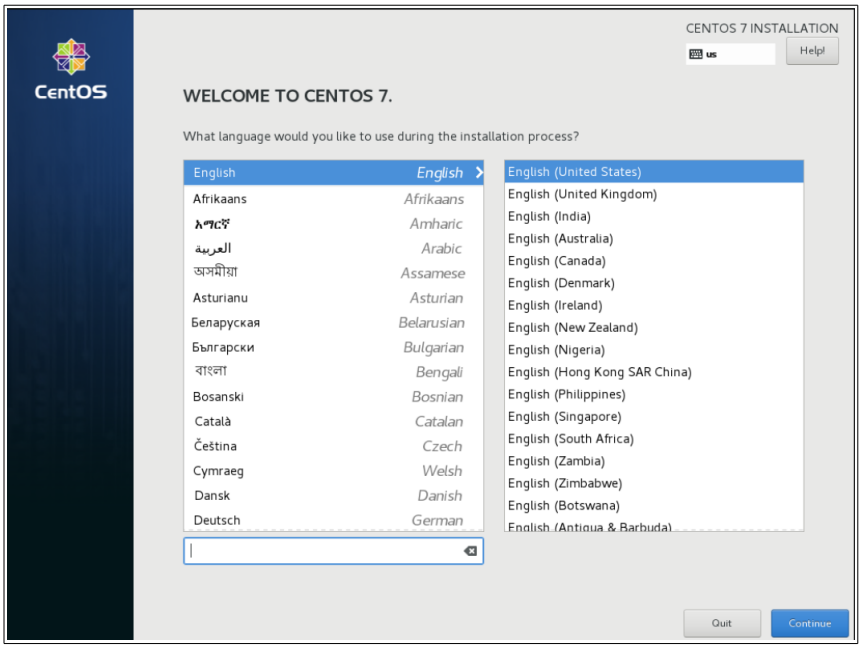
矢印キー「↑」「↓」を操作し、Install CentOS Linux 7 を選択し、[Enter]を押しインストールを開始します。以下の画面が表示されたあと、GUI が起動します。



※1 BIOS 設定の具体例は p.152「補足」を参照のこと。

10.2.2　言語の選択

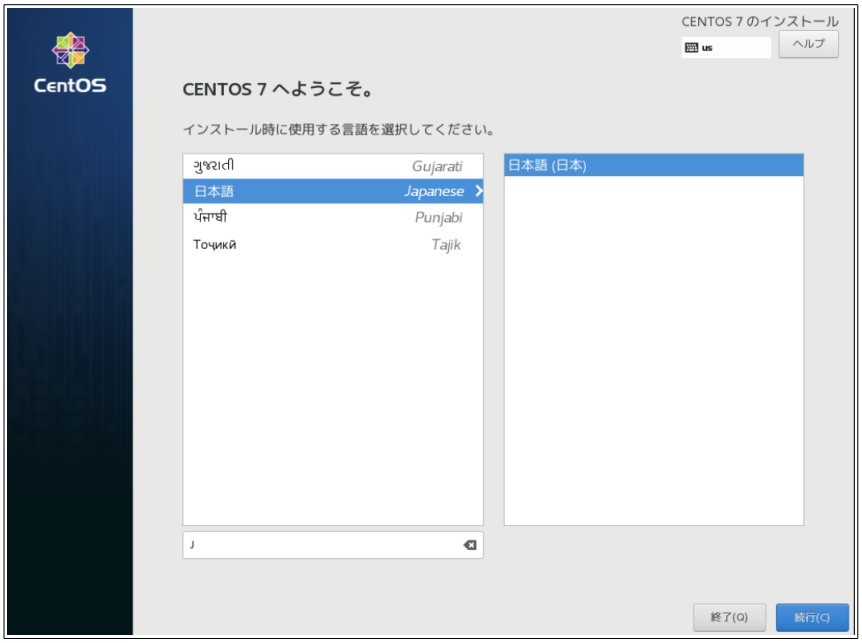
インストーラが正常に起動されると、GUI が起動し言語選択画面が表示されます。



国と言語を選択します。右下の[続行（C）] をクリックします。検索欄に「j」を入力すると素早く日本語を見つけられます。

国の選択

日本を選択し、[続行（C）] をクリックします。



この画面は複数の地域に展開している言語を意識しています。例えば英語であれば米国、英国、香港など複数の地域にまたがっています。

10.3 インストールの詳細設定

10.3.1 地域設定

言語を選択するとインストーラが規定値を設定します。

地域設定は国に基づき設定され、以下の値であることを確認します。

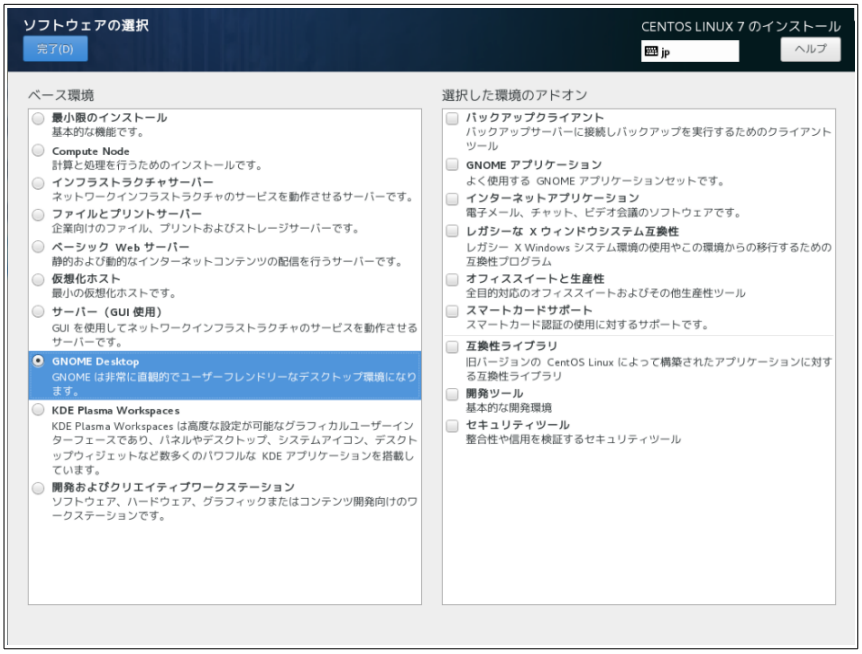
- 日付と時刻（T） アジア／東京タイムゾーン
- キーボード（K） 日本語（右上のキーボードアイコンに jp と表示）
- 言語サポート（L） 日本語（日本）

続く解説で、ソフトウェアとシステム項目を設定します。

10.3.2　ソフトウエア

ソフトウエア欄で、左側の「インストールソース」が「ローカルメディア」（DVD）であることを確認します。

続いて右側の「ソフトウエアの選択」をクリックします。



左側の「ベース環境」として「GNOME Desktop」をクリック（規定値では「最小限のインストール」）。右側の「選択した環境のアドオン」からは何も選択しません。確認できたら、左上の [完了（D）] をクリックします。

10.3.3　システム

インストール先

再び「インストール概要」画面から、「システム」の「インストール先」をクリックします。

デバイスの選択

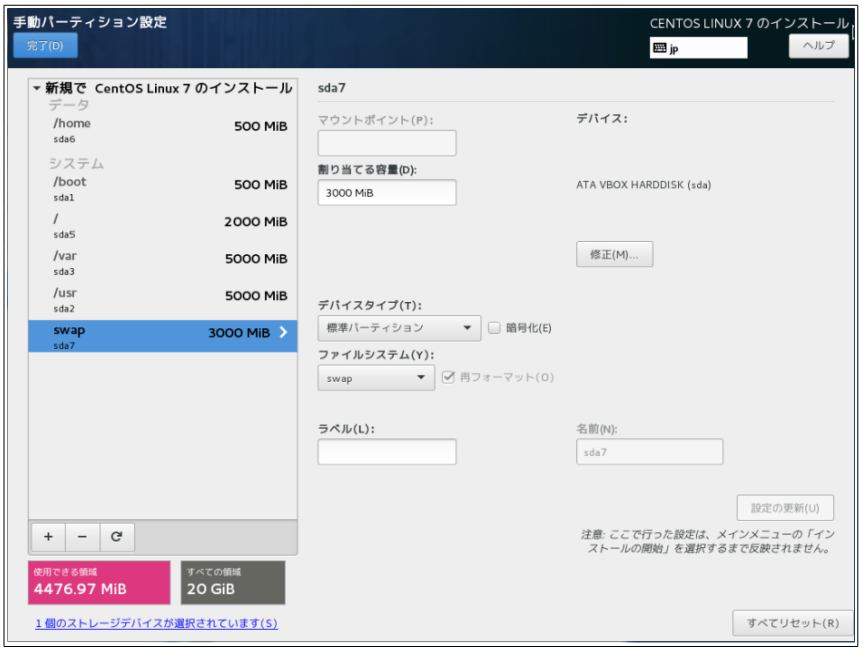
インストール先として外付けのハードディスク（sda、80GiB 程度）が選択されていることを確認します。

その他のストレージのオプション

パーティション構成の「パーティションを自分で構成する（I）」にチェックを入れ、左上の[完了（D）]をクリックします。

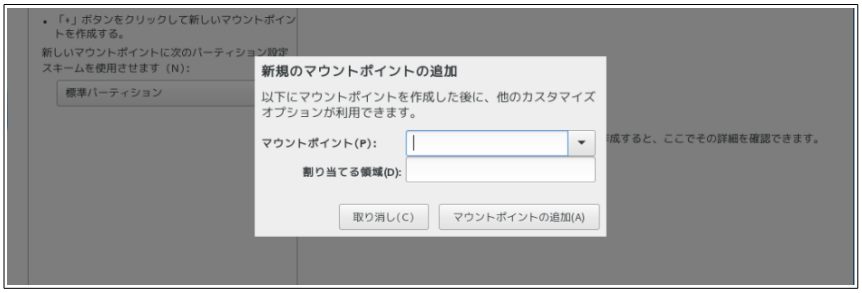
手動パーティションの設定

パーティションがすでに設定されている場合は左の枠内に、それらの一覧が表示されます。パーティションを選択し、左下の[-]ボタンをクリックするとパーティションが削除されます。新規追加は[+]ボタンにより実施します。



この時、左側のパーティション一覧に「ntfs」と表示されている場合は、内蔵ハードディスクを選択している可能性が高いので、[完了(D)]を 2 回クリックし、選択し直してください。

今回は一旦全てのパーティションを削除した後、以下の内容で再度作成します。追加を行うと「新規のマウントポイント追加」画面が表示されるので、



以下の要領で追加し [マウントポイントの追加]をクリックします。

・マウントポイントはリストから選択と、直接入力ができます。
・容量には単位（M、G）を指定できます。省略時は M です。
・ファイルシステムは SWAP 以外は XFS とします。

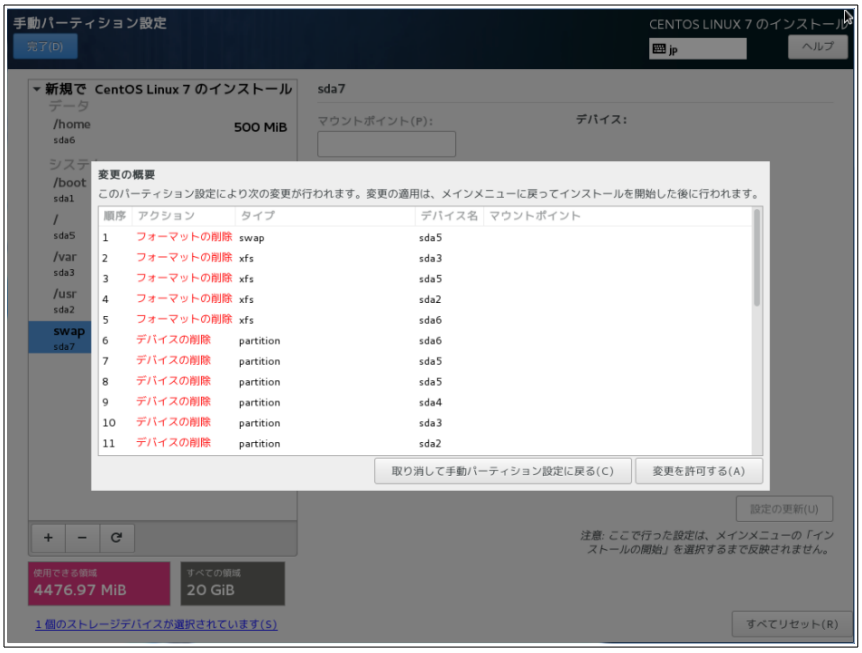
パーティション割り当て一覧※3

マウントポイント	割り当てる容量	ファイルシステム
/boot	500M	xfs
/	2,000M	xfs
/usr	5,000M	xfs
/home	500M	xfs
/var	1,000M	xfs
swap	3,000M	swap

※3 インストールのデフォルトでは、/boot, /, /home,swap の 4 パーティションが作成されます。

変更の概要

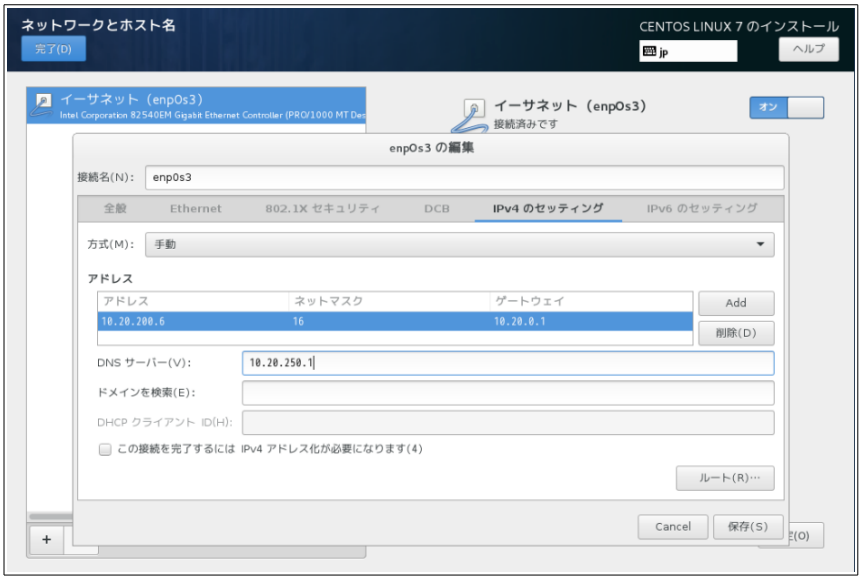
パーティションを全て登録し、左上の[完了 (D)] ボタンを押すと、確認画面が表示されます。



内容を確認し、[変更を許可する (A)] ボタンをクリックします。

ネットワークとホスト名

再び「システム」から「ネットワークとホスト名」をクリックします。



左の一覧から「イーサネット」を選択し、右上のスライダをクリックし[オン]にし、右下の[設定(C)]ボタンをクリックします。

「IPv4 のセッティング」タブから、以下の内容を入力します。実際に入力する値については講師の指示に従ってください。

- 方式　　手動
- アドレス 10.20.xxx.yyy(講師に確認します)
- ネットマスク　　16
- ゲートウェイ　　10.20.0.1
- DNS サーバ　　10.20.250.1

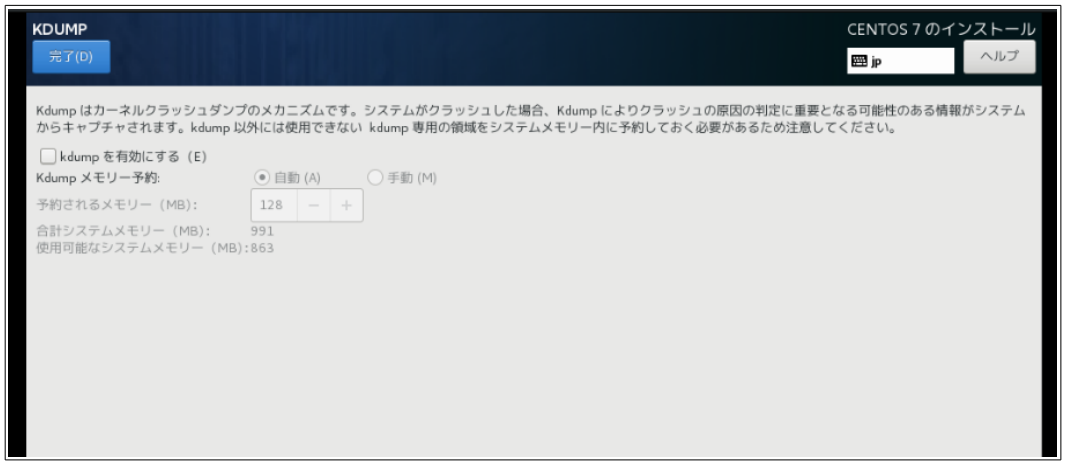
「IPv6 セッティング」タブで、方式に「無視」を設定します。

入力を終えたら、[保存(S)] ボタンをクリックします。

左下の「ホスト名(H):」にホスト名を入力し、左上の[完了(D)] ボタンを押します。

KDUMP

KDUMP はカーネルに異常があった際に、メモリの内容を全て HDD へ書き出し解析に役立てるものですが、今回は使用しません。



左上の「kdump を有効にする(E)」のチェックを外し［完了(D)］をクリックします。

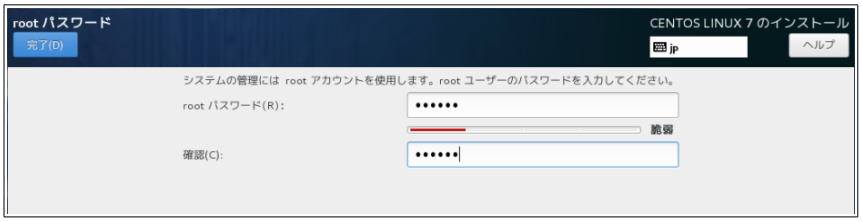
10.4 CentOS のインストール

「インストール概要」画面に戻り、右下の[インストール開始(B)] ボタンをクリックすると、インストールが開始されます。

インストール(ファイルのコピーや生成)と並行してユーザの設定を行います。この画面が表示されている間、DVD からハードディスクにソフトウエアのコピーが行われています。

10.4.1 root パスワード

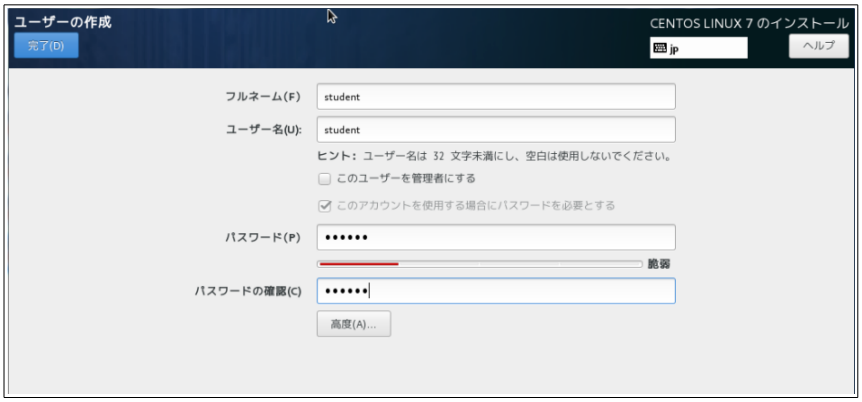
「root パスワード（R）」をクリックすると、設定画面が表示されます。ここで root のパスワードを入力し、2 行目にパスワードの確認として、同じ内容を入力します。



入力が終わったら左上の[完了（D）] ボタンを押します。パスワードが脆弱（簡単）な場合は、確認のため [完了（D）] ボタンを 2 回押す必要があります。

10.4.2 ユーザの作成

再びインストール画面から「ユーザの作成」をクリックし、ログイン時に使う一般のユーザを作成します。



以下のように入力します。

- フルネーム　　このユーザの説明を入力します、自身の氏名を入力します
  （例では student ですが、任意で構いません）
- ユーザ名　　student とします

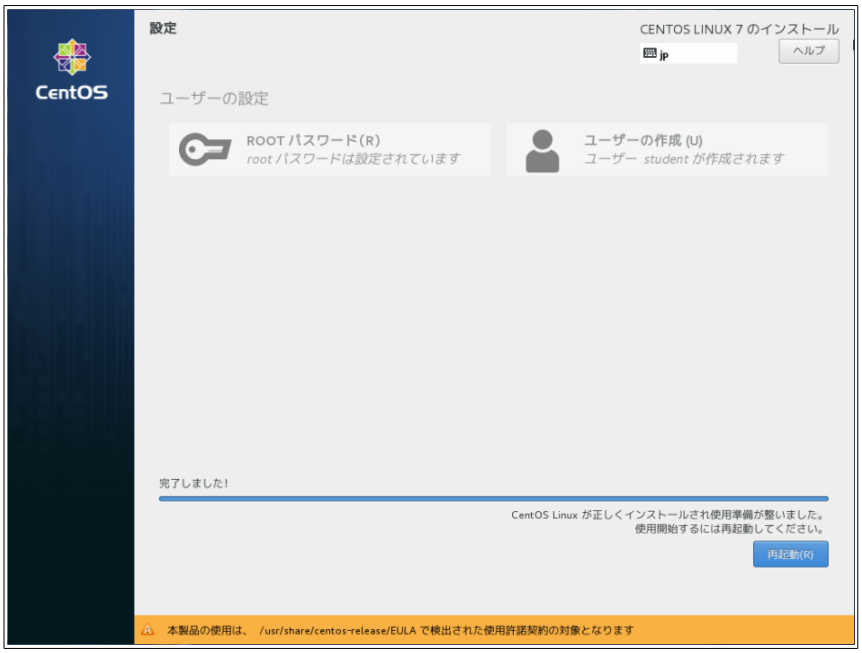
次の項目にチェックを入れます

- 「このアカウントを使用する場合にパスワードを必要とする」

パスワードは root と同様に 2 回入力し、[完了（D）」ボタンを押します、簡単なパスワードの場合は 2 回クリックが必要です。

10.4.3　完了

インストールが完了したら、右下の[再起動（R）] ボタンを押します。自動的に DVD が出てきますので、DVD を取り出します。



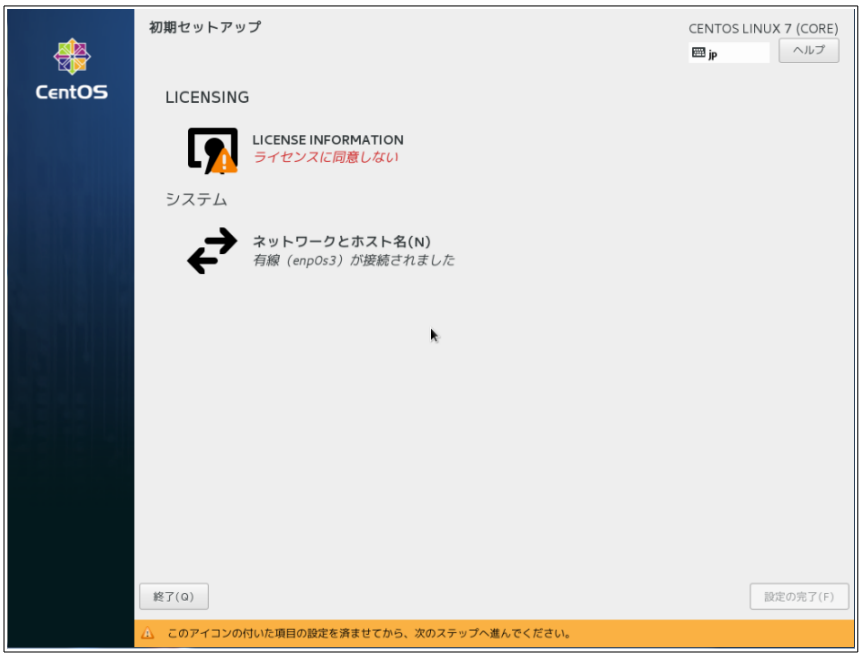
10.5 インストール後処理

再起動すると、普段利用する OS と、トラブル時の回復用 Rescue モードの 2 種類が選択できます。特に指定しない場合は5秒後に普段利用の OS が起動します。

矢印「↑」「↓」キーを使って OS を選択できます。

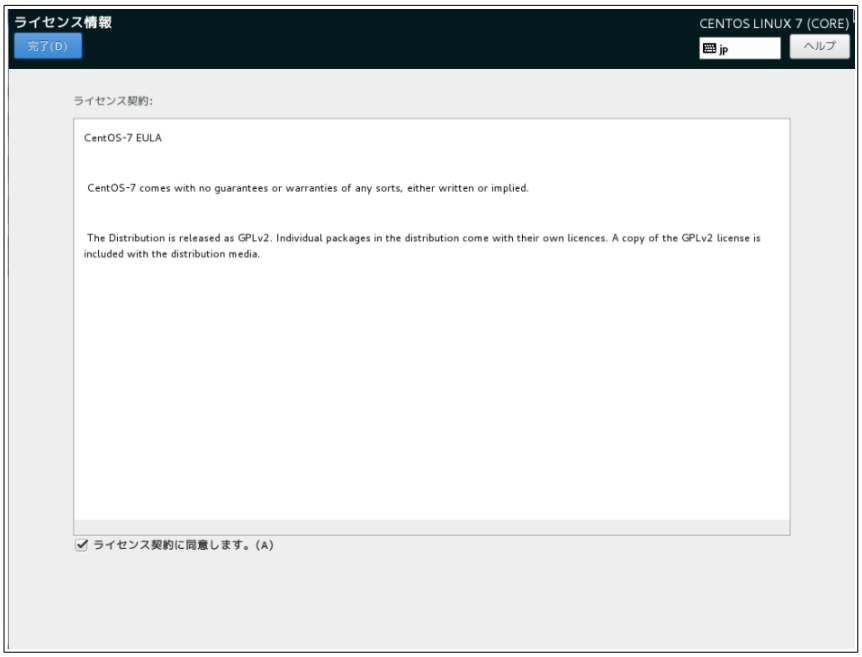
10.5.1 初期セットアップ

インストール後、初めて起動した際にはライセンス確認とネットワーク確認画面が表示されます。



ライセンス情報

ライセンスに関する情報が表示されます。必要に応じ内容を確認し、問題なければ「ライセンス契約に同意します。」にチェックを入れ、[完了 (D)] をクリックします。



ライセンスに同意すると、画面右下の［設定の完了(F)］が押せる状態になるので、クリックします。

ライセンスに同意できない場合は、CentOS を使用できません。ここで終了してしまいます。

10.5.2　ログイン

ユーザーの作成で指定した「フルネーム」が表示されるので、それをクリック、パスワード入力によるログインが開始されます。

パスワードを入力し、[サインイン]をクリックします。

言語設定（ようこそ）

初めてログインした際には、普段利用する言語の確認が行われます。

入力ソース選択

日本語の場合は、さらに文字入力方法を設定します。

日本語（「日本語入力（かな漢字）」）を選択します。

Wi-Fi

授業では使わないので、オフにして、スキップします。

プライバシー

位置情報サービスは「オフ」にし、[次へ(N)]をクリック。

オンラインアカウント

Google や Microsoft のアカウントは必要ないので、[スキップ(S)]をクリックします。

利用開始

以上で設定は終わりです。[CentOS Linux を使い始める(S)]をクリックします。

続いて自動的に GNOME ヘルプが立ち上がります。

使用しないので、右上の[X]をクリックし終了します。

セキュリティの設定変更

Linux Academy の Linux ベーシック、マスターコースでは設定を簡便に行うためセキュリティの強度をあえて抑えた環境を用いています。インストール完了後、以下の作業を行います。

1. /etc/selinux/config の SELINUX を enforcing から disabled へ変更

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=disabled        ←ここを修正。似た行が下にもあるので注意。
...
```

2. ファイアウォールの停止

```
# systemctl stop firewalld          ←稼働中のサービスを終了
# systemctl disable firewalld       ←自動起動しないよう抑制
```

3. 再起動後確認

システムを再起動し、上記の 2 点が正しく設定されているかを確認します。OS 起動後、下記のような結果が得れれば設定は完了です。

```
$ getenforce
Disabled
$ systemctl status firewalld
●firewalld.service - firewalld - dynamic firewall daemon
     Loaded: loaded (/usr/lib/systemd/system/firewalld.service;
disabled; vendor preset: enabled)
     Active: inactive (dead)
       Docs: man:firewalld(1)
```

10.6 システムの起動

10.6.1 ターゲットとランレベル

OS は起動すると、続いてネットワークや各種サーバといった必要な機能（サービス）を順次起動してゆきます。予め設定した内容に従い、起動するサービスやマウントするファイルシステムなどを準備します。このサービスやファイルシステムの組合せをターゲットまたはランレベルと呼びます。

順次サービスを起動する仕組みには Linux の黎明期から使われている init（SysVinit）、サービスの起動を並走させ効率化させた Upstart、より細かくサービスを制御しセキュリティ面を向上させた systemd があります。init（CentOS5 以前）、Upstart（CentOS6）ではランレベルが、CentOS7 の systemd ではターゲットを採用しています。

CentOS のランレベルとターゲット※4

ランレベル	ターゲット	動作
0	poweroff	システム停止（全てのサービスを停止後、OS 停止）
1	rescue	シングルユーザモード（必要最小限、保守用）
2		マルチユーザモード　NFS・GUI なし
3	multi-user	マルチユーザモード　GUI なし
4		マルチユーザモード　その他（CentOS では未使用）
5	graphical	マルチユーザモード　GUI あり
6	reboot	再起動（システム停止後、再度起動）

systemd では個々のサービスやファイルなどをユニットという単位で管理しています。

主な systemd のユニット

ユニット	解説
Target	複数のユニットの集合。OS の動作状況に応じたサービスの提供。
Service	サービスの提供（systemd 規定値）
Path	サービスの動作に必要なファイルやディレクトリ
Mount	マウントすべきファイルシステム
Socket	プロセス間通信
Slice	プロセス内のリソース割り当て

特定のサービスに必要となる種々のユニットを一括して管理することで、CPU、メモリといったコンピュータ資源を効率よく利用できるよう設計されています。このコンピュータ資源をまとめて管理する OS の機能を cgroup と呼びます。

※4 網掛けは CentOS7 で未対応となります。

10.6.2 ターゲットの切り替え

systemctl コマンド（CentOS7）または init コマンド（CentOS6 以前）により、現在のターゲットを変更することができます。

```
# systemctl rescue ←または init 1
（暗転後コンソールへ）
Welcome to rescure mode! After logging in, type “journalctl -xb” to view
system logs, "systemctl reboot" to reboot, “systemctl default” or ^D to boot into
default mode.
Give root password for maintenance
(or type Control-D to continue):
```

上記はマルチユーザモードからシングルユーザモードへ移行しています。

シングルユーザモードは必要最小限のサービスしか起動しないため、殆どのサービス、ネットワークも起動しません。さらにログインさへも起動しないため直ぐにプロンプトが表示され root ユーザとして操作することになります。なお exit により終了すると、自動的にマルチユーザモードへ移行します。

```
# exit
```

デフォルトターゲット

OS 起動時のターゲットをデフォルトターゲットと呼び、systemd では systemctl コマンドの set-default で変更できます。現在の値は get-default で参照できます。

以下は、multi-user ターゲットをデフォルトターゲットとした例です。

```
# systemctl set-default graphical.target
Removed symlink /etc/systemd/system/default.target.
Created symlink from /etc/systemd/system/default.target to
/usr/lib/systemd/system/multi-user.target.
# systemctl get-default
multi-user.target
```

実際に行われた操作が表示され、/etc/systemd/system/default.target が /usr/lib/systemd/system/ターゲット.target へのシンボリックリンクに置き換わっていることがわかります。

ターゲットにより起動するサービスやファイルシステムは、/usr/lib/systemd/system/ターゲット.target.wants/下に個々の定義ファイルをシンボリックリンクで格納します。

```
# ls /usr/lib/systemd/system/multi-user.target.wants/
brandbot.path     plymouth-quit-wait.service  systemd-logind.service
...
```

SysVinit や Upstart では、/etc/inittab ファイルの id:の直後にランレベルを記述します。例えばランレ

ベル3の設定は次のようになります。

```
# エントリ識別子：ランレベル：アクション：コマンド
id:3:initdefault:
```

[練習]

1. 現在のターゲットを確認します。

2. ターゲットを multi-user に変更し、再起動します。

3. root でログインし、ターゲットを graphical に戻します。

4. グラフィカルターゲットへ以降します。

10.6.3 自動起動設定

特定のサービスをシステム起動時に自動開始するか否かは systemctl コマンドで行います。登録されているサービスの一覧は list-unit-files サブコマンドを使います。

```
$ systemctl list-unit-files --type service
UNIT FILE                   STATE
abrt-ccpp.service           enabled
abrt-oops.service           enabled
（省略）
```

STATE 欄が enabled のサービスは自動起動設定、disabled は手動起動、static は他のサービスに依存していて変更できないことを表しています。

自動起動設定も systemctl コマンドで行います、以下が主なサブコマンドです。

systemctl enable　サービス	サービスを自動起動する
systemctl disable　サービス	サービスの自動起動はしない
systemctl start　サービス	手動によるサービスの起動
systemctl stop　サービス	手動によるサービスの停止
systemctl restart　サービス	手動によるサービスの再起動
systemctl status　サービス	サービスの状況表示

以下は CUP サービス（汎用印刷）の自動起動を無効化した例です。

```
# systemctl disable cups
Removed symlink /etc/systemd/system/multi-user.target.wants/cups.path.
Removed symlink /etc/systemd/system/multi-user.target.wants/cups.service.
Removed symlink /etc/systemd/system/sockets.target.wants/cups.socket
Removed symlink /etc/systemd/system/printer.target.wants/cups.service.
# systemctl status cups
●cups.service - CUPS Printing Service
Loaded: loaded (/usr/lib/systemd/system/cups.service; disabled; vendor preset:
enabled)
Active: active (running) since Wed 2017-08-02 09:40:32 JST; 19min ago
（省略）
```

サービスだけでなく、関連ファイル（path）、通信（socket）ユニットも合わせて無効化されていることがわかります。

11. ApacheHTTP サーバ（1）

11

ApacheHTTP サーバ（1）

本章のねらい

- Web の仕組みを学ぶ
- Apache Web サーバを起動する

予習編

Web ページの閲覧

現在「インターネット」は情報のインフラとしての地位が確立され、公私ともになくてはならない存在になりました。皆さんも Internet Explorer や Firefox, Chrome, Safari などを使って、アドレス（URL）の欄に「http://www.yahoo.co.jp/」などと入力し、情報を手にしたことがあるでしょう。この表示は「Web ページ」と呼ばれます。「インターネット」は、膨大なコンピュータ同士が相互に接続された巨大なネットワークを指します。そして、私たちは日頃「Web ページ」を「インターネット」を通して閲覧しているのです。

また、この Web ページを閲覧するために使う IE, Chrome, Safari などのソフトウェアを「ブラウザ」と呼びます。CentOS7 には FireFox と呼ばれるブラウザが付属しています。

HTML

Web ページの実体は、主に HTML（HyperText Markup Language）という形式のファイルです。HTML はテキストファイルで、その記法は独特のものです。エディタを使って HTML を編集することで Web ページを作成・変更することができます。Windows では HTML ファイルを開くとブラウザによって表示されますが、Internet Explorer の「表示」メニューの「ソース」を見ると、HTML の正体を見ることができます。

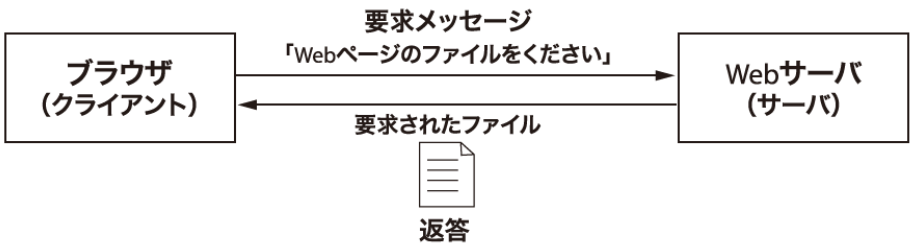
HTML の例

```
<html>
<head>
<title>LA.net</title>
</head>
<body>
Hello! World!
<a href="next.html">Click here!</a>
</body>
</html>
```

ブラウザの役割は、HTML 文書を Web ページの形に整形して画面に表示することです。なお、HTML だけでは画像などを表示できないため、HTML と画像ファイルを組み合わせてグラフィカルな Web ページにすることが多いようです。

HTTP

ブラウザを使って Web ページを閲覧する場合、ブラウザがサーバに対してデータの要求をします。サーバは要求されたデータファイルをブラウザに送ります。このとき、クライアントからの要求に応じてデータを送信するサーバを Web サーバといいます。また、Web サーバとクライアント間で、要求や返答などのやりとりに使われるプロトコルを「HTTP（HyperText Transfer Protocol）」といいます。



Web サーバの構築

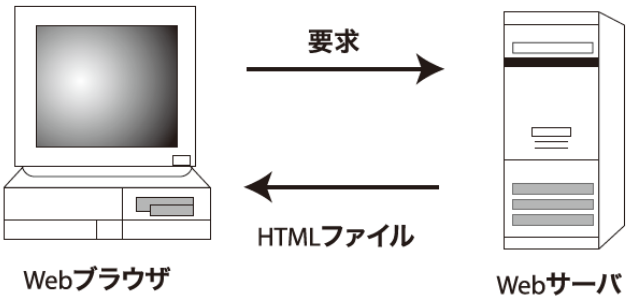
Web ページを公開するためには Web サーバが必要です。皆さんがこれから 3 回にかけて学ぶのは Web サーバの構築方法です。ここでは Web サーバプログラムとして大きなシェアを持つ Apache を利用します。

Apache に代表されるサーバプログラムの多くはデーモンプログラムとよばれます。一般的なプログラムがユーザーによって起動され処理を実行すると終了するのに対して、デーモンプログラムは常に起動された状態でユーザーからの要求に備えて待機し、要求があるとそれに応じた処理を行います。デーモンとよばれるのは、この働きが守り神（daemon）のようであることに由来しています。Apache の基本的な仕組みはいたってシンプルです。設定ファイル（httpd.conf）で指定したディレクトリに、Web ページのデータファイル（HTML ファイルなど）を置いておきます。クライアントから要求があると、Apache は要求に該当するファイルをこのディレクトリの中から探し、そのファイルデータをクライアントに送ります。

11.1 Web の仕組みと Web サーバ

11.1.1 Web の仕組み

私たちは普段、IE, Chrome, Firefox, Safari などの Web ブラウザを用いて、インターネット上の Web ページを閲覧しています。Web ページの多くは HTML（HyperText Markup Language）と呼ばれる言語で書かれたファイルです。Web ブラウザはこれらの HTML ファイルをダウンロードして、解釈し、表示や描画を行います。HTML ファイルを受け取る側である Web ブラウザに対し、HTML ファイルを提供する側にあたるのが Web サーバです。

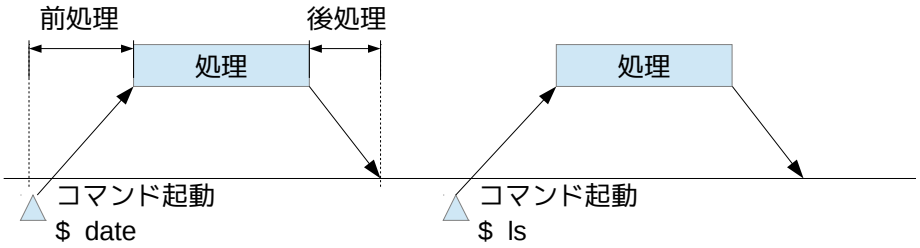


クライアントとサーバの通信においては、HTTP（HyperText Transfer Protocol）と呼ばれるプロトコルを用います。HTTP は TCP/IP の通信モデルでは、IP や TCP より上位のアプリケーション層で定義されます。

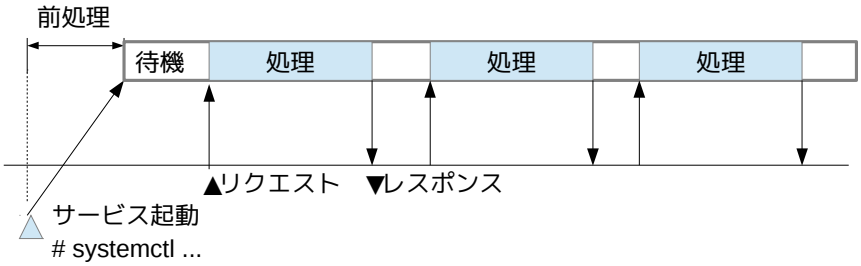
11.1.2　デーモンプログラム

Apache などの Web サーバに限らず、サーバソフトは一般にデーモン（daemon）と呼ばれる※1プロセスです。デーモンプロセスは、バックグラウンドで常駐稼働（OS 起動のち自動起動）します。通常のプログラムはユーザーによって起動され処理を行い、処理を終えると終了します。しかし、デーモンプロセスは起動するとリクエストの有無に関わらず起動し続け、ユーザーからの要求があると直ちに処理を行います。処理を終えてもそのまま起動し続け次の要求を待ちます。

通常のプロセス



デーモンプロセス



※1 デーモンは守り神を表す daemon の日本語読みです。demon（悪魔）ではありません。

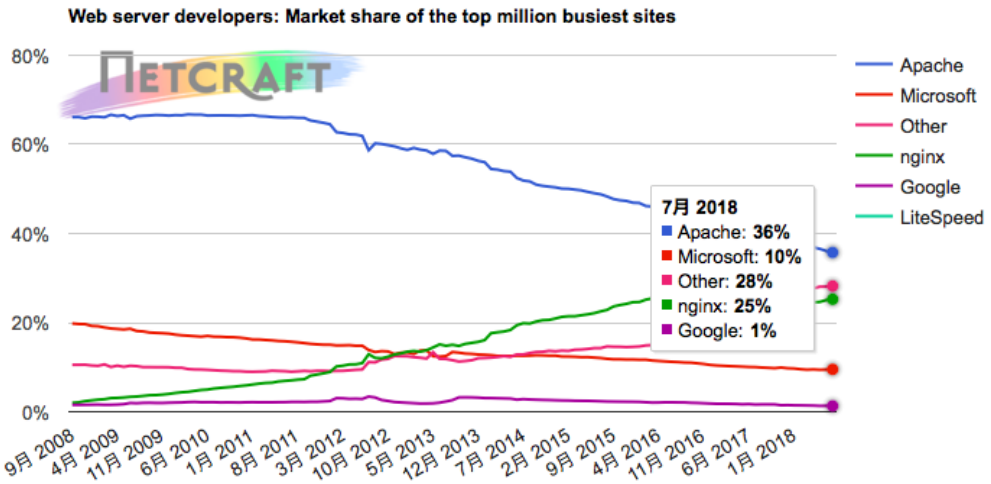
11.1.3 サーバと TCP ポート

サーバマシン上では、Web 以外にも様々なサービスが稼働しています。クライアントからの要求をサーバーが、個々のサービスごとに振り分けるためにポートと呼ばれる仕組みが用意されています。IP プロトコルは、データを宛先のコンピュータに届けるまでの処理を行いますが、どのサービスに接続するかは関知しません。どのサービスと接続するかは、TCP プロトコルが決定します。ポートには 0 から 65535 までの番号あり、この値を指定することでサービス決定します。特にサーバ側で待ち構えているポート番号はウェルノウン・ポート（well-known port）と呼び、0～1023 の範囲※2が割り当てられています。

例えば、Web サーバが使用するのは 80 番ポート、メールサーバは 25 番、名前解決は 53 番が割り当てられていて、この値は /etc/services に記載されています。

11.1.4 Web サーバ

インターネットの調査会社である Netcraft 社が発表した Web サーバーのシェアによると、Apache HTTP サーバは 36%となっています。また高速かつ膨大なリクエストにも耐える大規模向けサーバ Nginx（エンジン X）もシェアを伸ばしています。LPIC 試験（Level2）では、Apache と Nginx が試験範囲に含まれています。



Apache HTTP サーバや Nginx は複数の OS 上で動作しますが、Microsoft IIS（Internet Information Services）は Windows 系 OS に限定されます。

Web サーバの基本的な機能は、あらかじめ用意されたコンテンツ（HTML や画像データなど）を要求に応じ提供することですが、プログラムを動作させ要求に応じてコンテンツを生成する機能も搭載されています。前者を「静的なページ（Static pages）」と呼び、後者を「動的なページ（Dynamic pages）」と呼びます。

この「動的なページ」によって、オンラインショッピングやオンライントレード、動画ストリーミングなどが実現されています。

11.1.5 HTTP

※2 各アプリケーションに割当てられたポートについては、http://www.iana.org/assignments/port-numbers を参照のこと。

ブラウザと Web サーバの間の通信は、HTTP というプロトコルに基づいて行われています。ブラウザから次のような要求を送ることで、Web サーバが適切な処理を行います。このような要求のフォーマットをリクエストラインとよびます。

```
メソッド　　リクエスト URL　　HTTP バージョン
```

ある Web サーバから/index.html の情報を得るためには、以下のリクエストを送ります。

```
GET    /index.html    HTTP/1.0
```

ブラウから http://www.yahoo.co.jp にアクセスする場合、サーバ www.yahoo.co.jp に対して上記のリクエストが送信されます。主なメソッドは以下の通り。

メソッド	説明
HEAD	指定した URL に係るヘッダ情報（サーバーや、コンテンツの情報）を取得する
GET	指定した URL が表すリソース（HTML ファイルや画像ファイルなど）を取得する
POST	指定した URL が表すサーバに対してデータを転送する

[練習]

1. httpd が動いてなければ起動し、www.linuxacademy.ne.jp の 80 番ポートに telnet を用いて接続します。

```
$ telnet www.linuxacademy.ne.jp 80
```

2. トップページを取得します。次の行を入力後、Enter キーを 2 回入力します※5。

```
HEAD / HTTP/1.0      [Enter]を 2 回

```

※5 HTTP1.1 では加えて「Host:ホスト名」ヘッダが必須となる。

Web サーバは、リクエストラインを受け取ると適切な処理を行い応答します。応答の 1 行目に記述されているステータスラインで結果の概要を確認することができます。ステータスラインは、

```
HTTP のバージョン　ステータスコード　結果
```

となっています。ステータスコードは、要求が適切に処理されたかどうかを知る上でとても重要なものです。ステータスコードを見ると、処理が正常に完了したか失敗してしまったかなどを確認することができます。

HTTP ステータスコード

コード	説明
1XX	インフォメーション
2XX	正常終了
3XX	リダイレクト（処理の変更）
4XX	クライアントエラー
5XX	サーバエラー

11.2 Apache の起動と基本的な利用

11.2.1 Apache のインストール

CentOS 7 では、Apache 2.4 が採用されています、必要に応じ yum でインストールしてください。またはサーバソフトウエアはセキュリティの観点から常に最新バージョンに保つよう心掛けてください。

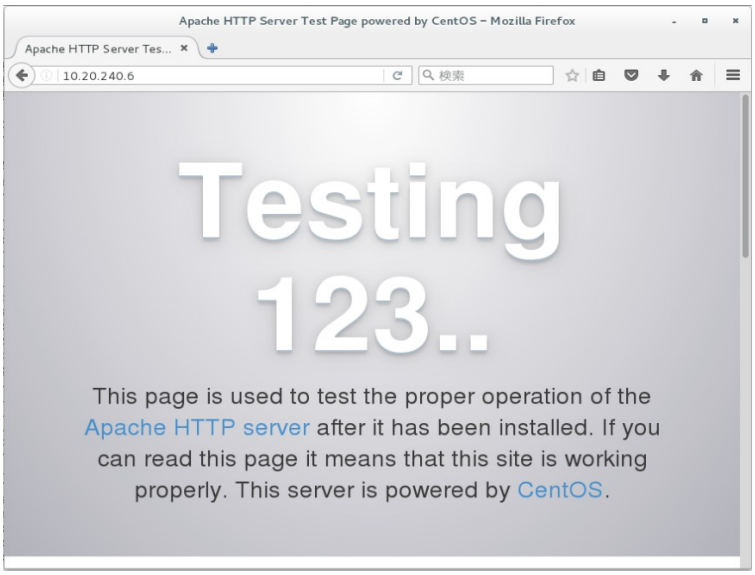
```
# yum update httpd
```

11.2.2 Apache の起動

Apache の起動は systemctl コマンドにより行います。特に問題なければ何も表示されません。

```
# systemctl start httpd
```

起動したら Web ブラウザ（Firefox）を使って、自分のマシンの IP アドレスを指定し接続します。以下のようなテストページが表示されます。



ファイアウォールが動作している場合は、外部からのアクセスを拒否してしまうためテストページが表示されません。うまく接続できない時は firewalld を停止してください。また当面は firewalld が自動起動しないようにしておきます。

```
# systemctl stop firewalld
# systemctl disable firewalld
```

11.2.3 Apache の実行プロセス

ps コマンドを使って、Apache の動作を確認します。

例

```
# ps aux | grep httpd
root   1913  0.0  0.5 230376  5172 ?  Ss  16:43  0:00 /usr/sbin/httpd -DFOREGROUND
apache 1914  0.0  0.3 230512  3708 ?  S   16:43  0:00 /usr/sbin/httpd -DFOREGROUND
. . .
```

プロセスには所有ユーザーが定義されています。プロセスは所有ユーザーの権限でプログラムが実行しています。上の ps コマンドの結果から、root ユーザーで実行されている httpd プログラムが 1 つと、複数の apache ユーザーで実行されているプロセスが※9あります。

pstree コマンドを用いると、この 9 つのプログラムの関係を見ることができます。

```
$ pstree
systemd─┬─NetworkManager─┬─2*[dhclient]
        │                └─2*[{NetworkManager}]
        ├─agetty
        ├─auditd───{auditd}
        ├─chronyd
        ├─crond
        ├─dbus-daemon
        ├─httpd───6*[httpd]
```

上の結果から、1 つの httpd プロセスから 6 つの httpd プロセスが生成されていることがわかります。生成されたプロセスを子プロセス、生成したプロセスを親プロセスとよびます。httpd の親プロセスは root 権限で動作している httpd プロセスです。

Apache では、子プロセスがクライアントへの応答処理を行い、親プロセスは子プロセスの管理などを行っています。親プロセスはクライアントからの要求を受け付ける 80 番ポートを管理し、要求があるたびにその応答処理を子プロセスに引き継ぎます。

※9 Apache のプロセス管理には、いくつかの方法がありますが、デフォルトは1つの親プロセスと複数の子プロセスで構成される prefork とよばれるものです。

11.2.4 ドキュメントの配置

Apache はローカルファイルシステムにあるディレクトリの一部をインターネット上に公開します。「http://10.20.142.6/」などの IP アドレスをブラウザに入力すると、ブラウザから Apache に対して、HTTP 要求メッセージが送られます。クライアントからの要求を受け取ると、Apache はローカルファイルシステムの特定のディレクトリの中の HTML ファイルをクライアントに送り返します。デフォルトの状態では、/var/www/html/ディレクトリ以下にあるファイルやディレクトリが公開されます。

Web サーバが公開するディレクトリの最上位（ルートに相当）ディレクトリを「ドキュメントルートディレクトリ」と呼びます。このディレクトリは、クライアントがサーバ名のみを指定した場合にアクセスされるディレクトリです。ここでは/var/www/html がドキュメントルートディレクトリ※10になります。

また、Apache のデフォルト設定では、ファイル名が指定されていないアクセスにはディレクトリ内の index.html ファイルが送られるようになっています。Apache のドキュメントルートディレクトリは、後述の Apache 設定ファイル httpd.conf において設定されます。

[練習]

1. 起動した Apache にアクセスします。

```
http://127.0.0.1/
```

2. root ユーザーになります。
3. 次の HTML が記述されたファイル、/var/www/html/index.html を作成します。

```
<html>
<body>Hello,World!</body>
</html>
```

4. Web サーバに再度アクセスして、表示されるページが変更されたことを確認します。

※10/var/www/html 以下に index.html がない場合は/usr/share/httpd/noindex/index.html が表示されるように設計されています。

column

ユーザの公開するWebページ

/var/www/html ディレクトリは所有者が root であり、一般ユーザには書き込み権限が与えられていません。このため、一般ユーザは HTML ファイルを/var/www/html ディレクトリ内に置くことはできません。

```
$ ls -l /var/www
合計 0
drwxr-xr-x 2 root root 6  4月  13 06:04 cgi-bin
drwxr-xr-x 2 root root 6  4月  13 06:04 html
```

そこで、一般ユーザーも Web ページを公開できるように「ユーザーディレクトリ機能」というものが用意されています。これはホームディレクトリ下にサブディレクトリ public_html を作成し、ここを Web ページとして公開できるようにするものです。このディレクトリ内に置いたファイルは、「http://127.0.0.1/~ユーザー名/ファイル名」としてアクセスできるようになります。「http://127.0.0.1/~ユーザー名」とすると、ユーザーのホームディレクトリ以下の public_html ディレクトリ内にある index.html が参照されます。

このホームユーザディレクトリ機能は、CentOS に含まれている httpd のパッケージではデフォルトで無効になっています。そのため/etc/httpd/conf.d/userdir.conf の該当箇所を以下のように変更し、Apache を再起動する必要があります。

```
<IfModule mod_userdir.c>
    #
    # UserDir is disabled by default since it can confirm the
presence
    # of a username on the system (depending on home
directory
    # permissions).
    #
#    UserDir disabled  ←先頭に#を入れ、無効にする。
    #
    # To enable requests to /~user/ to serve the user's
public_html
    # directory, remove the "UserDir disabled" line above,
and uncomment
    # the following line instead:
    #
    UserDir public_html←先頭の#を削除し、有効にする。
</IfModule>
```

次に、ユーザー student になってホームディレクトリ以下に public_html ディレクトリを作成し、その下に index.html という HTML ファイルを作成します。

```
$ cd ~
$ mkdir public_html
$ cd public_html
$ vi index.html
```

index.html

```
<html>
<body>
Here is my web page!<br>
This page is /home/student/public_html/index.html.
</body>
</html>
```

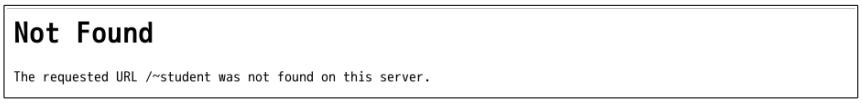
ブラウザからこの HTML ファイルにアクセスします。

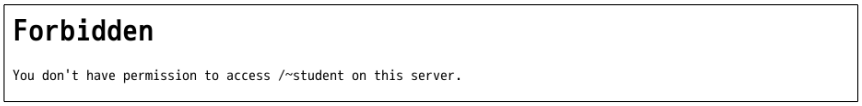
```
http://127.0.0.1/~student/
```

ブラウザに「Forbidden」と表示され、目的の HTML ファイルにアクセスできません。
これは apache ユーザーが public_html ディレクトリ以下のファイルにアクセス権がないためです。
今回の場合は、/home/student ディレクトリが第三者である apache に対して実行権を許していないため Forbidden メッセージが表示されたのです。student ユーザーのホームディレクトリのパーミッションを適切に直し、再度アクセスします。

```
$ chmod 701 /home/student/
```

「Not Found」は実際にファイルが存在しない場合で、「Forbidden」はファイルがあるが、権限がなく表示できないことを表します。





Apache のログファイル

Apache HTTP は以下の2つのログ（動作記録）を生成します。

```
/var/log/httpd/access_log
/var/log/httpd/error_log
```

access_log はクライアントからの全ての要求を記録しており、error_log はサーバー内で発生したエラーを記録しています。ログファイルの保存先は、CustomLog, ErrorLog ディレクティブにより、指定されます。
ログはファイルに追記され、下にいくほど新しい記録になります。随時ログをチェックするためには、tail コマンドを用いるのが良いでしょう。ログファイルはどんどん増えていくため、定期的にバックアップを取得し古いものは削除します。

例

access_log

```
192.168.56.1 - - [11/Aug/2018:16:53:22 +0900] "GET /cgi-bin/sample.cgi
HTTP/1.1" 200 68 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6)
AppleWebKit/605.1.15 (KHTML, like Gecko) Version/11.1.2 Safari/605.1.15”
127.0.0.1 - - [11/Aug/2018:17:06:27 +0900] "GET /cgi-bin/count.cgi
HTTP/1.1" 200 2 "-" "Wget/1.14 (linux-gnu)"
```

アクセスしてきたクライアントの IP アドレス、日時、リクエスト内容、ブラウザの種別などが記録されます。

error_log

```
[Sat Aug 11 16:43:49.581328 2018] [core:notice] [pid 1913] AH00094:
Command line: '/usr/sbin/httpd -D FOREGROUND'
[Sat Aug 11 20:43:42.141073 2018] [access_compat:error] [pid 1918]
[client ::1:33590] AH01797: client denied by server configuration:
/var/www/html/uso800
```

ファイルへのアクセス権不正や、設定内容のエラーなどが記録されます。

不正アクセスの例として、CodeRed ウィルス、Nimda ウィルスに感染したマシンからのアクセスを記録したログの例を挙げます。

CodeRed ウィルスからのアクセスの例 [access_log]

```
172.16.0.210- - [10/Jan/2002:03:25:16 +0900 "Get /default.ida?NNNNNN（中
略）%u00=a HTTP/1.0" 400 322
```

Nimda ウィルスからのアクセスの例 [error_log]

```
[Wed Jan 9 17:31:03 2002] [error] [client 172.16.0.210] File does not
exist:/var/www/html/_vti_bin/..%5C../..%5C../..
%5C../winnt/system32/cmd.exe
```

12. ApacheHTTP サーバ（2）

12

ApacheHTTP サーバ（2）

本章のねらい

- Apache の設定方法を学ぶ
- 基本認証の考え方と設定を学ぶ

予習編

アクセス制限と基本認証

Web ページを公開するとき、特定の人のみに公開したい、あるいは一部の人には公開したくないということがあります。そのときに使うのが「アクセス（接続）制限」です。アクセス制限をかけると、特定の IP アドレスからの接続を拒否することができます。一部の人にのみ Web ページを見せたい場合や、悪意ある者からのアクセスを拒否したい場合などに有効です。

アクセス制限を用いると、特定のマシンやネットワークからの接続を拒否することができます。しかし、これを用いても、相手がマシンやネットワークを変えればアクセスが可能となります。

この様な場合は、Web ページにアクセスするためにユーザー名とパスワードを求めることで、特定の人からのアクセスのみを受け付けるように制限することができます。この仕組みが認証システムです。認証システムはプロバイダのメンバー専用ページなどでも利用されています。あらかじめユーザー名とパスワードを登録して、アクセス時にそれを照合するという最も簡単な認証方式を基本認証と呼びます。それほど機密性の高くない情報であれば、基本認証を使って保護すれば十分といえます。

12.1 Apache の設定

12.1.1 Apache 設定ファイル

主要な設定は/etc/httpd/conf/httpd.conf で行います。それ以外にも機能ごとに小分けされた設定ファイルが /etc/httpd/conf.d/、/etc/httpd/conf.modules.d/の下にあります。

httpd.conf の各項目の書式は、

```
#   コメント
設定名      設定値
```

のように、設定名と設定値を空白で区切って記述します。設定名はディレクティブと呼ばれます。また、複数のディレクティブを設定するために、< >で囲んだタグと呼ばれるものもあります。

```
<タグ名     設定対象>
     設定名１      設定値１
     設定名２      設定値２
     ...
</タグ名>
```

設定ファイルの文法チェックは apachectl コマンドで行います。

```
エラーがある場合
# apachectl -t
AH00526: Syntax error on line 42 of /etc/httpd/conf/httpd.conf:
Port must be specified
正常な場合
# apachectl -t
Syntax OK
```

httpd.conf を修正した場合は、

```
# systemctl restart httpd
```

として、Apache を再起動※1する必要があります。

※1 Apache だけでなく多くのサーバソフトでも、設定変更を反映にするにはサービスの再起動が必要です。

12.1.2 Apache の基本設定

httpd.conf の主要な設定項目は以下のものがあります。

#	ディレクティブ	初期値	意味
1	ServerRoot	/etc/httpd	設定ファイルなどが置かれる頂点となるディレクトリ
2	Listen	80	リクエストを受け付ける TCP ポート番号
3	User	apache	子プロセスの所有ユーザー
4	Group	apache	子プロセスの所有ユーザーの属するグループ
5	ServerAdmin	root@localhost	Web サーバの管理者への連絡先(e-mail アドレス)
6	ServerName	なし	Web サーバのホスト名
7	DocumentRoot	/var/www/html	ドキュメントルートディレクトリ
8	DirectoryIndex	index.html	ここで指定したページがディレクトリ内のデフォルトのトップページになる。http://127.0.0.1/のようにファイル名を省略してブラウザでアクセスした場合、表示されるのはここで指定されたファイルになる。この設定はサブディレクトリにも引き継がれる。

[練習]

1. root ユーザーになります。
2. Listen ディレクティブで指定するポート番号を 80 から 8080 に変更します。
   httpd.conf を編集[2]後 Apache を再起動し、ブラウザから下記にアクセスします。

   ```
   http://自身の IP アドレス:8080/
   ```

3. Listen ディレクティブのポート番号を 80 に戻し、Apache を再起動します。

※2 httpd.conf は非常に大きいので「/検索文字列[Enter]」を使い、Listen を探すと良いでしょう。

12.1.3 <Directory>タグ

<Directory>タグは以下のように</Directory>タグと対になって使用されます。

```
<Directory  ディレクトリ>
    ディレクティブを記述
</Directory>
```

<Directory>タグは特定のディレクトリ内のみで有効になる設定を記述するためのものです。デフォルトの httpd.conf の中には

```
<Directory "/var/www/html">
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

という箇所があり、ドキュメントルートディレクトリの設定がされています。<Directory>タグ内で用いられる主なディレクティブは次の通りです。

ディレクティブ	説明
Options [option]	以下のオプションにより追加機能を制御する。 ・ExecCGI　CGI の実行許可 ・FllowSymLinks　シンボリックリンクを有効にする ・Includes　SSI 実行許可 ・Indexes　ファイル一覧表示許可 ・All　上記全てを許可（規定値） ・None　上記全てを禁止
AccessFileName	アクセスコントロールファイルのファイル名を指定する。
AllowOverride	アクセスコントロールファイルの設定変更が可能な範囲を指定する。
Order	Web サーバへのアクセスを許可（allow）、拒否（deny）リストの検索順序を設定する。 ・allow,deny は、デフォルトは拒否で、例外を Allow で定義 ・deny,allow は、デフォルトは許可で、例外を Deny に定義
Allow	from に続けて、アクセス許可するホストを指定する。 ホストの指定には、以下の書式が利用できる。 ・all（全てのホスト） ・.example.com のようなドメイン ・IP アドレス ・ネットワークアドレス/ネットマスク
Deny	Allow と同様に、アクセス拒否するホストを指定する。

[練習]

1. Order, Allow, Deny ディレクティブを利用して、自分のホスト以外からは Web サーバにアクセスできないようにします。<Directory "/var/www/html">タグ内の設定を次のように変更します。Allow from で指定する IP アドレスは、自分のホストに割り当てられているものを記述します。

```
Order deny,allow
Deny from all
Allow from 10.20.142.6
```

2. ブラウザから次の URL にアクセスします。

```
http://10.20.142.6/index.html
```

また、隣の方のホストからアクセスした時、どのようになるか確認します。

3. 確認できたら、設定を元※3に戻します。

※3 設定を変更する場合は、変更前のファイルのバックアップを作成するように心掛けます。

12.2 基本認証

12.2.1 基本認証の概要

インターネット上にはユーザー名とパスワードの入力を求める Web ページがあります。これは特定のユーザーにのみ Web ページを公開するための仕組みです。逆にいうと、パスワードを知らないユーザーのアクセスを拒否する仕組みともいえます。最も基本的なユーザー認証は、ユーザー名と暗号化したパスワードを記述したファイルを用意することで実現できます。この認証方法を基本認証（basic authentication）と呼びます。

12.2.2 基本認証の設定

基本認証はディレクトリ単位で設定できます。あるディレクトリに対して、基本認証が設定されていれば、そのサブディレクトリに対しても認証を行う必要が生じます。

ディレクトリに対する基本認証の設定は、httpd.conf 内の<Directory>タグ内で行います。基本認証を行うディレクトリを<Directory>タグで指定し、以下のような設定をします。

```
<Directory  ”ディレクトリ”>
    AuthType                Basic
    AuthName                ”認証証名（認証時に表示される）”
    AuthBasicProvider       認証の手段
    AuthUserFile                    パスワードファイルのパス名
    Require user            認証対象となるユーザ名
</Directory>
```

ドキュメントルートの下に staff サブディレクトリを作成し、認証対象とするには以下のようにします。

```
<Directory  ”/var/www/html/staff”>
    AuthType                Basic
    AuthName                ”Private area”
    AuthBasicProvider       file
    AuthUserFile            /var/www/html/staff/.htpasswd
    Require user            staff
</Directory>
```

12.2.3　パスワードファイルの作成

htpasswd コマンドを用いて、AuthUserFile ディレクティブで指定したパスワードファイルを作成します。

```
htpasswd -c [パスワードファイルへのパス] [ユーザー名]
```

-c オプションを付けて実行するとパスワードファイルが新規に作成され、指定したユーザーとパスワードが登録されます。既存ファイルに対し-c を指定すると上書きされ直前の内容は失われます。

例

```
# htpasswd -c /var/www/html/staff/.htpasswd staff
New password: (パスワードの入力)
Re-type new password: (パスワードの入力)
Adding password for user staff
```

基本認証とシステムのユーザ（/etc/passwd の内容）とは無関係です。htpasswd コマンドで、システムのユーザーとは異なる管理を行うことができます。

[練習]

1. /var/www/html/staff ディレクトリを作成し、ディレクトリ内に次の index.html を準備します。

```
<html>
<body>
    staff only!
</body>
</html>
```

2. /var/www/html/staff ディレクトリにアクセスを行うと認証が要求されるように httpd.conf を設定します。認証を要求するユーザーは staff、パスワードは staff とします。httpd.conf を編集した場合は、Apache を再起動するのを忘れないようにします。

3. 自分のマシンに対する基本認証が確認できたら、隣の方に確かめてもらいます。

column

.htaccess ファイル

Apache の設定は基本的に httpd.conf で行います。しかし、httpd.conf は root ユーザー以外は編集することができないため、各ユーザーは自分のユーザーホームディレクトリに関する設定を各ユーザーが変更することができません。このような場合は、アクセスコントロールファイルを利用します。
アクセスコントロールファイルは、設定を記述したテキストファイルです。ファイル名は httpd.conf 内の AccessFileName ディレクティブで指定されたものを用い、デフォルトでは.htaccess という名前になります。

アクセスコントロールファイルに、httpd.conf の<Directory>タグ内で指定できるディレクティブを記述して、各ユーザーが任意のディレクトリ（通常は自身のホームディレクトリ以下）に配置することにより、記述した設定がそのディレクトリ以下で適用されます。つまり、httpd.conf の<Directory>タグ内における設定と同様の設定を root 権限なしで実現できます。

但し、.htaccess を利用するには、/etc/httpd/conf.d/userdir.conf 内で「AllowOverride AuthConfig」もしくは「AllowOverride All」などを設定し、アクセスコントロールファイルによる設定変更を可能にする必要があります（デフォルトで設定済み）。

例
各ユーザーの~/public_html 下でアクセスコントロールファイルによる上書きを許可する場合

```
<Directory “/home/*/public_html>”
    AllowOverride AuthConfig
</Directory>
```

student ユーザーのユーザーホームディレクトリに関する設定を変更してみます。student ユーザーになり、下記の.htaccess ファイルを/home/student/public_html ディレクトリの中に配置します。

.htaccess ファイルの設定例

```
Order allow,deny
Allow from all
Deny from 172.16.0.210
```

これにより、IP アドレスが 172.16.0.210 であるホストから、student ユーザーのユーザーホームディレクトリに対するアクセスが制限されます。

また、本章で解説した「基本認証」も.htaccess ファイルに記述することで実現できます。例えば、secret というユーザー名で認証を求める設定は以下のようになります。

```
AuthType            Basic
AuthName            “Secret file”
AuthBasicProvider   file
AuthUserFile        /
                    home/student/public_html_.htpassw
                    d
Require user        secret
```

htpasswd コマンドで基本認証用のパスワードファイルを作成します。

```
$ htpasswd -c /home/student/public_html/.htpasswd secret
New password:（パスワードの入力）
Re-type new password:（パスワードの入力）
Adding password for user secret
```

これにより、student ユーザーのユーザーホームディレクトリへのアクセスは認証が必要※5になります。

※5 httpd.conf における設定変更を反映させるには httpd サービスの再起動が必要でしたが、.htaccess ファイルによる設定の反映にはサービスの再起動は必要ありません。

Apache の子プロセス管理

Apache では、親プロセスが子プロセスを生成し、その子プロセスが Web ブラウザからの要求に応えています。子プロセスを複数生成し処理を同時並行的に行うことで、複数のリクエストに効率よく応答するのです。したがって、大量のアクセスに滞りなく応じるには、Apach の子プロセスの管理が重要になります。

Apache を standalone モードで起動すると、親プロセスが常駐します。クライアント（Web ブラウザ）からリクエストが発せられると、親プロセスは子プロセスを生成（fork）し、その処理を子プロセスに担当させます。しかし、この子プロセスの生成には時間がかかるため効率よく管理する必要があります。

子プロセスの制御を行うための主なディレクティブは、以下の通りです。

- StartServers
- MinSpareServers
- MaxSpareServers
- MaxClients
- MaxRequestsPerChild

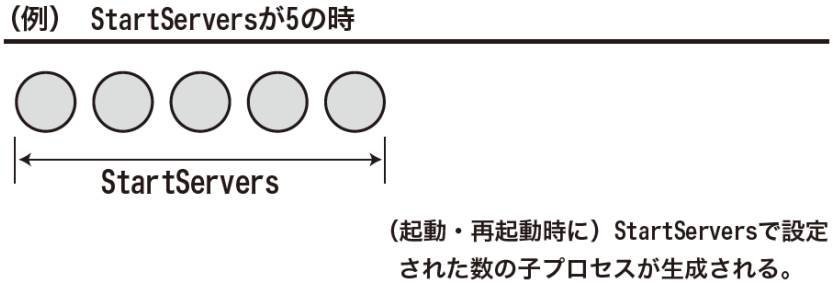
・StartServers

Apache の起動時にあらかじめ子プロセスを生成し、待機させておくことでクライアントからのレスポンスに素早く対応することができます。アクセス数が非常に多いことが見込まれる場合は、最初から多くの子プロセスを生成しておきます。

起動時に常駐させる子プロセスの数を指定するのが、StartServers ディレクティブです。8 個の子プロセスを生成させる例は以下の通りです。

```
StartServers  8
```

但し、この設定は起動時の子プロセス数を定義するもので、起動して時間が経過すると、MinSpareServers や MaxClients ディレクティブの値が、より子プロセス数に影響を与えるようになります。再起動をあまり行場合は、StartServers はさほど気にする必要はありません。



・MinSpareServers

すべての子プロセスがクライアントからの要求を処理している（待ち状態のプロセスが存在しない）時に、リクエストが追加されると、親プロセスは新たに子プロセスを生成します。しかし、リクエストを受けてから子プロセスを生成していては時間がかかりすぎてしまいます。

そこで、待ち状態の子プロセスを確保するために用いるのが MinSpareServers ディレクティブです。MinSpareServers は待機する子プロセスの最小値を設定します。例えば、常に 5 個以上のリクエスト待ち状態の子プロセスを常駐したい時は、

```
MinSpareServers  5
```

と記述します。このように記述しておくと、待機状態の子プロセスが 4 つ以下に減少したら、新たに子プロセスを生成するようになります。

・MaxClients

子プロセスを数多く生成し、それらを常駐させておけば、性能面で非常に有利になるはずです。しかし、多くのプロセスを常駐させると、サーバのリソース（メモリや CPU などのハードウェア資源）がその分だけ多く消費されます。実際、多くのプロセスを常駐させすぎると実メモリが不足してしまいます。実メモリが不足してしまうと、ハードディスクにデータを待避させる（スワップ）処理が発生するため、逆にパフォーマンスが著しく低下します。

それを防ぐため、MaxClients ディレクティブを用いて常駐するプロセスの上限を設定します。同時に常駐できる子プロセスを 150 個に制限したい場合は、以下のようにします。

```
MaxClients  150
```

この値をあまり小さな値にすると、同時に処理できるリクエスト数が少なくなり、リクエストが待ち状態になりやすくなります（レスポンスの低下）。中規模・大規模の Web サーバを設定する場合は、ある程度大きな値を設定する必要があります。

・MaxSpareServers

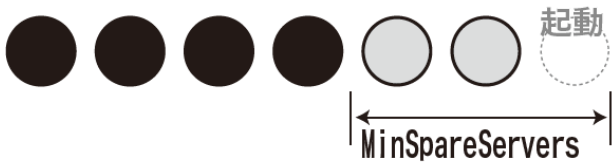
リクエストを処理し終えた子プロセスは、再びリクエスト待ち状態になり常駐します。同時に大量のアクセスを処理した後は、待ち状態の子プロセスが多く常駐することになります。しかし、多くの子プロセスを常駐させておくことはリソースの浪費につながるため、待機することのできる子プロセスの数を制限する必要があります。そのために用いられるのが、MaxSpareServers ディレクティブです。

MaxSpareServers の値を超える待機プロセスは停止されます。例えば、リクエスト待ち子プロセスを 20 個までに制限したい時は、以下のようにします。

```
MaxSpareServers  20
```

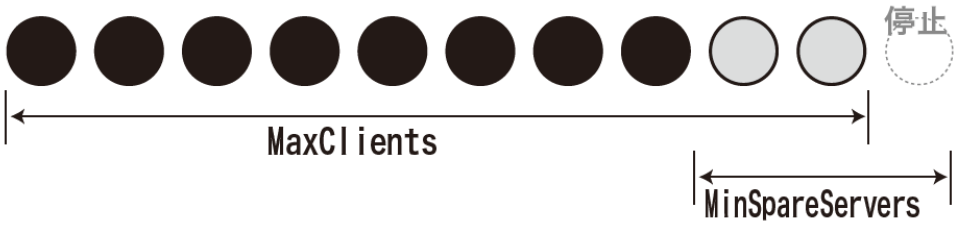
（例）MinSpareServers 3、MaxClients 10、MaxSpareServers 5の時

[ケースⅠ]



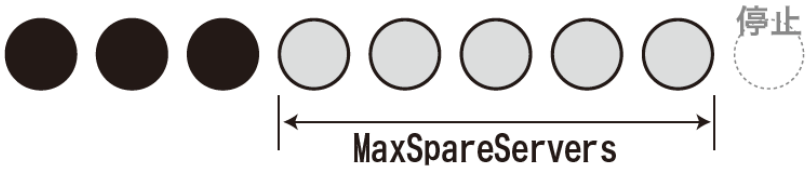
待機子プロセスがMinSpareServers以下に
なったら、子プロセスが生成される。

[ケースⅡ]

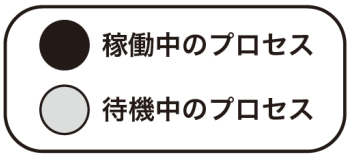


子プロセスの数はMaxClientsを越えることは
できない。

[ケースⅢ]



リクエスト待ちの子プロセスの数は
MaxSpareServersを越えることはできない。



・MaxRequestsPerChild

子プロセスはリクエスト処理をした後、リクエスト待ち状態で常駐します。再びリクエストを受け付けるとまた処理状態になり、この動作を繰り返します。すなわち、同じプロセスが待機・動作を繰り返すわけです。

しかし、同一プロセスを長期間利用すると、メモリを大量に消費するなど異常動作をする可能性が高くなります。そこで子プロセスのリクエスト処理回数の上限を決め、それに達したプロセスは一旦終了するよう設定することができます。この処理回数を設定するのが、MaxRequestsPerChild ディレクティブです。例えば「1000 回リクエスト処理した子プロセスは終了する」という設定は、以下のように記述します。

```
MaxRequestsPerChild  1000
```

13. ApacheHTTP サーバ（3）

13

ApacheHTTP サーバ（3）

本章のねらい

- Apache のモジュール構造を学ぶ
- CGI の仕組みと設定方法について学ぶ

予習編

CGI

閲覧者の要求に応じて変化するWebページが作れるとWebの活用範囲が大きく広がります。実際、インターネット上でよく使われている「掲示板」は、閲覧者の書き込みに応じてWebページが変化します。これを実現する仕組みの1つが、CGI（Common Gateway Interface）です。閲覧者の要求に応じてサーバ内に予め用意しておいたプログラムやスクリプトを実行し、その実行結果をWebページとして出力しているのです。

CGIを使うには、実行するプログラムを準備し、その上でhttpd.confでCGIが利用できるように設定します。CGIを使うと閲覧者からのアクセスに応じてこれらのプログラムが実行され、その結果がWebページになります。

CGIは非常に便利で、無料で公開されているものも多数存在することから手軽に使うことができます。CGIプログラムにアクセスすれば、誰でもサーバ内のプログラムを実行することができます。これはアカウントを持たないユーザーによってプログラムがみだりに実行される可能性があるため大変危険です。実際、CGIを悪用したクラッキングの事例は数多く存在します。そのため、CGIを使うためにはhttpd.confで適切な設定を行う必要があります。多くの場合は、特定のディレクトリでのみCGIが利用できるようにします。

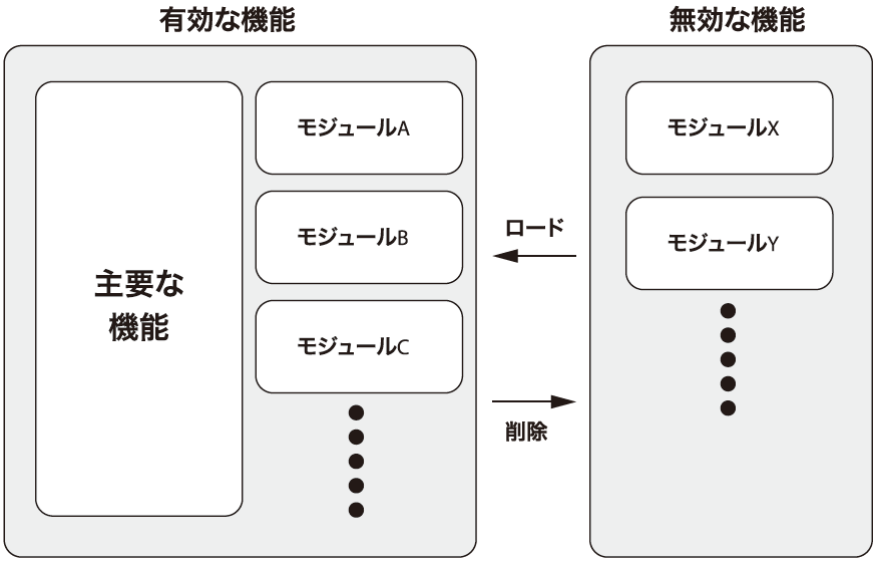
13.1 Apache の機能拡張

13.1.1 モジュールとは

Web サーバの主な機能は、HTTP ポート（通常は TCP ポート 80 番）を監視し、クライアントからの要求に応じて HTML ファイルや画像などのファイルを送信するものです。Apache はこの最も基本的な機能に加え、ディレクトリごとのアクセス制限、基本認証などの多くの重要な機能を提供しています。これらの機能はモジュール（module）と呼ばれるパーツに分割して提供されています。

モジュールを利用することで、Apache 本体に特定の機能を追加（ロード）したり、削除したりできます。

モジュールを使用した機能の有効化・無効化



モジュールを用いて機能を有効にしたり無効にしたりする仕組みは、Apache や Linux のカーネルなど様々なプログラムで用いられています。

13.1.2 Apache のモジュール

Apache は、必要最小限の機能を担う Core モジュールと追加機能を担う複数のモジュールから成り立っています。

Apache の機能はモジュールによって構成されているため、その設定を行うディレクティブはモジュールのロード状況によって利用できたりできなかったりします。各ディレクティブの設定を有効にするためには、その機能を提供するモジュールをロードする必要があります。

例えば、「http://www.linuxacademy.ne.jp/」などのようにファイル名が明示されていない要求に対しては、DirectoryIndex ディレクティブで指定したファイル（index.htm など）が読み込まれ、「http://www.linuxacademy.ne.jp/index.html」と要求した場合と同じように Web ページが表示されます。この機能を提供するのは mod_dir モジュールであるため、このモジュールがロードされずに Apache が起動された場合はこの機能は無効になります。

13.1.3 Apache モジュールの追加・削除の設定

モジュールをロードするかしないかは、/etc/httpd/conf.modules.d ディレクトリ内の各ファイルで設定します。モジュールをロードするには、LoadModule ディレクティブを用います。

```
LoadModule  [モジュールの識別名]  modules/[モジュールのファイル名]
```

コンパイル済みのモジュールは、/etc/httpd/modules ディレクトリ以下に配置されており、「.so」という拡張子が付けられています。LoadModule は各モジュールに付けられた識別名とモジュールへのパスを併せて、記述することにより設定します。モジュールへのパスは、ServerRoot ディレクティブで指定されるディレクトリを基準として指定します。

デフォルトでは、/etc/httpd ディレクトリが ServerRoot ディレクティブに指定されているので、「modules/mod_dir.so」などと指定します。

例

```
mod_dir モジュールのロードの指定（00-base.conf から抜粋）
                （中略）
LoadModule    dir_module      modules/mod_dir.so
```

モジュールに関する設定を変更した場合は、ファイルを保存し Apache を再起動する必要があります。

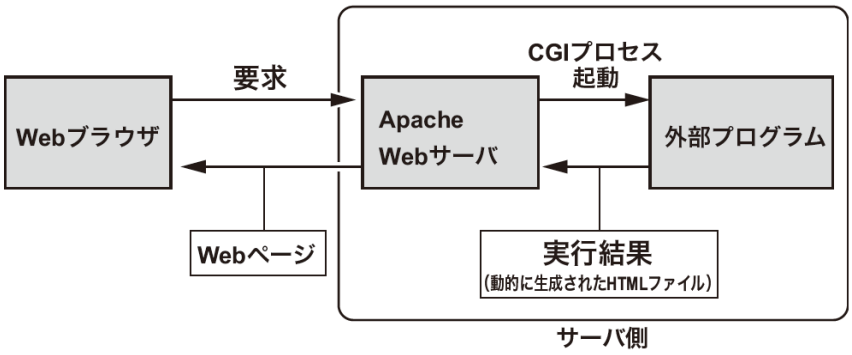
13.2 CGI

Web サーバのファイルシステムの中にあらかじめ配置されている Web ページは、静的な Web ページと呼ばれています。これは、ページを見るユーザーなどによらずユーザーのアクセスに応じて表示されるデータが同じだからです。

今日では、静的な Web サイトだけではなく、オンラインショッピングや検索エンジンなどに見られるように、動的に表示される（内容が変わる）Web ページも存在します。これらの動的なページを実現する方法として CGI（Common Gateway Interface）や SSI（Server-Side Include）などといった仕組みが用いられます。

13.2.1 CGI の仕組み

ブログや検索エンジンなど動的なページの多くは、CGI という機構を用いて実現されています。CGI は、シェルスクリプトや Perl スクリプト、C 言語などで作成されたバイナリプログラム（以下、CGI プログラムと呼びます）を Web サーバから起動し実行することで動的なページを生成する仕組みです。



CGI プログラムを記述するためのプログラム言語には特に指定はありません。CGI プログラムは、標準出力が httpd プロセスにリダイレクトされるため、標準出力に結果を返すことのできる言語であれば何でも利用できます。

13.2.2 CGI の利用設定

CGI を利用するためには、mod_cgi モジュールがロードされている必要があります。CentOS7 ではデフォルトでロードされています。また、次のように設定ファイル内で CGI に関する設定を行う必要があります。

[1] ScriptAlias ディレクティブを用いる

ScriptAlias ディレクティブで、下記のように指定します。

```
ScriptAlias  [要求先ディレクトリ]  [CGI プログラムを配置するディレクトリ]
```

例えば、デフォルトの httpd.conf においては、

```
ScriptAlias  /cgi-bin/  "/var/www/cgi-bin/"
```

という設定行が存在します。これは「http://…/cgi-bin/…」というディレクトリ内のファイルへの要求があった場合、ファイルシステムの中の/var/www/cgi-bin ディレクトリ内のファイルへのアクセスと見なされ、このディレクトリの中にあるファイルは全て CGI プログラムとして実行されることを表しています。一般的に root ユーザーなどの Web サーバ管理者のみがファイルを配置できるディレクトリに対してこの ScriptAlias ディレクティブの指定を行います。

[2] Options ディレクティブを用いる

ScriptAlias ディレクティブで指定したディレクトリ以外にあるディレクトリ内の特定のファイルを CGI プログラムとして実行させるには、Options ディレクティブを用いて指定します。どのファイルが CGI プログラムであるかは、AddHandler ディレクティブを用いて指定します。

例

```
<Directory  "ディレクトリ">
    Options +ExecCGI
    Order allow,deny
    Allow from all
</Directory>
    （中略）
AddHandler  CGI-script  .cgi
```

この例では、[ディレクトリ] より下位にある「～.cgi」というファイルは全て CGI プログラムとして実行されるように指定しています。※1

※1 Directory タグで指定した Options ディレクティブはそれ以下のディレクトリすべてに適用されます。よって下位のディレクトリで別の Options ディレクティブを指定した場合、複数の Options ディレクティブが適用可能となります。このような場合は最も近いもののみが適用され、他は無視されます。ただし、左の例のように「+」をつけることによってより上位のディレクトリにて指定されているディレクティブに追加して適用されます。詳しくは「http://httpd.apache.org/docs/2.4/mod/core.html#options」を参照してください。

[練習]

1. 前述の解説を参考にして、httpd.conf 内にて、/var/www/cgi-bin ディレクトリ内の CGI プログラムが実行可能になっていることを確認します。
2. 以下のシェルスクリプトで記述された CGI プログラムを実行します。これは、「Hello, World」と表示し、whoami コマンドを実行するものです。/var/www/cgi-bin ディレクトリの下に sample.cgi を作成したら、スクリプトに実行権と読み取り権を与えるのを忘れないようにします。CGI プログラムは一般的にパーミッションを 755（第三者に実行、読み取り権を与えるのが重要）に設定するのがよいとされています。

sample.cgi

```
#! /bin/bash
# Sample CGI
myname=`whoami`
cat <<EOD
Content-type: text/html

<html><body><p>Hello, world!</p>
I am $myname
</body>
</html>
EOD
```

3. ブラウザでテストしたあと、CGI で出力したページの HTML ソースを参照します。

CGI は、プログラミング言語を選びませんが、「Content-Type: text/html」[改行][改行]」や「Content-Type: text/plain[改行][改行]」などのヘッダを出力する必要があります。上のスクリプトでは、cat ヘヒアドキュメントを用いてヘッダを出力しています。

13.3 確認問題

次のスクリプトは、このページが参照された回数を表示させるプログラム（アクセスカウンタ）です。このプログラムを CGI として動かします。

```
#! /bin/bash
# Simple access counter
DAT=count.dat
echo "Content-type: text/plain"
echo ""

if [ -f $DAT ]
then
    cur=`cat $DAT`
else
    echo "Not found $DAT"
    exit 1
fi

new=`expr $cur + 1`
echo "$new" | tee count.dat
```

1. 上記のスクリプトを/var/www/cgi-bin/count.cgi として保存します。
2. このスクリプトが行っている処理を理解した上で、count.dat ファイルを準備します。
3. count.cgi および count.dat ファイルの所有者およびパーミッションなどを適切に設定します。
4. 必要であれば、httpd.conf の設定を変更し、Apache を再起動します。
5. count.cgi を Web ブラウザから参照し、正常に動作していることを確認します。

column

.htaccess ファイルを用いた CGI の設定

.htaccess ファイルを用いることで、一般ユーザーでもディレクトリ毎に CGI の実行を許可することができます。Options ディレクティブと AddHandler ディレクティブを.htaccess ファイルに記述することにより、CGI プログラムが実行可能になります※2。この際、.htaccess ファイルを配置するディレクトリに対し、設定ファイル内で AllowOverride ディレクティブが適切に設定され、.htaccess ファイルの利用が可能になっている必要があります。

例

```
<Directory [ディレクトリ]>
    AllowOverride Options
    （省略）
</Directory>
```

.htaccess

```
Options +ExecCGI
AddHandler CGI-script .cgi
```

.htaccess ファイルを使用して CGI の使用を許可した場合、以下のコマンドを root ユーザで実行している必要があります。

```
# mv /usr/sbin/suexec /usr/sbin/suexec.off
# systemctl restart httpd
```

これはセキュリティ対策を行うための、suEXEC 機構をオフにするためのコマンドです。

※2 student ディレクトリで作業するときには必ず student ユーザーで行います。

SSI（Server-Side Include）

SSI※3は Apache に組み込まれているインタープリタで、HTML ファイルに簡単な文字列を動的挿入する仕組みです。ファイルの最終更新日時やアクセスカウンタの表示などに利用できます。SSI を利用するためには、mod_include、mod_cgi モジュールが必要です。デフォルトでは、mod_include、mod_cgi モジュールも LoadModule ディレクティブでロードされるように設定されています。

SSI が有効になった状態で、HTML ファイル内に下記を記述した場合。

```
<html>
<body>
最終更新日時: <!--#echo var="LAST_MODIFIED" -->
</body>
</html>
```

「<!--#echo var="LAST_MODIFIED" -->」の部分が、環境変数 LAST_MODIFIED（ファイルの最終更新日時）の値に置き換られます。

SSI の echo コマンドで表示できる環境変数

DOCUMENT_NAME	ファイルのファイル名
DOCUMENT_URI	ファイルの URI パス名
DATE_LOCAL	アクセス時の時刻（ローカルタイム）
DATE_GMT	アクセス時の時刻（グリニッジ標準時）
LAST_MODIFIED	ファイルの最終更新時刻

SSI を有効にするには、CGI と同様に Options ディレクティブを用いて、Includes を指定します。また、SSI を実行するファイルには、拡張子「.shtml」がよく用いられます。ファイル名の末尾に「.shtml」がついている場合にのみ、ファイルを解析し（パースする, parse）、文字列などの挿入が行われるように指定できます。

例

```
<Directory [ディレクトリ]>
    Options Includes
    Order allow,deny
    Allow from all
</Directory>
    （中略）
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
```

.shtml ファイルの中に記述できる SSI の命令を SSI ディレクティブと呼びます。SSI ディレクティブは一般的に次のような形式となります。

```
<!--#命令  データ名="データ" -->
```

※3 SSI（の機能を提供する mod_include モジュール）についての詳細は、「http://httpd.apache.org/docs-2.4/mod/mod_include.html」を参照して下さい。

主なSSIディレクティブの一覧

・外部ファイルの読み込み

includeを用いれば、記述した箇所に別のHTMLファイルなどの外部ファイルを読み込むことができます。fileでファイル名を指定する場合、その外部ファイルは呼び出し元のファイルと同じディレクトリに存在している必要があります。virtualで指定するパスは、DocumentRootを/とする絶対パスまたは呼び出し元のファイルからの相対パスで指定します。

```
<!--#include file="ファイル名" -->
<!--#include virtual="パス" -->
```

・ファイルサイズの表示

指定したファイルのサイズを表示します。

```
<!--#fsize file="ファイル名" -->
<!--#fsize virtual="パス" -->
```

・ファイルの最終更新時刻の表示

指定したファイルの最終更新時刻を表示します。

```
<!--#flastmod file="ファイル名 " -->
<!--#flastmod virtual="パス" -->
```

・コマンド実行

指定したコマンドやCGIプログラムを実行し、その結果を表示します。
SSIからのコマンド実行はセキュリティ的に危険な場合が多いので、これを禁止することもできます。禁止する場合は、設定ファイルに「Options Includes」ではなく、「Options IncludesNOEXEC」を指定します。こうすることで、execを用いるSSIディレクティブのみ無視されます。

```
<!--#exec cmd="コマンド" -->
<!--#exec cgi="CGIプログラム" -->
```

14. BIND DNS サーバ（1）

14

BIND DNS サーバ（1）

本章のねらい

- インターネット上の名前解決の仕組みを学ぶ
- ゾーンと管理する DNS サーバの関係を学ぶ
- 名前解決用ツールの使い方を学ぶ
- BIND の設定を学ぶ

予習編

IP アドレスの別名、FQDN

ネットワークに接続されたコンピュータを識別するには IP アドレスが使われます。しかし、IP アドレスは人間から見れば何の脈略もない数字の集まりなので、覚えるには不向きですし何かと不便です。たとえば、情報処理推進機構（IPA）の Web サーバの IP アドレスは 202.122.141.45 です。ブラウザで IPA の Web ページを閲覧するときに、この IP アドレスを使って「http://202.122.141.45/」と入力すれば閲覧することができます（実際にやってみるとよいでしょう）。しかし、これは不便でかつ覚えにくいものです。IP アドレスではなく、文字列「http://www.ipa.go.jp/」のように入力して閲覧できるようになれば便利です。これは Web サーバ以外のホスト（メールサーバなど）に関しても同様です。そのため、ネットワークにあるコンピュータを表す手段として、IP アドレスのほかにホスト名やドメイン名が用意されています。ホスト名やドメイン名は上の「www.ipa.go.jp」のようにドットで区切られた英数字を使った文字列（FQDN と呼ぶ）でホストを表すものです。

ドメイン

上のように FQDN はドットで区切られています。上の「www.ipa.go.jp」は、右から jp が「日本」、go が「政府機関」、ipa が「情報処理推進機構」であることを表していて、最後の www がホストの名前を表しています。このように FQDN はネットワーク上にあるコンピュータを、人間にわかりやすいグループごとにまとめて分類します。このそれぞれのグループを「ドメイン」といいます。

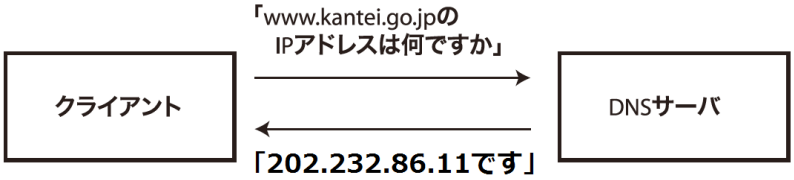
ドメインは、ファイルのディレクトリと同じような階層構造を持っています。これは住所に例えると、「日本」という大きなくくりの中に「東京都」という小さなくくりがあり、その中に「新宿区」という更に小さなくくりがある、と考えるのと同じです。ホスト名の表記では、左に向かうほど範囲が小さくなり、「www.ipa.go.jp」となります。

DNS の役割

ホスト名は人間が理解しやすい形の名前ですが、コンピュータは IP アドレスを使ってホストを識別します。したがって、ホスト名を使うには、ホスト名と IP アドレスを変換する仕組みが必要になります。この仕組みを「名前解決」といいます。

インターネットがまだ小規模だった時代はホストの数が少なかったため、各々のコンピュータ内にホスト名と IP アドレスの対応表を用意しておけば十分でした。しかし、インターネットが大規模になった現在では、この方法では対応しきれませんでした。そこで、名前解決を行なうサーバが必要になりました。このサーバが「DNS（Domain Name System）サーバ」（あるいは「ネームサーバ」）です。DNS サーバはクライアントから FQDN の名前解決の問い合わせを受けると、それに対応する IP アドレスを返答します。また、逆に IP アドレスの問い合わせに対して FQDN を返すこともできます。

DNS のイメージ図



DNS サーバに用いる Linux 標準のデーモンプログラムは「BIND」というプログラムです。

名前解決の仕組み

インターネットがあまりに巨大なネットワークとなった今日では、一台のネームサーバですべてのホストの FQDN と IP アドレスの対応を管理するのは不可能です。そのため、インターネットでは複数の DNS サーバを使って名前解決を行なっています。そして、各々のネームサーバは、受け持つ範囲が決まっています。この範囲を「ゾーン」といいます。たとえば、あるネームサーバは「jp」のドメインを受け持っていて、別のネームサーバは「go.jp」、さらに別のサーバは「ipa.go.jp」を受け持っています。ここでは説明のために、「jp」を受け持つネームサーバを NS1、「go.jp」を受け持つネームサーバを NS2、「ipa.go.jp」を受け持つネームサーバを NS3 とします（NS1、NS2、NS3 も実際は FQDN を持っていますが、ここでは説明を簡単にするためにこのように決めます）。NS1～NS3 は各々が IP アドレスを持っていることに注意してください。

さて、ここでクライアントが、たとえば NS4 というサーバに対して「www.ipa.go.jp」の問い合わせを行なっても、NS4 は www.ipa.go.jp の情報を持っていません。そこで、NS4 は ipa. go.jp のネームサーバ（NS3）に問い合わせをしたいところですが、NS3 の IP アドレスの情報も持ち合わせていないので問い合わせができません。そのため、NS4 は自分が情報を持っていない FQDN の問い合わせを受けると、まず「ルートサーバ」というネームサーバに「jp」を管理するネームサーバ（つまり NS1）の IP アドレスを問い合わせます。なお、ルートサーバは一番上の階層のドメイン（jp、tw、uk、com など）の情報を持つネームサーバを管理するネームサーバです。するとルートサーバから NS1 の IP アドレスが返ってくるので、NS4 は NS1 に対して「go.jp」を受け持つ NS2 の IP アドレスを問い合わせます。すると NS2 の IP アドレスが判明するので、NS4 は NS2 に NS3 の IP アドレスを問い合わせ、判明した NS3 の IP アドレスを使って、NS3 に「www.ipa.go.jp」の IP アドレスを問い合わせます。NS4 はこうして判明した IP アドレスをクライアントに返します。このようにドメインの上の層から順次問い合わせを行なうことによって名前解決を行ないます。

ネームサーバは今やインターネットにサーバを公開するときに欠かせないものになっています。また、設定を間違えると多くの人に迷惑をかけることが往々にしてあります。しっかりと学んでおきましょう。

14.1 インターネット上の名前解決

Web ページを見るとき「www.linuxacademy.ne.jp」などの URL をブラウザに入力することにより、目的の Web サーバからデータを受け取ることができます。これは名前解決という仕組みがあるためです。本来、個々のホストの識別には IP アドレスを用いますが、Web サーバなどにアクセスする度に目的の Web サーバの IP アドレスを手動で調べるのは大変です。名前解決とは、ホスト名から IP アドレスを調べたり、その逆を行うことをいいます。ホスト名から IP アドレスへの変換を正引き、IP アドレスからホスト名への変換を逆引きと呼びます。インターネット上の名前解決はインターネット上にある DNS（Domain Name System）サーバ（単にネームサーバとも呼ぶ）が行っています。

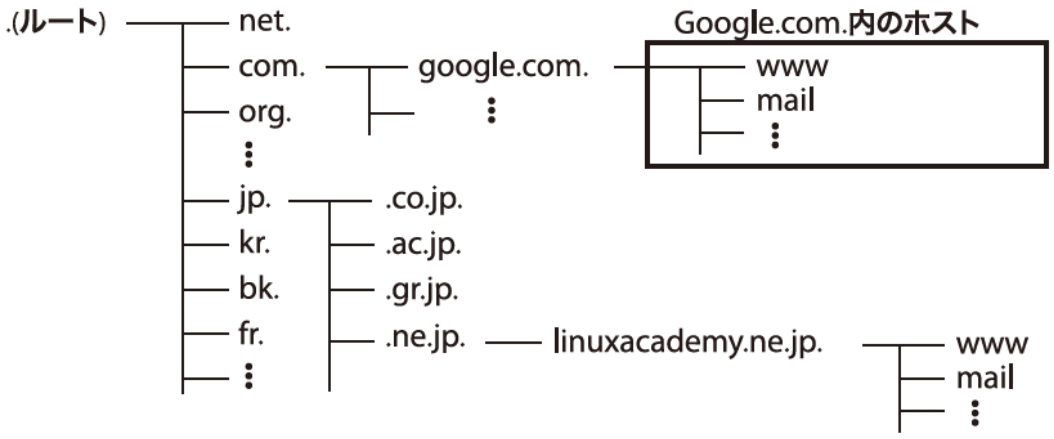
14.1.1 FQDN とドメイン

FQDN（Fully Qualified Domain Name、完全修飾ドメイン名）は、インターネット上のホストを指し示す名前で、ホスト名とドメイン名を連結した形になっています。例えば、「www.linuxacademy.ne.jp」というホストの場合、ホスト名「www」とドメイン名「linuxacademy.ne.jp」が「.」で区切られて表されています。

```
 www      .  linuxacademy.ne.jp
ホスト名          ドメイン名
```

www というホスト名を持つホストは、「www.google.com」や「www.kernel.org」など複数ありますが、それぞれが異なるドメインに属しているので、互いに区別することができます。

ドメインは階層構造で管理されています。次の図はドメインの階層を表しています。



14.1.2 DNS サーバの役割

インターネット上に稼働するホストが少なかった時は、各ホストの IP アドレスとホスト名の組を hosts ファイルに記述することにより、名前解決を行っていました。Linux における「/etc/hosts」ファイルや Windows における「C:¥WINDOWS¥system32¥drivers¥etc¥hosts」などはこの名残りです。しかし何千万台ものホストを 1 つのファイルで扱うのは、ファイルを巨大にし、また管理を困難にします。今日では hosts ファイルの代わりに DNS サーバと呼ばれるサーバを多数動作させ、それらを連携することによりインターネット上の膨大な数のホスト名の名前解決を行っています。

DNS サーバの役割は、クライアントからの名前解決要求に対して IP アドレスや FQDN を返すことです。ブラウザに FQDN を入力して Web サーバにアクセスする場合、クライアントはまず DNS サーバにアクセスし FQDN に対応するホストの IP アドレスを受け取ります。受け取った IP アドレスを用いて、目的の Web サーバにアクセスできるのです。

名前解決要求は、リゾルバ（resolver）と呼ばれるプログラムが行います。リゾルバは、ネームサーバに対して調べたい情報（IP アドレス、ホスト名など）を問い合わせます。

FQDN は前節のような階層構造によって管理されています。頂点の「.（ルートドメイン）」の管理は世界中に配置されている 13 台※1の DNS サーバ（ルートサーバ）が行っています。ルートサーバは一つ下の階層である「net.」や「com.」、「jp.」などのドメインを管理する DNS サーバの情報（IP アドレス）を持っています。「jp.」を管理する DNS サーバは「linuxacademy.ne.jp.」などの下位ドメインを管理する DNS サーバの情報を持っています。「linuxacademy.ne.jp.」を管理する DNS サーバは、そのドメイン内に属するホストの IP アドレスの情報を持っています。このように、個々の DNS サーバは単独で動作しながら、下位の階層にあるホストの IP アドレスの情報を管理しています。ルートサーバから順に下の階層※2に辿っていくと、目的のホストの IP アドレスを見つけることができるようになっています。

14.1.3 リゾルバと DNS サーバ

リゾルバは DNS クライアントとして動作するプログラムです。プログラムといっても、普通のプログラムとは異なり、ライブラリとして機能が提供されています。名前解決を必要とするアプリケーション（ブラウザやメールソフト）はこのライブラリを利用することにより、名前解決を行います。リゾルバの設定ファイルは /etc/resolv.conf です。この resolv.conf に問い合わせを行う DNS サーバを登録しておくことにより、名前解決を行うことができるようになります。

クライアントが要求する FQDN に対応する IP アドレスを問い合わせ先の DNS サーバが知らない場合、DNS サーバがリゾルバに代わり、目的の IP アドレスをルートサーバーから順に問い合わせしていきます。DNS サーバによっては、クライアントからの要求に応じて問い合わせをした結果を、キャッシュとして一時的に保存する機能※3を持つものもあります。

※1 13 台のルートサーバはすべて同じ情報を持っていて、負荷分散しています。そのうち 1 台は日本にあり、WIDE Project が管理・運用しています。
※2 このように階層構造で情報を管理する仕組みを一般的にディレクトリサービスと呼びます。
※3 今回、授業で扱う DNS サーバソフトである BIND もこのキャッシュ機能を持っています。

14.1.4 名前解決とレコード

ネームサーバは名前解決専用のデータベースともいえます。個々のホストの FQDN や IP アドレス、下部ドメインを管理しているネームサーバの情報など、実に様々な情報を管理しています。データベースの持つ情報を一般にレコード（record）と呼びます。ネームサーバにも様々なレコードが存在します。レコードの種類をリソースレコードタイプ（resource record type, RR タイプ）と呼びます。

RR タイプ	意味
NS レコード (Name Server)	ドメインを管理するネームサーバ名（FQDN）を指定するレコード
MX レコード (Mail exchanger)	そのドメイン宛てのメールを配信すべきメールサーバ名（FQDN）を指定するレコード
A レコード (Address)	ドメイン内のホストの IP アドレスを指定するレコード 正引きには A レコードが参照される
CNAME レコード (Canonical Name)	あるホストの正規名（Canonical Name）に対する別名を指定するレコード CNAME レコードを用いて、1 つのホストに複数の FQDN を割り当てることができる
PTR レコード (Pointer)	ドメイン内のホストの FQDN を指定するレコード 逆引きには PTR レコードが参照される
TXT レコード (Text)	他のサーバと連携するための補足情報を指定するレコード なりすましメール防止やクラウドとの連携に用いる
AAAA レコード (Quad A、4 倍の A)	IPv6 の IP アドレスを指定するレコード

ネームサーバはこれらのレコードを格納し、リゾルバや外部のネームサーバの問い合わせに対し応答する機能を持っています。

14.2 名前解決ツール

あるホストの IP アドレスやゾーン内のネームサーバの情報などを知るために、nslookup や dig などのコマンドが用意されています。これらのコマンドは名前解決を行うクライアントプログラムで、正しく名前解決が行われているかをチェックしてくれる、非常に便利なツールとして重宝されています。

14.2.1 nslookup コマンド

nslookup コマンドは、使い方に多少の違いはあるものの Linux 上でも Windows 上でも利用することができる名前解決ツールです。

```
nslookup [オプション] FQDN [ネームサーバの IP アドレス]
```

問い合わせ先のネームサーバの IP アドレスを省略した場合は、/etc/resolv.conf に記載されているネームサーバに問い合わせします。

nslookup の実行例

```
$ nslookup www.google.com
Server:         172.16.0.1
Address:   172.16.0.1#53

Non-authoritative answer:
Name:      www.google.com
Address:   172.217.26.36
```

上の例では、ネームサーバ（172.16.0.1）に問い合わせたところ、www.google.com というホストの IP アドレスは「172.217.26.36」であるということを表しています。

リソースレコードタイプを指定して問い合わせをすることもできます。あるドメイン宛のメールが配信されるべきメールサーバの FQDN（MX レコード）を知りたい場合は、-query オプションを指定して、以下のように行います。

```
nslookup  -query=[RR タイプ]  [問い合わせ先ドメイン]
```

例

```
$ nslookup -query=MX google.com
Server:         172.16.0.1
Address:   172.16.0.1#53

Non-authoritative answer:
google.com      mail exchanger = 10 aspmx.l.google.com.
Google.com      mail exchanger = 20 alt1.aspmx.l.google.com.
                    (省略)
Authoritative answers can be found from:
                    (省略)
aspmx.l.google.com          internet address = 74.125.23.27
alt1.aspmx.l.google.com     internet address = 74.125.30.27
                    (省略)
```

[練習]

1. www.kernel.org の IP アドレスを調べます。
2. kernel.org ドメインの MX レコード、NS レコードを調べます。
3. 存在しないホスト（例えば、xxx.yyy.zzz）に対するネームサーバの応答は、NXDOMAIN（No eXistent DOMAIN、存在しないドメイン）となります。実際に、xxx.yyy.zzz というホストを指定して、このエラーメッセージを確認します。
4. nslookup で www.google.com の IP アドレスを確認します。
   実行結果の中に「Non-authoritative answer:」があります。これは google.com ドメインを管理している DNS サーバに直接問い合わせていない事を表します。
5. google.com ドメインを管理するネームサーバの IP アドレスを調べ、そのネームサーバを指定して google.com ドメインの MX レコードを参照してみましょう。

14.2.2 dig コマンド

dig コマンドは nslookup と同様の働きをします。細かい動作は異なりますが、基本的には nslookup の非対話モードのように用います。

```
dig  [@問合せ先ネームサーバ]  FQDN やドメイン  [RR タイプ]
```

問合せ先ネームサーバと RR タイプを省略すると、/etc/resolv.conf にあるサーバに対して、A レコード（IP アドレス）を問合せます。

問合せた結果表示は「DNS の設定ファイル（ゾーンファイル）に流用できる形式」であること、「サーバ動作に関する詳細な情報が付加されている」ことからシステム管理ツールとしても利用されます。

例

```
$ dig google.com MX

; <<>> DiG 9.9.4-RedHat-9.9.4-37.el7 <<>> google.com MX
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45518
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 4, ADDITIONAL: 15

;; QUESTION SECTION:
;google.com.                            IN  MX

;; ANSWER SECTION:
google.com.                        590   IN  MX  40 alt3.aspmx.l.google.com.
                        :
;; AUTHORITY SECTION:
google.com.                        2470 IN  NS  ns1.google.com.
                        :
;; ADDITIONAL SECTION:
alt3.aspmx.l.google.com.    283   IN  A    209.85.237.25
                        :
;; Query time: 15 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue May 5 01:44:43 2017
;; MSG SIZE rcvd: 503
```

dig の結果は主に 4 つのセクションから成り立っています。

1. **QUESTION SECTION（問い合わせ内容）**
   要求したドメイン名または FQDN と、RR タイプが表示されます。

2. **ANSWER SECTION（検索結果）**
   要求に対する回答が表示されます。この例では google.com ドメインに関する MX レコードが表示されています。

3. **AUTHORITY SECTION（情報の委譲先）**
   問い合わせた情報を管理している（「権威がある」と言います）サーバの情報が表示されます。

4. **ADDITIONAL SECTION（補足情報）**
   問合せ内容に関連した補足情報が表示されます。この例では ANSWER SECTION で表示した各メールサーバの A レコード（IP アドレス）などが表示されます。

dig の結果は nslookup よりも詳しく、ヘッダ情報はネームサーバからの問合せ結果（status）が表示されています。

```
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45518
```

status に続くキーワードが「NOERROR」は正常に回答を得たことを表します。他に「NXDOMAIN」はデータが存在しないという回答を受けたことを、「SERVFAIL」はネームサーバに異常があった（回答がなかった）ことを表します。

[練習]

1. dig コマンドを用いて www.ipa.go.jp の IP アドレスを調べます。
2. ipa.go.jp ドメインの NS レコードを調べます。
3. ipa.go.jp ドメインの MX レコードを調べます。

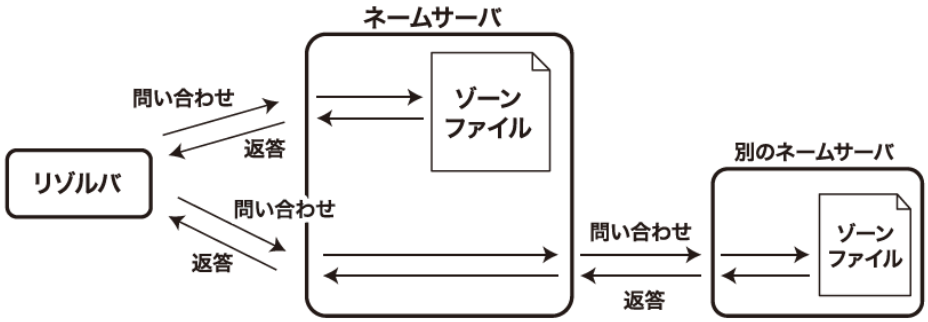
14.3 ネームサーバの構成

14.3.1 ネームサーバの機能

ネームサーバの基本的機能は、リゾルバからの名前解決要求に応答することです。その機能はネームサーバが該当するゾーン情報を持っているか否かによって大きく 2 つに分類することができます。

1 つ目は、自分が持っているゾーン情報をもとに応答する機能です。これは、ネームサーバ内にゾーンファイル（ホスト名と IP アドレスの対応表）が存在する場合を指します。要求に対して、自サーバ内ゾーンファイルに基づいてリゾルバに応答します。これを DNS コンテンツサーバと呼びます。

2 つ目は、問い合わせを受けたゾーン情報を自サーバ内に持たない場合の機能です。この場合、ルートサーバから順番に問い合わせ、ゾーンファイルをもつネームサーバから情報を取得します。そして、その情報をリゾルバに応答※4します。これを DNS キャッシュサーバと呼びます。

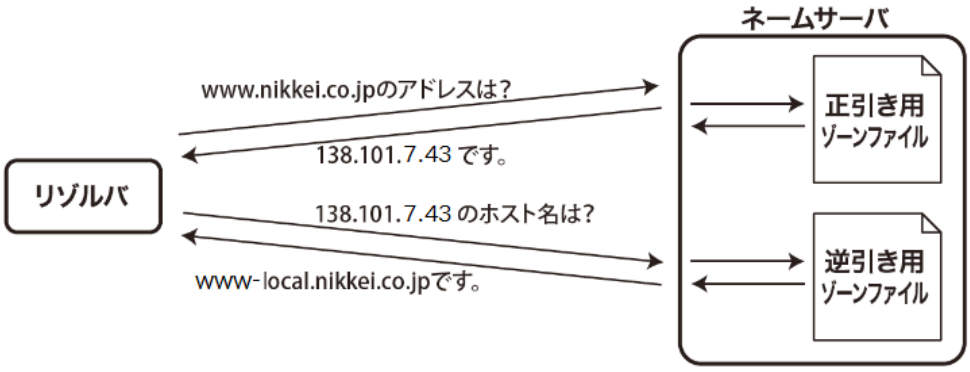


※4 実際は、問い合わせの履歴を一定期間保持しておき、過去に名前解決を行った情報は履歴をもとに返答します（キャッシュサーバ機能）。

14.3.2　ゾーンファイルとゾーン転送

名前解決要求に応じるのはネームサーバですが、応答すべき情報が記述されているのは、その中に配置されているゾーンファイルです。ゾーンファイルには、ホストと IP アドレスの対応一覧などが記述されています。ネームサーバはこのゾーンファイルの内容をもとに問い合わせに対して応答を行います。

ゾーンファイルは正引き用のものと逆引き用のものにわかれています。正引き用のゾーンファイルとはホスト名やドメイン名がわかっている時に IP アドレスを照会するためのものです。逆引き用ゾーンファイルは IP アドレスがわかっている時にホスト名を照会するためのものです。



知りたい情報を持つネームサーバをトップレベルドメインから順次調べる DNS の仕組みでは、特定のゾーンファイルを持つネームサーバは本来 1 つで十分です。しかし、ネットワークやハードウェアのトラブルでそのサーバが停止してしまった時などに備えて、予備のネームサーバを準備しておくのが一般的です。予備のネームサーバに対して、ゾーンファイルを転送（ゾーン転送）しておけば、予備のネームサーバが返答できるようになります。この時、主にゾーンファイルを管理している方をプライマリ（マスター）ネームサーバ、予備の方をセカンダリ（スレーブ）ネームサーバといいます。

14.4 確認問題

DNS クライアントから自身の管理するゾーン外の名前解決の要求をうけた DNS サーバが、再帰的に名前解決をする様子を dig コマンドを使用して再現してみます。ここでは「www.linuxacademy.ne.jp」を例にとります。

1. まず、ルートドメイン(.)を管理しているネームサーバに対して、jp ドメインを管理しているネームサーバの IP アドレスを問い合わせます。
   ルートドメイン(.)を管理しているネームサーバの IP アドレスは/var/named/named.ca 内に記述されています。今回は日本にある M.ROOTSERVERS.NET.に対して問い合わせます。

```
$ dig @202.12.27.33 jp. NS
```

2. 判明した DNS サーバに対し、linuxacademy.ne.jp ドメインを管理しているネームサーバの IP アドレスを問い合わせます。

```
$ dig @[(1)の IP アドレス] linuxacademy.ne.jp. NS
```

3. 最後に、linuxacademy ドメインを管理するネームサーバに対して、www.linuxacademy.ne.jp.の A レコードを問い合わせます。

```
$ dig @[(2)の IP アドレス] www.linuxacademy.ne.jp. A
```

15

BIND DNS サーバ(2)

本章のねらい

- ゾーンの定義を行う
- 正引きの設定を行う

予習編

ゾーンの定義

DNS コンテンツサーバは、SOA レコードで宣言した範囲をゾーンとし、その内容を管理します。ゾーン情報はテキストファイルで、中には RR タイプを使った情報を記述します。リゾルバからの問い合わせに対し、自身が管理するゾーン情報であれば回答し、そうでなければ上位の DNS サーバへ問い合わせを行います。

正引きの設定

ゾーンファイルには、ホスト名から IP アドレスへの変換である正引きと、IP アドレスからホスト名へ変換する逆引があります。本章では、基本である正引き用を詳しく見ていきます。正引き用のゾーンファイルの IP アドレス定義部分は以下のようになります。

s142.la.net のゾーンファイル

```
h010.s142.la.net.           10.20.142.10
h020.s142.la.net.           10.20.142.20
h030.s142.la.net.           10.20.142.30
h040.s142.la.net.           10.20.142.40
h050.s142.la.net.           10.20.142.50
h060.s142.la.net.           10.20.142.60
```

これは、h010.s142.la.net の IP アドレスが 10.20.142.10 であることを表しています。

あるドメインに所属するホストを名前解決するには、関連する情報をゾーンファイルとして定義する必要があります。

実際のゾーンファイルには、FQDN と IP アドレスの対応以外の情報も記述されていますが、それは授業の中で見ていくことにしましょう。

15.1 ゾーンの定義

BIND（Berkeley Internet Name Domain）は古くからインターネット上で利用されているネームサーバソフトです。現在では、ISC（Internet Systems Consortium）により配布されています。ISC の Web サイト（http://www.isc.org/）において、最新版※1の BIND ソースパッケージを無償でダウンロードすることができます。

BIND をインストールします。

```
# yum -y install bind
```

15.1.1 ゾーンの委譲とネームサーバの設置

インターネット上のドメイン名は、重複などのトラブルが起きないよう統一的に管理されています。誰もが自由に適当なドメインを使うは許されず、申請し許可を受けて初めて使用できます。ドメインを取得するということは、その上位ドメインから管理を委任されているということになります。

例えば、「linuxacademy.ne.jp」というドメインは上部の「ne.jp」というゾーンから管理を委任されており、「ne.jp」ゾーンは「jp」ゾーンから、「jp」ゾーンはルートゾーンから管理を委任されているということになります。

以下の講義においては、「ne.jp」や「jp」などの上位ゾーンではなく、「linuxacademy.ne.jp」などの具体的なドメインに関する名前解決を BIND を用いて提供するための方法を解説していきます。

※1 BIND は 2017 年現在、9.9 と 9.10 と 9.11 が提供されています。

15.1.2 named.conf

ネームサーバは、サーバ機能を設定する/etc/named.confと、ゾーン情報の２種類を定義する必要があります。named.confを以下のように修正※2します。

named.conf

```
options {
        listen-on port 53 { 127.0.0.1; 自ホストのIP アドレス;  };
        listen-on-v6 port 53 { ::1; };      ↑ IP アドレス追加
        directory "/var/named";
             （中略）
        dnssec-enable no;          ←  yesをnoに変更
        dnssec-validation no;      ←  yesを noに変更
        empty-zones-enable no;     ←  １行追加

        /* Path to ISC DLV key */
        bindkeys-file "/etc/named.iscdlv.key";
             （後略）
};
```

修正を行ったら、named-checkconfigを使って文法チェックを行います。エラーがなければ何も表示されません。

```
# named-checkconf /etc/named.conf
```

以下に主なnamed.confの構文を解説します。

構文	意味
options	BIND全体の動作に関わる設定。主な項目としては以下がある。 ・listen-on port ポート番号 待受けポート { IP; }; ・directory “パス”; ゾーンファイル格納先 ・allow-query { クライアント; }; 問合せ許可クライアント情報 ・dnssec-enable yes/no; DNSSECの機能の有無 ・dnssec-validation yes/no; DNSSECにゾーン情報確認の有無 ・empty-zones-enable yes/no; 無効IP解決抑止の有無
logging	BINDが出力するログの制御を行う。
zone	ゾーン情報（ドメイン名、サーバ属性など）の定義を行う。
include	外部ファイルの取り込み。

※2 最後の「}（閉じ括弧）」のあとの「;（セミコロン）」を忘れると、BINDはうまく起動しません。注意しましょう。

15.1.3 ローカルゾーンファイルの準備

ローカルホストに関するゾーン情報は /etc/named.rfc1912.zones にまとめて記載されています。具体的には以下の通りです。またゾーンファイル自体は /var/named 下に格納されています。

ゾーンファイル名	内容
named.localhost	localhost および localhost.localdomain の正引き
named.loopback	::1(1::ip6.arpa)および、127.0.0.1(1.0.0.127.in-addr.arpa)の逆引き

ルートサーバの IP アドレスを格納するルートキャッシュは named.ca ですが、以下の方法で最新版を入手することができます。

```
# dig . NS @202.12.27.33 > /var/named/named.ca
```

これは m.root-servers.net(日本で運営しているルートネームサーバ,202.12.27.33)に対し、ルートドメイン(.)のネームサーバを問い合わせています。

15.1.4 ゾーンの定義

zone 文を用いて、ゾーンの定義を行います。zone 文は以下のような構造になっています。

```
zone "ゾーン名" IN {
    設定;
};
```

例えば、「s142.la.net」というドメインを管理することになった場合、ネームサーバが管理するゾーン名は「s142.la.net」であり、以下のように記述します。

ゾーン定義の例

```
zone "s142.la.net" IN {
    type master;
    file "named.s142.la.net";
};
```

zone 文中の設定には、ゾーンタイプ、ゾーンファイル名、オプションを記述します。

ゾーンタイプ（zone type）

```
type [ゾーンタイプ];
```

ゾーンタイプでは、設定するゾーンに対してどのような役割を持っているかを定義します。以下に代表的な 3 つのゾーンタイプを紹介します。

ゾーンタイプ	意味
master	設定するゾーンに対する要求に権威ある回答をすることができるネームサーバであることを表す。最新で最も正確な情報を持つのはマスターネームサーバである。
slave	type に”master”が指定されているネームサーバのコピー。冗長性を確保するためにネームサーバは複数設置される。
hint	ルートネームサーバへの参照。

権威のある回答を返すことのできるネームサーバをマスターネームサーバと呼びます。あるゾーンに対するマスターネームサーバを設定するためには、ゾーンタイプを「master」とします。

ゾーンファイル名

```
file [ゾーンファイル名];
```

ゾーンファイルとは、あるゾーンに属するホストの IP アドレスやメールサーバの情報など、名前解決で利用される様々なレコードを実際に記述するファイルです。named.conf 内でゾーンファイル名を指定します。ゾーンファイルは、options 文の directory オプションで指定されたディレクトリ（デフォルトでは/var/named/）内に作成し、ファイル名はそのディレクトリからの相対パスで指定できます。

オプション

設定するゾーンに関する細かなアクセス制限や動作に関する設定を行います。問い合わせを受け付けるホストを指定する allow-query オプション、クライアントからのゾーン情報更新（動的 DNS）を可能にする allow-update オプションなど様々なものがあります。

15.2 正引きの設定

15.2.1 正引きゾーンファイルの作成

完成したゾーンファイルの例

```
$TTL 86400
s142.la.net. IN SOA ns.s142.la.net. root.s142.la.net. (
              2017080702     ; Serial
              28800          ; Refresh
              14400          ; Retry
              3600000        ; Expire
              86400 )        ; Negative
s142.la.net.            IN NS       ns.s142.la.net.
s142.la.net.            IN MX 10    smtp.s142.la.net.
ns.s142.la.net.         IN A        10.20.142.6
smtp.s142.la.net.       IN A        10.20.142.6
h006.s142.la.net.       IN A        10.20.142.6
www.s142.la.net.        IN CNAME    h006.s142.la.net.
```

ゾーンファイルは、そのゾーンに関する情報（ゾーンを管理するネームサーバ、そのゾーンに含まれている各ホストの IP アドレスなど）のデータベースで、一行に 1 レコードずつ記載されています。それぞれのレコードは以下の 5 つのフィールドから構成されています。

フィールド名	説明
名前	レコードの名前、問合せキー
TTL	このレコードのキャッシュ残存時間（秒[3]）、省略時は$TTL を採用
クラス	IN 固定（Internet）
レコードタイプ	レコードタイプ（NS, MX, A, CNAME など）
値	リゾルバへの回答

レコードタイプにより名前フィールドと値フィールドの組合せが決まります。

冒頭にある$TTL（Time To Live, 有効期限）は、各レコードのキャッシュ保持時間のデフォルト値です。先の例では、全レコードの TTL が省略されているので、$TTL で指定されている 86400 秒＝1 日が適用されます。BIND9 以降では、$TTL は必須です。

ゾーンファイルの 2 番目のレコードは、「s142.la.net.」というゾーンの「NS」（ゾーンを管理しているネームサーバ）が「ns.s142.la.net.」であること定義しています。更に、4 番目のレコードで、「ns.s142.la.net.」の「A」（IP アドレス）は「10.20.142.6」としています。

※3 時間の単位は他にも分（M）、時間（H）、日（D）、週（W）が指定できます。

名前フィールドと値フィールドにて記述するゾーン名、ホスト名はすべて FQDN で表記します。また、「s142.la.net.」「h006.s142.la.net.」のように、ルートドメインを表す「.」を最後に付けます。忘れた場合は後述するゾーン名の補完が行わるため、最後の「.」は必ず付けてください。

以下それぞれのレコードタイプについて、解説します。

SOA レコード（Start Of Authority）

```
[ゾーン名]  IN  SOA  [マスターネームサーバ] [管理者のメールアドレス]  ([オプション])
```

SOA レコードは、ゾーンファイル自体の定義を行います。名前フィールドには、ゾーン名を指定します。値フィールドには、そのドメインを管理しているマスタネームサーバを指定し、次に管理者のメールアドレスを指定します。BIND 内部で「@」は特別な意味があるため、メールアドレスの「@」は「.」に置き換えます。最後は（）内に以下のオプション値を指定します。

オプション名	意味
Serial	ゾーンファイルの更新番号（シリアル番号）
Refresh	スレーブがマスターのゾーンファイルをチェックする間隔（秒）
Retry	マスターへの接続に失敗したとき、再び接続を試みるまでの間隔（秒）
Expire	マスターと連絡不能になってから、スレーブがデータを破棄するまでの時間（秒）
Negative	ネガティブキャッシュの TTL とも呼ばれ、問い合わせを受けたホストがゾーンの中に存在しないという情報を保持しておく時間（秒）

オプションのうち特に重要なのは Serial 値です。スレーブリサーバは、マスターサーバの Serial 値を手持ちの値と比較し、マスターサーバ側が大きな場合のみゾーン転送を実行します。1 から順番に数値を増やしていっても構いませんが、一般的には「2017080702」のように更新日付8桁※4と、その日の更新回数 2 桁を合わせた 10 桁で指定します。この場合、2017 年 8 月 7 日の 2 回目の更新を意味します。ゾーンファイルを変更した場合、必ず Serial の値を増やしてください。

※4 10 桁以上の数値を指定するとエラーになり、BIND 起動時にそのゾーンファイルの読み込みに失敗します。

NS レコード（Name Server）

```
[ゾーン名]　IN　NS　[ゾーン内のネームサーバ名])
```

ゾーン内のネームサーバ名を指定します。ゾーンに対して必須のレコードです。

MX レコード（Mail exchanger）

```
[ゾーン名]　IN　MX　[優先度]　[ゾーン内のメールサーバ名]
```

ゾーン内のメールサーバ名（SMTP）を指定します。「優先度」はゾーン内に複数のサーバが存在するとき、優先度が高いサーバ順に値を指定していきます。優先度の値が小さいほど優先され、値が同じものが複数ある場合はラウンドロビン方式で順番に負荷分散されます。

A レコード（Address）

```
[ホスト名]　IN　A　[ホストの IP アドレス]
```

ゾーン内の個々のホスト名と IP アドレスを対応付けます。A レコード以外に値フィールドが IP アドレスになることはありません。

CNAME レコード（Canonical NAME）

```
[別名]　IN　CNAME　[A レコードで指定したホストの正規名]
```

一つのサーバに複数の名前を対応させるには CNAME レコードを記述します。正式名は A レコードで指定します。例えば、正式名「h142.s142.la.net」に別名「www.s142.la.net」を付けたい場合、例題のような CNAME レコードを記述します。
CNAME の別名は、他のレコードの値フィールドとして使用できません。

[練習]

1. 前節で定義したゾーンについて、例を参考にゾーンファイル/var/named/named.s142.la.net を作成します。
2. named-checkzone を使って作成したゾーンファイルの文法チェックを行います。

```
# named-checkzone s142.la.net. /var/named/named.s142.la.net
zone s142.la.net/IN: loaded serial 2017080702
OK
```

15.2.2 BIND の起動

BIND ネームサーバのプロセス名は「named」となります。Apache Web サーバの起動と同様に、systemctl コマンドを用います。

```
systemctl [start|stop|restart] named
```

上手く起動しない場合は、ログを確認します。root ユーザになって、/var/log/messages を参照してみましょう。ログファイルは末尾に新しいログが追記されていくので、ファイルの末尾から指定行文を参照できる tail コマンド※5を使用するといいでしょう。

```
# tail /var/log/messages
```

[練習]

1. BIND を起動します。
2. dig を用いて、前節までに設定したゾーンに対する名前解決ができているかどうかを確かめます。ネームサーバは自分自身を指定します。

例

```
$ dig h006.s142.la.net
```

3. 「 sxxx.la.net」ゾーンの NS レコードを dig で調べます。
4. 隣の方が成功しているようであれば、隣の方のゾーンに関する情報を調べます。

※5 tail コマンドで表示する行数を指定したい場合は「-（行数）」というオプションをつけます。また、「-f」というオプションをつけるとリアルタイムでログを確認することができるので便利です。

15.2.3 ゾーンの省略

ゾーンファイル中、ゾーン名の部分を@で代用することができます。これによりゾーンファイルは非常に簡潔に記述できるようになります。@で代用したゾーン名は、/etc/named.conf におけるゾーンの定義から補われますが、下記のように、$ORIGIN ディレクティブを使用して補完するゾーン名を明示することもできます。また、名前フィールドを省略した場合は、前の行の名前フィールドの値が補完されます。

省略したゾーンファイルの例

```
$TTL 1D
$ORIGIN s142.la.net.
@        IN   SOA       ns    root (
              2017080702     ; Serial
              8H             ; Refresh
              4H             ; Retry
              1000H          ; Expire
              1D )           ; Negative
         IN   NS        ns
         IN   MX   10   smtp
ns       IN   A         10.20.142.6
smtp     IN   A         10.20.142.6
h006     IN   A         10.20.142.6
www      IN   CNAME     h006
```

ゾーン名を省略する場合は、名前の末尾に付けていた「.」も外します。ゾーンファイルで FQDN の記述に用いられる場合の末尾の「.」は、ルートドメインであるという特別な意味を持っています。名前の末尾に「.」が付いていない場合は、そのあとにゾーン名が省略されているとみなされ、$ORIGIN パラメータで指定されたゾーン名が補完されます。

例

```
$ORIGIN   s142.la.net.
              （省略）
ns.s142.la.net   IN    A     10.20.142.6
```

上記のように指定してしまった場合、「ns.s142.la.net」の後ろに「s142.la.net.」が補完されて、「ns.s142.la.net.s142.la.net.」とみなされます。ゾーンファイルを記述するときは、ルートサーバを表す「.」の有無に十分注意してください。

column

キャッシュネームサーバ

キャッシュの重要性

一般にネームサーバの持つ機能として、次の 2 つが挙げられます。

1. あるゾーンを管理し、他のネームサーバやクライアントからの要求に対して権威ある回答を示す。（マスターネームサーバ、スレーブネームサーバ）
2. ゾーンは管理せず、ルートネームサーバから再帰的に他のネームサーバを問い合わせることによりクライアントからの名前解決要求に答える。問い合わせした結果をキャッシュとして残すものもある。

（1）の機能を持つネームサーバは今までに設定してきたネームサーバが持つ機能です。ゾーンを管理し、他のホストがゾーン内のホストの正引きや逆引きを参照できるようにします。
（2）の機能を持つネームサーバを再帰ネームサーバと呼びます。併せてキャッシュの機能を持つネームサーバを、キャッシュネームサーバと呼びます。

名前解決においてキャッシュは非常に重要です。再帰ネームサーバは、ルートネームサーバから順に辿って目的のドメインやホストの IP アドレスを調べますが、これには複数回の問い合わせが必要となります。キャッシュなしに全てのネームサーバが毎回同じようにルートネームサーバから再帰的に参照していくことは、ネットワークに大量の参照要求が流れることになり多大な負荷につながります。BIND では、デフォルトの設定で既にキャッシュサーバとしての機能が有効になるように設定されており、一度問い合わせが成功したホストの IP アドレスの情報は一定期間保存するようになっています。

キャッシュネームサーバの設定

BIND においては、再帰ネームサーバとキャッシュネームサーバの違いを意識することはほとんどありません。再帰ネームサーバの設定を行うと、キャッシュネームサーバとしての機能も果たすからです。

デフォルトの/etc/named.conf ファイルには、以下の記述があります。

```
zone "." IN {
    type hint;
    file "named.ca";
};
```

サーバの種類を「hint」に指定することで、ルートネームサーバへの参照を明示します。再帰ネームサーバは 13 台あるルートネームサーバの IP アドレスを知っているので、再帰的な問い合わせを行うことができます。「file」で指定されている「named.ca」は各ルートネームサーバの NS レコードと A レコードを保持しています。

/var/named/named.ca

```
    ...
;; ANSWER SECTION:
.                       518400      IN    NS    a.root-servers.net.
.                       518400      IN    NS    b.root-servers.net.
    （省略）
.                       518400      IN    NS    m.root-servers.net.
;; ADDITIONAL SECTION:
a.root-servers.net.     3600000     IN    A     198.41.0.4
a.root-servers.net.     3600000     IN    AAAA  2001:503:be3e::2:30
    （省略）
m.root-servers.net.     3600000     IN    A     202.12.27.33
m.root-servers.net.     3600000     IN    AAAA  2001:dc3::35
```

a.～m.root-servers.net.までの 13 台のネームサーバそれぞれについて、NS レコード、A レコードが記述されています。このうち、m.root-servers.net は日本の WIDE プロジェクト（http://www.wide.ad.jp）が管理しています。

管理外のゾーンに関してクライアントからの問い合わせを受けた場合、ネームサーバは、/var/named/named.ca ファイルを参照してルートサーバのいずれか 1 つに問い合わせします。あとは再帰的に目的のホストの IP アドレスや FQDN などが分かるまで、他のネームサーバを辿っていき、権威のあるネームサーバから目的の情報が得られた場合、その結果をクライアントに送り返し、同時にその情報をキャッシュとして保存します。保存されたキャッシュは named を停止もしくは再起動すると消去されます。

再帰ネームサーバの確認

BIND は基本的に再帰ネームサーバとして機能します。それを確認するため、named を再起動し、dig コマンドを用いて確かめてみます。

```
# systemctl restart named
```

次に管理外のゾーンである、www.google.com を自分のホストに問い合わせします。

```
$ dig @127.0.0.1 www.google.com A

; <<>> DiG 9.9.4-RedHat-9-9-4-el7_3.1 <<>> @127.0.0.1 www.google.com A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23683
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 5

;; QUESTION SECTION:
;www.google.com.                    IN      A

;; ANSWER SECTION:
www.google.com.      300            IN      A       172.217.27.164

;; AUTHORITY SECTION:
google.com.                 172800  IN      NS      ns3.google.com.
     (省略)
;; ADDITIONAL SECTION:
ns2.google.com.      172800  IN      A       216.239.34.10
     (省略)
;; Query time: 226 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Mon Aug 07 14:31:30 2017
;; MSG SIZE rcvd: 195
```

少し時間がかかった（226 msec）のち、www.google.com に関する情報が表示されます。

同様に www.kernel.org を問合せますが、今度は再帰呼び出しを行わないよう、「+norec」（No Recursion）オプションを付けて実行します。

```
$ dig @127.0.0.1 www.kernel.org +norec

; <<>> DiG 9.9.4-RedHat-9.9.4-50.el7_3.1 <<>> @127.0.0.1 www.kernel.org A +norec
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 35711
;; flags: qr ra; QUERY: 1, ANSWER: 0, AUTHORITY: 13, ADDITIONAL: 27

;; QUESTION SECTION:
```

```
;www.kernel.org.                        IN          A

;; AUTHORITY SECTION:
.                       517918  IN      NS      e.root-servers.net.
.                       517918  IN      NS      h.root-servers.net.
     （省略）
;; ADDITIONAL SECTION:
l.root-servers.net.  604318  IN      A       199.7.83.42
l.root-servers.net.  604318  IN      AAAA    2001:500:9f::42
     （省略）
;; Query time: 12 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Mon Aug 07 14:32:30 2017
;; MSG SIZE rcvd: 826
```

この例では、www.kernel.org の表示の代わりに最初に問い合わせを実行するルートサーバに関する情報が表示されます。再帰問い合わせを無効にしたため、ルートサーバの次のネームサーバーへ問い合わせがされなかったためです。

キャッシュネームサーバの確認

キャッシュネームサーバの動作確認を行います。前節で既に www.google.com に関してはすでに再帰問い合わせが実行されており、キャッシュネームサーバとして動作しているのであれば、問い合わせの結果がローカルのネームサーバ側に保存されているはずです。これを確認するために、再帰問い合わせを無効にして dig コマンドを実行してみましょう。もし、キャッシュとして残されているのであれば、ローカル側に保存されたデータがすぐに表示されるはずです。またキャッシュとして残されていなければ、先ほど www.kernel.org に問い合わせした場合のようにルートネームサーバへの参照情報が表示されるだけとなるはずです。

```
$ dig @127.0.0.1 www.google.com A +norec
     （中略）
;; Query time: 3 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Mon Aug 07 14:33:30 2017
;; MSG SIZE rcvd: 179
```

今度はきちんと情報が返ってきます。また、レスポンスが早くなっていることが分かります（3msec）。

BIND のログ

どのようなサービスを運営する上でも、ログに関して注意深くなることは非常に重要なことです。BIND に関しても同様なことが言えます。ここでは、デフォルトのログ設定とログの取り方に関する設定方法について紹介していきます。Linux では、通常 syslogd（CentOS7 では rsyslogd）と呼ばれるログ専用デーモンにより、様々なサービスのログが記録されます。デフォルトの設定では、/var/log/messages ファイルに BIND の起動や停止、アクセス拒否などの基本的なログが出力されます。

BIND 起動時のログ

```
Aug 3 21:31:58 h142 named[4393]: starting BIND 9.9.4-RedHat-9.9.4-
50.el7_3.1 -u named
Aug 3 21:31:58 h142 named[4393]: built with '--build=x86_64-redhat-linux-
gnu' ...
    （省略）
Aug 3 21:31:58 h142 named[4393]: sdjusted limit on open files from 4096
to 1048576
Aug 3 21:31:58 h142 named[4393]: found 2 CPUs, using 2 worker threads
Aug 3 21:31:58 h142 named[4393]: using 2 UDP listeners per interface
Aug 3 21:31:58 h142 named[4393]: using up to 4096 sockets
Aug 3 21:31:58 h142 named[4393]: loading configuration from
'/etc/named.conf'
    （省略）
```

named.conf やゾーンファイルを記述する際に、入力ミスをしてしまうことがあるかもしれません。これらのファイルにエラーがある場合、named が起動しなかったり、ゾーンが正しく読み込まれなかったりします。エラーログの中にはエラーの在処を教えてくれるヒントが残されていることが多々あります。

付録
確認問題 解答例

第９章

9.2.1 練習

```
1. 11000000  10101000  00000000  00000001
2. 01100100  10010110  11001000  11111010
3. 172.20.0.1
```

9.2.2 練習

```
1. 24 bit
2. 8 bit
3. ネットワーク部が192.168.0であるので
    ネットワークアドレス     192.168.0.0
    ブロードキャストアドレス 192.168.0.255
```

9.2.5 練習

```
1. ネットワークアドレス   192.168.0.64
   ブロードキャストアドレス 192.168.0.127
2. 62個（2^6=64 から、ネットワーク・ブロードキャストアドレスの２つ除く）
3. 29 bit
```

9.2.6 練習

```
(2)のみ。各選択肢のネットワークアドレスを比較結果は以下の通り。
1. 172.20 ≠ 172.1            X
2. 172.16 = 172.16            ○
3. 192.168.1 ≠ 192.168.2      X
4. 192.168.1.(00110010)₂ ≠
   192.168.1.(11001000)₂      X
```

9.4.1 練習の回答

```
[student@h006 ~]$ ip route show
default via 192.168.10.1 dev enp0s25 proto dhcp metric 100
192.168.10.0/24 dev enp0s25 proto kernel scope link src 192.168.10.129 metric 100
192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
     上記下線部がデフォルトゲートウェイ
[student@h006 ~]$ ping 192.168.10.1
PING 192.168.10.1 (192.168.10.1) 56(84) bytes of data.
64 bytes from 192.168.10.1: icmp_seq=1 ttl=255 time=0.430 ms
64 bytes from 192.168.10.1: icmp_seq=2 ttl=255 time=0.442 ms
^C
--- 192.168.10.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.430/0.436/0.442/0.006 ms

[student@h006 ~]$ 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
     valid_lft forever preferred_lft forever
2: enp0s25: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
default qlen 1000
     link/ether 00:21:cc:d6:3c:42 brd ff:ff:ff:ff:ff:ff
     inet 192.168.10.129/24 brd 192.168.10.255 scope global noprefixroute dynamic
enp0s25
     (省略)
[student@h006 ~]$ ping 192.168.10.129
PING 192.168.10.129 (192.168.10.129) 56(84) bytes of data.
64 bytes from 192.168.10.129: icmp_seq=1 ttl=64 time=0.070 ms
64 bytes from 192.168.10.129: icmp_seq=2 ttl=64 time=0.068 ms
^C
--- 192.168.10.129 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.068/0.069/0.070/0.001 ms

（隣の IP が  192.168.10.113 の場合）
[student@h006 ~]$ ping -c 5 192.168.10.113
PING 192.168.10.113 (192.168.10.113) 56(84) bytes of data.
64 bytes from 192.168.10.113: icmp_seq=1 ttl=64 time=1.37 ms
（省略）
64 bytes from 192.168.10.113: icmp_seq=5 ttl=64 time=0.724 ms

--- 192.168.10.113 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4001ms
rtt min/avg/max/mdev = 0.418/0.762/1.374/0.325 ms
[student@h006 ~]$ ping -c 10 www.example.net
PING www.example.net (93.184.216.34) 56(84) bytes of data.
64 bytes from 93.184.216.34 (93.184.216.34): icmp_seq=1 ttl=50 time=104 ms
（省略）
64 bytes from 93.184.216.34 (93.184.216.34): icmp_seq=10 ttl=50 time=104 ms

--- www.example.net ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9001ms
rtt min/avg/max/mdev = 104.013/104.414/107.526/1.047 ms
```

9.4.2 練習の回答（一部省略）

```
[student@h006 ~]$ traceroute 127.0.0.1      （ローカルルークバック）
traceroute to 127.0.0.1 (127.0.0.1), 30 hops max, 60 byte packets
 1  localhost (127.0.0.1)  0.068 ms  0.024 ms  0.022 ms
```

```
[student@h006 ~]$ traceroute 192.168.10.129（先の ip コマンドの結果）
traceroute to 192.168.10.129 (192.168.10.129), 30 hops max, 60 byte packets
 1  h006.s121.la.net (192.168.10.129)  0.073 ms  0.026 ms  0.022 ms

[student@h006 ~]$ traceroute 192.168.10.1
traceroute to 192.168.10.1 (192.168.10.1), 30 hops max, 60 byte packets
 1  gateway (192.168.10.1)  0.451 ms  0.394 ms  0.358 ms
[student@h006 ~]$
[student@h006 ~]$ traceroute www.linuxacademy.ne.jp
traceroute to www.linuxacademy.ne.jp (203.174.70.235), 30 hops max, 60 byte packets
 1  gateway (192.168.10.1)  0.272 ms  0.172 ms  0.354 ms
 2  172.30.0.1 (172.30.0.1)  3.038 ms  2.956 ms  2.913 ms
 3  * * *
 4  web2.linuxacademy.ne.jp (203.174.70.235)  25.127 ms  25.453 ms  26.144 ms
```

9.5 確認問題の回答

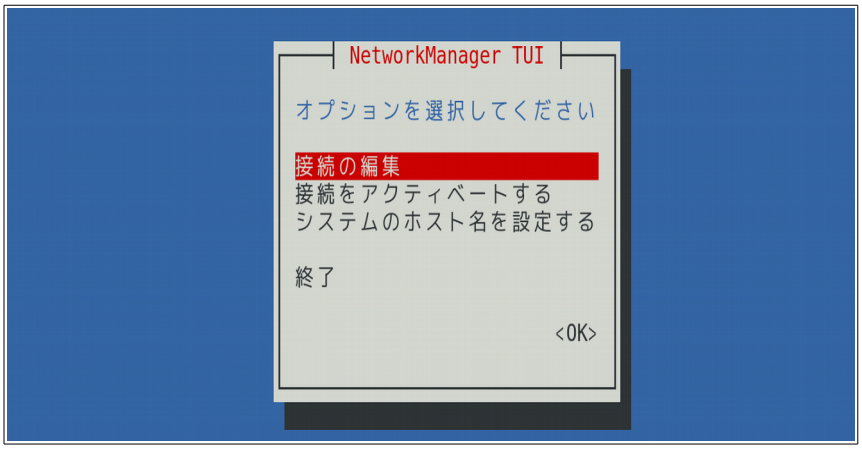
1. $11001010.11100000.10000000.01100100_{2}$
2. ネットワークアドレス　202.224.128.0
   ブロードキャストアドレス　202.224.128.255
3. 254 個

4. IP アドレスの確認

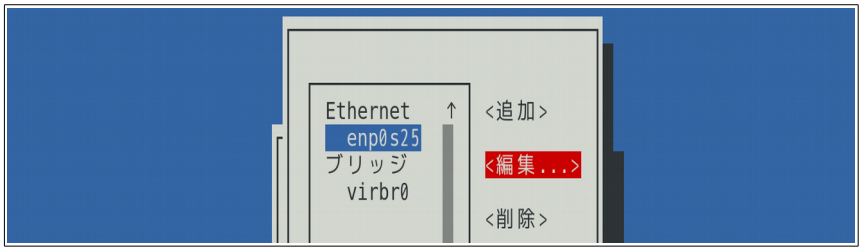
```
[student@h006 ~]$ ip addr show  enp0s25
2: enp0s25: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
default qlen 1000
    link/ether 00:21:cc:d6:3c:42 brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.129/24 brd 192.168.10.255 scope global noprefixroute dynamic
enp0s25
    valid_lft 6851sec preferred_lft 6851sec
    inet6 fe80::221:ccff:fed6:3c42/64 scope link
    valid_lft forever preferred_lft forever
```

5. ネットワーク設定の変更

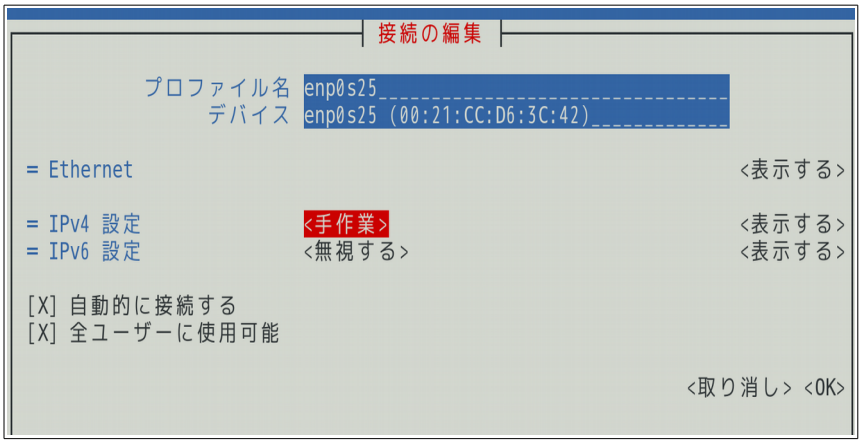
```
[root@h006 ~]# nmtui
```



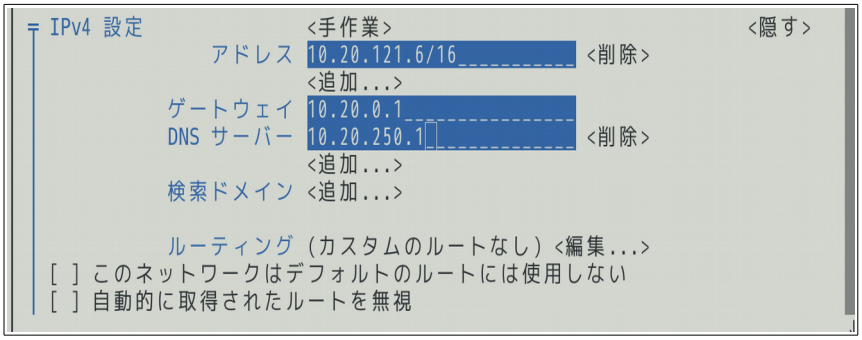
「接続の編集」をクリック



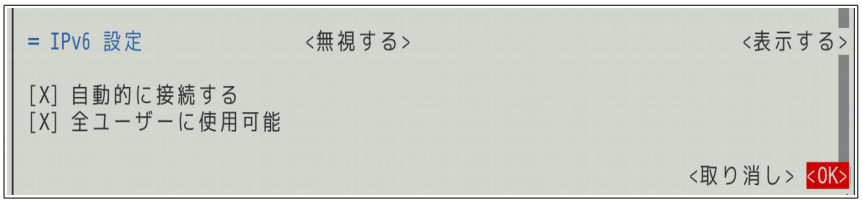
「enp」で始まるインタフェースを選択し、＜編集＞をクリック



IPv6 を＜手作業＞に変更し、＜表示する＞をクリックし、入力画面を開く。

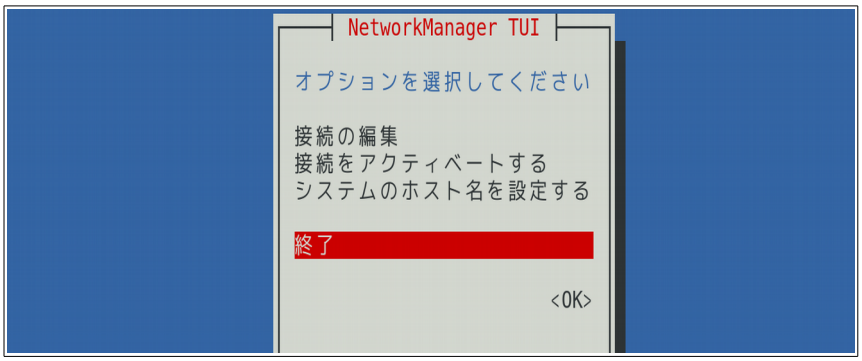


指定された値を設定する。



画面下の＜OK＞をクリックする。

インタフェース選択画面で、＜戻る＞



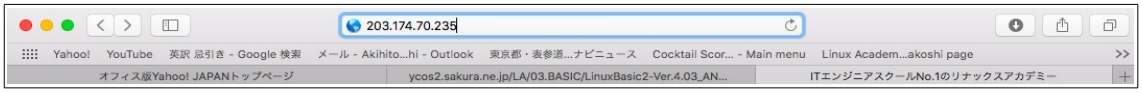
終了を選択し、nmtui を終了する。

```
6. ネットワーク再起動と内容の確認
[root@h006 ~]# systemctl restart network
[root@h006 ~]# ip addr show enp0s25
2: enp0s25: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
default qlen 1000
    link/ether 00:21:cc:d6:3c:42 brd ff:ff:ff:ff:ff:ff
    inet 10.20.121.6/16 brd 10.20.255.255 scope global noprefixroute enp0s25
    valid_lft forever preferred_lft forever
    inet6 fe80::221:ccff:fed6:3c42/64 scope link
    valid_lft forever preferred_lft forever
[root@h006 ~]# cat /etc/resolv.conf
# Generated by NetworkManager
search s121.la.net
nameserver 10.20.250.1

[student@h006 ~]$ ping www.linuxacademy.ne.jp
PING www.linuxacademy.ne.jp (203.174.70.235) 56(84) bytes of data.
64 bytes from web2.linuxacademy.ne.jp (203.174.70.235): icmp_seq=1 ttl=62 time=5.14 ms
64 bytes from web2.linuxacademy.ne.jp (203.174.70.235): icmp_seq=2 ttl=62 time=1.92 ms
^C
--- www.linuxacademy.ne.jp ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.921/3.532/5.144/1.612 ms
[student@h006 ~]$ nslookup www.linuxacademy.ne.jp
Server:     192.168.10.1
```

```
Address:    192.168.10.1#53

Non-authoritative answer:
Name:    www.linuxacademy.ne.jp
Address: 203.174.70.235
```

10. http://www.linuxacademy.ne.jp

11. http://IP アドレス




```
[student@h006 ~]$ traceroute www.kernel.org
traceroute to www.kernel.org (147.75.46.191), 30 hops max, 60 byte packets
 1  gateway (10.20.0.1)  0.295 ms  0.206 ms  0.150 ms
 2  172.30.0.1 (172.30.0.1)  2.910 ms  2.871 ms  2.694 ms
 3  203.174.70.254 (203.174.70.254)  10.957 ms  13.315 ms  13.299 ms
 4  203.174.66.29 (203.174.66.29)  15.889 ms  15.865 ms  15.826 ms
 5  125-6-112-1.data-hotel.net (125.6.112.1)  15.782 ms  15.727 ms  15.682 ms
 6  203.174.64.94 (203.174.64.94)  15.639 ms  12.767 ms 203.174.64.98 (203.174.64.98)
12.695 ms
 7  3257.tyo.equinix.com (203.190.230.95)  15.026 ms  14.422 ms  14.356 ms
 8  xe-1-2-0.cr1-tyo1.ip4.gtt.net (89.149.133.122)  106.077 ms  98.899 ms  99.200 ms
 9  ip4.gtt.net (103.232.18.78)  21.329 ms  9.662 ms  9.903 ms
10  ve5.fr3.tyo3.llnw.net (203.77.184.229)  9.882 ms lag58.fr4.sin.llnw.net
(111.119.23.32)  80.823 ms ve5.fr3.tyo3.llnw.net (203.77.184.229)  10.204 ms
11  ve5.fr3.sin.llnw.net (117.121.248.1)  78.854 ms tge7-1.fr3.sin.llnw.net
(117.121.255.215)  85.896 ms  86.332 ms
12  111.119.23.77 (111.119.23.77)  116.695 ms  90.588 ms  90.519 ms
13  147.75.46.191 (147.75.46.191)  77.199 ms  81.103 ms  75.816 ms
```

第 10 章

10.1.2 練習

```
[root@h006 ~]# fdisk /dev/sda

The device presents a logical sector size that is smaller than
the physical sector size. Aligning to a physical sector (or optimal
I/O) size boundary is recommended, or performance may be impacted.
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


コマンド (m でヘルプ): p

Disk /dev/sda: 80.0 GB, 80026361856 bytes, 156301488 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O サイズ (最小 / 推奨): 4096 バイト / 33553920 バイト
Disk label type: dos
ディスク識別子: 0x000316c4

デバイス ブート    始点         終点     ブロック   Id  システム
/dev/sda1   *    65535       1048559        491512+  83  Linux
/dev/sda2        1048560     11272019       5111730   83  Linux
/dev/sda3        11272020    21495479       5111730   83  Linux
/dev/sda4        21495480   156300974       67402747+   5  Extended
/dev/sda5        21561015    25624184       2031585   83  Linux
/dev/sda6        25689720    26672744       491512+  83  Linux
/dev/sda7        26738280    32833034       3047377+  82  Linux swap / Solaris

コマンド (m でヘルプ): q
```

10.6.2 練習

```
[root@h006 ~]# systemctl get-default
graphical.target
[root@h006 ~]# systemctl set-default multi-user.target
Removed symlink /etc/systemd/system/default.target.
Created symlink from /etc/systemd/system/default.target to
/usr/lib/systemd/system/multi-user.target.
[root@h006 ~]# reboot
```

コンソールから root でログイン

```
[root@h006 student]# systemctl set-default graphical.target
Removed symlink /etc/systemd/system/default.target.
Created symlink from /etc/systemd/system/default.target to
/usr/lib/systemd/system/graphical.target.

[root@h006 student]# systemctl default
     または
[root@h006 student]# init 5
```

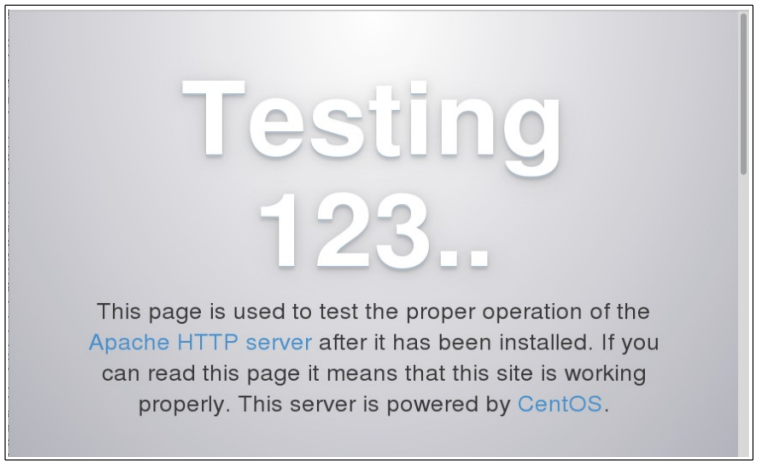
第 11 章

11.1.5 練習

```
[student@h006 ~]$ telnet www.linuxacademy.ne.jp 80
Trying 203.174.70.235...
Connected to www.linuxacademy.ne.jp.
Escape character is '^]'.
HEAD / HTTP/1.0      ←[Enter]は２回

HTTP/1.1 301 Moved Permanently
Date: Sat, 11 Aug 2018 06:55:44 GMT
Server: Apache
Location: http://www.linuxacademy.ne.jp/
Connection: close
Content-Type: text/html; charset=iso-8859-1

Connection closed by foreign host.
```

11.2.4 練習

自身「http://127.0.0.1」へアクセスし、テストページが表示されることを確認。



index.html を新規作成

```
# vi /var/www/html/index.html
# cat /var/www/html/index.html
<html>
<body>Hello, World!</body>
</htl>
```

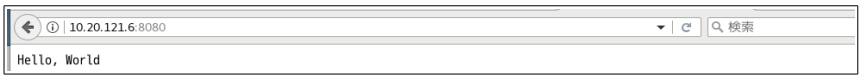
再び自身へアクセス

第 12 章

12.1.2 練習

```
[root@h006 ~]# vi /etc/httpd/conf/httpd.conf

# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
#Listen 12.34.56.78:80
Listen 8080          ← 8080 に変更

[root@h006 ~]# apachectl -t
Syntax OK
[root@h006 ~]# systemctl restart httpd
```

http://自身の IP アドレス:8080/へアクセス

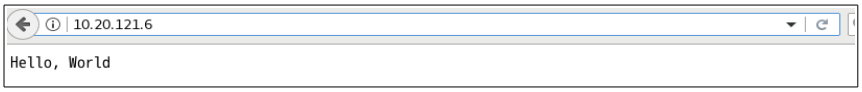



```
[root@h006 ~]# vi /etc/httpd/conf/httpd.conf
     （先の修正箇所を 8080 から 80 に戻す）
[root@h006 ~]# systemctl restart httpd
```

12.1.3 練習　自身の IP アドレスが 10.20.121.6 の場合

```
[root@h006 ~]# vi /etc/httpd/conf/httpd.conf
# Further relax access to the default document root:
<Directory "/var/www/html">
        Order   deny,allow          ←この 3 行を追加
        Deny from all
        Allow from 10.20.121.6
[root@h006 ~]# systemctl restart httpd
```

自身の IP アドレスへアクセス

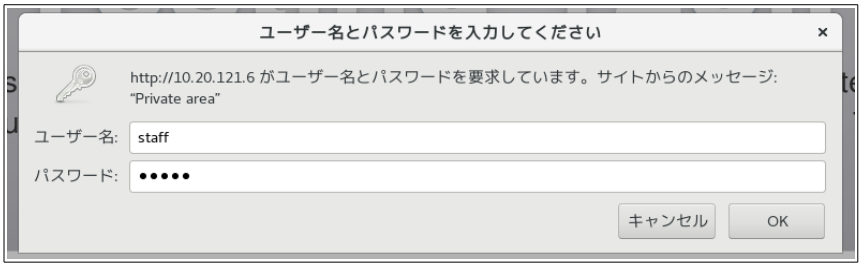


ローカルループバックアドレス（127.0.0.1）へのアクセスは、ページがない扱い。

12.2.3 練習

```
[root@h006 ~]# mkdir /var/www/html/staff
[root@h006 ~]# cd /var/www/html/staff
[root@h006 staff]# vi index.html
[root@h006 staff]# cat index.html
<html>
<body>
     staff only!
</body>
</html>
[root@h006 staff]# vi /etc/httpd/conf/httpd.conf
# Basic authentication sample       ←  ファイルの末尾に以下を追加
<Directory "/var/www/html/staff">
     AuthType    Basic
     AuthName    "Private area"
     AuthBasicProvider       file
     AuthUserFile    /var/www/html/staff/.htpasswd
     Require user    staff
</Directory>

[root@h006 staff]# htpasswd -c .htpasswd staff
New password:
Re-type new password:
Adding password for user staff
[root@h006 staff]# systemctl restart httpd
```

「http://自身の IP アドレス/staff/」へアクセス



ユーザ名「staff」パスワード同を入力し、該当ページが表示される。

第 13 章

13.2.2 練習の回答

```
[root@h006 ~]# vi /etc/httpd/conf/httpd.conf
    ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"      ←この行が有効であることを確認。
[root@h006 ~]# cd /var/www/cgi-bin/
[root@h006 cgi-bin]# vi sample.cgi
[root@h006 cgi-bin]# cat sample.cgi
#! /bin/bash
myname=`whoami`
cat <<EOD
Content-Type:    test/html

<html><body><p>Hello, world!</p>
I am $myname
</body></html>
EOD
[root@h006 cgi-bin]# chmod 755 sample.cgi
[root@h006 cgi-bin]# ./sample.cgi   ←コンソールで実行しエラーがないことを確認
Content-Type:    test/html

<html><body><p>Hello, world!</p>
I am root
</body></html>
```

「http://自身の IP アドレス/cgi-bin/sample.cgi」へアクセス

コンソールで実行した時とユーザ名が違うことに注意

13.3 確認問題

```
[root@h006 ~]# cd /var/www/cgi-bin/
[root@h006 cgi-bin]# vi count.cgi
（テキストの内容を作成）
[root@h006 cgi-bin]# chmod 755 count.cgi
[root@h006 cgi-bin]# echo 0 >  count.dat
[root@h006 cgi-bin]# chown apache:apache count.dat
```

アクセスするたびに、数字が増える。

第 14 章

14.2.1 練習（重複内容は省略）

```
1.
[student@h006 ~]$ nslookup www.kernel.org
Server:         192.168.3.1
Address:  192.168.3.1#53

Non-authoritative answer:
www.kernel.org  canonical name = git.kernel.org.
git.kernel.org  canonical name = sin.git.kernel.org.
Name:     sin.git.kernel.org
Address: 147.75.46.191
2.
[student@h006 ~]$ nslookup -query=mx kernel.org
     (省略)
mail.kernel.org  internet address = 198.145.29.99
ns11.constellix.com  internet address = 96.45.80.1
     (省略)
[student@h006 ~]$ nslookup -query=ns kernel.org
     (省略)
Non-authoritative answer:
kernel.org nameserver = ns41.constellix.net.
     (省略)
Authoritative answers can be found from:
ns11.constellix.com  internet address = 96.45.80.1
     (省略)
[student@h006 ~]$ nslookup xxx.yyy.zzz
     (省略)
** server can't find xxx.yyy.zzz: NXDOMAIN
[student@h006 ~]$ nslookup www.google.com
     (省略)
Non-authoritative answer:    ←直接 google.com を管理していない事を表す。
Name:     www.google.com
Address: 172.217.31.164

[student@h006 ~]$ nslookup -query=ns google.com
     (省略)
Non-authoritative answer:
google.com nameserver = ns4.google.com.
     (省略)
Authoritative answers can be found from:
ns1.google.com  internet address = 216.239.32.10  ←google.com を管理するサーバ
     (省略)
[student@h006 ~]$ nslookup  www.google.com  216.239.32.10
```

```
Server:         216.239.32.10
Address:   216.239.32.10#53
                                    ←「Non-authoritative」が表示されない。
Name:      www.google.com
Address: 172.217.161.68
```

14.2.2 練習

```
[student@h006 ~]$ dig www.ipa.go.jp
; <<>> DiG 9.9.4-RedHat-9.9.4-61.el7 <<>> www.ipa.go.jp
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61348
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.ipa.go.jp.              IN      A

;; ANSWER SECTION:
www.ipa.go.jp.       300    IN      A       202.122.141.45

;; AUTHORITY SECTION:
ipa.go.jp.       53263       IN      NS      ipa-ns.ipa.go.jp.
. . .

;; ADDITIONAL SECTION:
. . .
ipa-ns.ipa.go.jp.    281319  IN      A       192.218.88.1
ipa-ns2.ipa.go.jp.   281319  IN      A       202.229.63.234
     (省略)
[student@h006 ~]$ dig ipa.go.jp NS
     (省略)
;; QUESTION SECTION:
;ipa.go.jp.                 IN      NS

;; ANSWER SECTION:
ipa.go.jp.       47049       IN      NS      ipa-ns.ipa.go.jp.
     (省略)
;; ADDITIONAL SECTION:
     (省略)
ipa-ns.ipa.go.jp.    302106  IN      A       192.218.88.1
ipa-ns2.ipa.go.jp.   302106  IN      A       202.229.63.234
     (省略)
[student@h006 ~]$ dig ipa.go.jp MX
```

```
     (省略)
;; QUESTION SECTION:
;ipa.go.jp.                  IN      MX

;; ANSWER SECTION:
ipa.go.jp.       10800       IN      MX      20 ipa-sfw2.ipa.go.jp.
ipa.go.jp.       10800       IN      MX      10 ipa-sfw1.ipa.go.jp.

;; AUTHORITY SECTION:
ipa.go.jp.       2420 IN     NS      dns-a.iij.ad.jp.
ipa.go.jp.       2420 IN     NS      ipa-ns.ipa.go.jp.
ipa.go.jp.       2420 IN     NS      ipa-ns2.ipa.go.jp.

;; ADDITIONAL SECTION:
ipa-sfw1.ipa.go.jp.  10800   IN      A       192.218.88.2
ipa-sfw2.ipa.go.jp.  10800   IN      A       202.229.63.236
dns-a.iij.ad.jp. 73670       IN      A       210.130.1.17
dns-a.iij.ad.jp. 73670       IN      A       202.232.2.16
ipa-ns.ipa.go.jp.    395319  IN      A       192.218.88.1
ipa-ns2.ipa.go.jp.   395319  IN      A       202.229.63.234
     (省略)
```

14.4 確認問題(一部省略)

```
[student@h006 ~]$ dig @202.12.27.33 jp. NS

; <<>> DiG 9.9.4-RedHat-9.9.4-61.el7 <<>> @202.12.27.33 jp. NS
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 38846
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 8, ADDITIONAL: 16
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;jp.                  IN      NS

;; AUTHORITY SECTION:
jp.              172800      IN      NS      f.dns.jp.
     (省略)
jp.              172800      IN      NS      d.dns.jp.

;; ADDITIONAL SECTION:
a.dns.jp.        172800      IN      A       203.119.1.1    ←.jp管理サーバー
     (省略)
```

```
h.dns.jp.       172800      IN      AAAA    2a01:8840:1ba::25
     (省略)
[student@h006 ~]$ dig @203.119.1.1 linuxacademy.ne.jp. NS
     (省略)
;; QUESTION SECTION:
;linuxacademy.ne.jp.         IN      NS

;; AUTHORITY SECTION:
linuxacademy.ne.jp.  86400   IN      NS      zns02.datahotel.ne.jp.
linuxacademy.ne.jp.  86400   IN      NS      zns01.datahotel.ne.jp.

;; ADDITIONAL SECTION:
zns01.datahotel.ne.jp.       86400   IN      A       203.174.65.4
zns02.datahotel.ne.jp.       86400   IN      A       203.174.65.68
     (省略)
[student@h006 ~]$ dig @203.174.65.4 www.linuxacademy.ne.jp
     (省略)
;; QUESTION SECTION:
;www.linuxacademy.ne.jp.            IN      A

;; ANSWER SECTION:
www.linuxacademy.ne.jp.      3600    IN      A       203.174.70.235

;; AUTHORITY SECTION:
linuxacademy.ne.jp.  3600    IN      NS      zns01.datahotel.ne.jp.
linuxacademy.ne.jp.  3600    IN      NS      zns02.datahotel.ne.jp.

;; ADDITIONAL SECTION:
zns01.datahotel.ne.jp.       3600    IN      A       203.174.65.4
zns02.datahotel.ne.jp.       3600    IN      A       203.174.65.68
     (省略)
```

15.2.1 練習

```
[root@h006 named]# named-checkzone s142.la.net. /var/named/name.s142.la.net
zone s142.la.net/IN: loaded serial 2017080702
OK
```

15.2.2 練習

```
[root@h006 named]# systemctl start named
[root@h006 named]# systemctl status named
● named.service - Berkeley Internet Name Domain (DNS)
   Loaded: loaded (/usr/lib/systemd/system/named.service; disabled; vendor preset:
disabled)
   Active: active (running) since 土 2018-08-11 17:55:02 JST; 5s ago
  Process: 2529 ExecStart=/usr/sbin/named -u named -c ${NAMEDCONF} $OPTIONS
```

```
(code=exited, status=0/SUCCESS)
  Process: 2527 ExecStartPre=/bin/bash -c if [ ! "$DISABLE_ZONE_CHECKING" == "yes" ];
then /usr/sbin/named-checkconf -z "$NAMEDCONF"; else echo "Checking of zone files is
disabled"; fi (code=exited, status=0/SUCCESS)
 Main PID: 2531 (named)
   CGroup: /system.slice/named.service
           └─2531 /usr/sbin/named -u named -c /etc/named.conf

 8月 11 17:55:02 h006.s143.la.net named[2531]: zone s142.la.net/IN: loaded ...1
 8月 11 17:55:02 h006.s143.la.net named[2531]: zone 1.0.0.0.0.0.0.0.0.0.0.0...0
 8月 11 17:55:02 h006.s143.la.net named[2531]: zone localhost.localdomain/I...0
 8月 11 17:55:02 h006.s143.la.net named[2531]: zone localhost/IN: loaded se...0
 8月 11 17:55:02 h006.s143.la.net named[2531]: all zones loaded
 8月 11 17:55:02 h006.s143.la.net named[2531]: running
 8月 11 17:55:02 h006.s143.la.net named[2531]: error (network unreachable) ...3
Hint: Some lines were ellipsized, use -l to show in full.
[root@h006 named]# dig h006.s142.la.net @127.0.0.1
     (省略)
;; QUESTION SECTION:
;h006.s142.la.net.          IN     A

;; ANSWER SECTION:
h006.s142.la.net.    86400  IN     A      10.20.142.6

;; AUTHORITY SECTION:
s142.la.net.         86400  IN     NS     ns.s142.la.net.

;; ADDITIONAL SECTION:
ns.s142.la.net.      86400  IN     A      10.20.142.6
     (省略)
[root@h006 named]# dig NS s142.la.net @127.0.0.1
     (省略)
;; QUESTION SECTION:
;s142.la.net.               IN     NS

;; ANSWER SECTION:
s142.la.net.         86400  IN     NS     ns.s142.la.net.

;; ADDITIONAL SECTION:
ns.s142.la.net.      86400  IN     A      10.20.142.6
     (省略)
```

補足

ダウンロード

本テキストの補足的な内容は、以下のサイトからダウンロードできます。

コマンド一覧

http://s.linuxacademy.ne.jp/linuxbasic_apdx2.pdf

シェルスクリプト

http://s.linuxacademy.ne.jp/linuxbasic_apdx3.pdf

BIOS 操作例

LA 環境の PC（Lenovo ThinkPad T430）での BIOS 調整例

1. 「Startup Interrupt Menu」の呼び出し
   電源を投入し、「ThinkPad」ロゴの下に「To interrupt normal setup, press Enter」が表示されたら［Enter］を押し続ける。

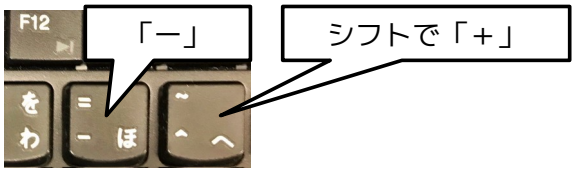
2. BIOS 呼び出し
   「Startup Interrupt Menu」が表示されたら［F1］を押す。
   「F1 to enter the BIOS Setup Utility]

3. ブートシーケンスの調整
   ［←］［→］キーで、タブを移動し「Startup」を選択
   「▶Boot」を［↑］［↓］で選択し［Enter］を押す

4. 順序の調整
   装置一覧から［↑］［↓］で選択し、［+］［-］で順序を前後させ、以下の順位並び替える。

   ```
   1. ATAPI CD0  内蔵 DVD    [+]
   2. USB HDD        外付 HDD        ↑
   3. USB CD         外付 DVD        ↓
   4. ATA HDD0   内蔵 HDD    [-]
   ```

   ただし BIOS は英語キーボード配列となっており、+/-の配置が異なるので注意。

5. 再起動
   ［F10］で内容を保存後、BIOS を終了する。