

序 論

昨今、人工知能（機械学習とビッグデータ処理）関連の技術の進展が目覚しく、人間の行う職業がなくなりつつあるように見受けられる。例えば、自動運転技術が進展することで、バスやトラック、タクシーなどの職業が衰退することも考えられる。エンジニア職も例外ではない。ネットワークエンジニアは将来なくなる職業にとりあげられることもある。しかしながら、人工知能技術は、明確なルールがあるものや、単調な作業などはすぐに置き換えることができるが、複雑なものになればなるほど、機械学習アルゴリズムでは学習ができない。一方、ネットワーク技術が様々な技術の集合体であり、場所によって異なる形で構築がされるような複雑なシステムである。全ての機器が設計通りに動いているのならばまだしも、停電や経年劣化、障害等で一部の機器が動作停止や誤動作するようなこともある。こうした複雑かつ困難な状況に、即座に柔軟に対処できるようなAIはまだ見当たらない。このような観点から考えると、ますます複雑になるネットワークシステムを支える技術者は、今後しばらくは（少なくとも我々が職業を持って働く間は）なくなることはないであろう。

さらに言えば、Facebook、Twitter、LINEなどのSNS、Google、Dropbox、Amazon、Microsoft等の様々なサービスやクラウド技術など、インターネットサービスが発展し、多くの人々が利用している。これらは、インターネットのインフラが稼働していないと全く利用できない。すなわち、これらに障害が起きると生活に支障が出るくらい、重要な生活のインフラストラクチャ（基盤）になっていると言え、これらを支えることは現代では電気・エネルギー・水道、運輸インフラを支えると同じくらい重要なことである。

また、セキュリティも重要な事項であり、他のインフラと異なり、インターネットインフラはサイバー攻撃による影響がとても大きく、一つの企業や国の生活に影響を与えるほどである。実際に、通常の軍隊と同様に、サイバー軍を創設し他国への攻撃、他国からの攻撃に備える国も増えている。さらには、先ほどの人工知能も人々のインターネットの利用が進み、さらにはIoT (Internet of Things)などで多くの機器がインターネット接続され、多くのデータが蓄積されたことで、はじめて人工知能が構築可能になったとも言える。

以上のことから、インターネット技術を実際に体験し習得することは、今後の情報技術者にとって重要なことであるのは間違いない。この実験では、こうしたインターネット技術を支える技術について、実際に構築をしながら、幅広い角度からネットワーク技術を学ぶ。本実験で扱う技術は、必ずしも最新のサービスではなく、今後も長く使われ続ける基礎技術に重点を置いている。UNIX系OS(Ubuntu Linux, CentOS), Windows, Macintoshといった広く普及しているオペレーティングシステムを用いて、それぞれのコンピュータのネットワーク設定ならびに、サーバでのWWWやデータベースシステム、暗号化など、身近に使われているネットワークサービスの構築を行う。さらに、ルーティング、ネットワーク設計、スイッチ設定、VLAN、セキュリティといったインターネットを支える基盤技術の構築を、ネットワーク業界で広く使われるCisco IOSを実際に用いて学ぶ。IOSはCCNA試験など、ネットワーク業界で認められている資格試験にも使われており、多くのベンダがIOS互換のコマンド体系を採用していることからネットワークを学ぶのに最も適したシステムといっても良い。

本実験の内容は初学者にとっては十分に複雑で難しいものであるかも知れない。しかし、本実験の内容はまだまだ基礎であり、この先にはさらに複雑で設計の難易度の高いシステムが数多く存在している。ぜひ、この実験を通して基礎技術、基盤技術を身に付け、将来、更なる発展技術の習得、および新技術、新サービスの開発へと進んでいき、インターネットの発展に寄与して欲しいと考える。

全ての専攻の学生が、このテキストと実験での経験をもとに、ネットワーク技術の仕組みを理解し、様々な業界で活躍することを心より期待し、巻頭の言とする。

2019年4月13日
情報学群実験3i・4C教育担当スタッフ一同

レポートについて

レポートは、下記の体裁にて提出すること。全てのレポートについて、1つでも下記に該当する場合、採点および単位認定されないので注意すること。

- 指定された期日・時刻までに提出されない場合
- 指定された様式でない場合、特に抜けている項目がある場合

加えて、下記の場合は、単位認定されない上に、不正行為として処罰される行為であるので、厳に慎むこと。

- 他人のレポートからの転載
- 他の Web ページ、本などからの転載

本科目のレポートは、全て、KUT LMS (Learning Management System, <https://lms.kochi-tech.ac.jp>) の本科目の項目から PDF にて提出する。提出されたレポートは、Turnitin システムにより、過去も含む他人のレポート、Web ページ、出版された本などと一致する文章があるか照合され、剽窃チェックが行われる。

LaTeX (platex) にて、実験の冒頭で指定されたテンプレートを用いて作成すること。

章立ては下記の通りとすること。下記の項目に従っていないもの、一部の項目に抜けがある不完全なものは、提出されても受理しない。

1. **目的** それぞれの回で、どのようなことができるシステムを構築するのかと、なぜそのシステムが必要であるか、構築することでどのような機能・サービスが実現できるのかを述べる。ただし、自らの学習目的、学ぶ内容を説明しないように注意する（○○を習得する、学ぶ、などと書かない）（2点満点）。
2. **内容** 構築するシステムの技術的な説明（どのような要素技術を用いて、どのような仕組み・原理で実現するのか）を、システムの全体像が分かるような説明図を1つ以上含めて説明する。図を用いて、実際の挙動の例などを説明する（2点満点）。
3. **要素技術** システムで用いられる要素技術について、指定されたものについて、それがどのようなものであるか、技術的背景、仕組み、構造、動作について説明する。ただし、あくまで技術の解説をし、単なる用語説明にならないように注意する。また、適切な参考文献を挙げて根拠を示すこと（2点満点）。
4. **作業記録** システムを構築するために、具体的にどのような作業をしたかを記録する。用いたソフトウェア、行った操作、コマンド、設定した内容、編集したファイルなどをもれなく書く。システムが壊れた場合など、一から構築するときも、記述された作業を行うことで、全く同じシステムを構築することができるよう書く（2点満点）。
5. **考察** 考察のポイントを踏まえて、取り扱うシステム・技術の利点や欠点、その技術の本質、類似システムとの相違・相似などの比較・考察を行い、自分なりの考え、結論を書く（結論は感想ではない。○

○という技術は△△の一種であるから本質的にこのような問題を抱えている。ゆえに、将来□□を改良・解決した◇◇などの研究・開発が重要である、など) (2点満点)。

参考文献 第1節から第5節までで述べたことのうち、自ら考案したオリジナルの意見でなければ(あるいは考案したことであっても、過去同じことが発明・提案されているならば)、その内容について述べている文献を適切に挙げること。記述の例として、下記のように書くこと。**最低5つ程度の参考文献は挙げること。**また、一般的のWebページの引用は原則できない。文中で適切に引用すること。参考文献がほとんどないものは項目抜けとして扱う。

- 本を参考にする例

(例) 佐伯サチオ、福富エイジ著、福本昌弘監修、吉田真一訳:「情報実験の方法論」, pp. ○～○ (どのページに書いてあるか), KUT出版, 2012 (発行年)。

- 雑誌を参考にする例

(例) 吉田真一:「情報実験の方法論 (記事名)」, ○○雑誌名, ○巻, ○号, pp. ○～○ (ページ to ページ), KUT出版, 2012 (発行年)。

- 英文等の例

(例) Shinichi Yoshida, Sachio Saiki, and Eiji Fukutomi, “Computer Network,” Journal of KUT, Vol. 3, No. 4, pp. 501-511, 2012.

2種類のダブルクォーテーション、始まり「“」(LaTeXでは、バッククォーテーション‘’を2つタイプ‘‘’)と、終わり” (LaTeXでは、シングルクォーテーション‘’を2つタイプ，’)に注意し、項目の区切りには半角コンマ「,’」を使い、コンマの後ろには必ず半角空白を1つタイプする。終わりのダブルクォーテーションの後に項目を続ける場合は、コンマはダブルクォーテーションの前に入れ、最後は半角ピリオド「.’」で終わる。

(付録や感想) その他、気がついたことで報告しておきたいことを付録として報告しても良い。また感想を付けても構わない。ただし、節番号を付けないこと。ただし、「付録」「感想」とも直接的には採点の対象とはならない。また、「付録」「感想」に書かれた内容は、レポートの採点には(良い方にも悪い方にも)影響しない。

忘れがちな点を下記に示す。

- 日本語文の場合、コンマ「,’」やピリオド「.’」は全角を用いる。
- 英文の場合、コンマ「,’」やピリオド「.’」は半角を用い、コンマの後に半角空白を1つ、ピリオドの後に半角空白を2つタイプする。
- 意味的に1つのトピックごとに段落分け(パラグラフ)し、パラグラフの区切りには、*LATEX*の場合、空行を1行入れると、自動的に次段落1行目を1文字インデントするので、これを用いる。**強制改行「\\」あるいは「\YY」は段落分けには用いず、段落の1行目の最初にも全角空白「 ’」は挿入しない。**

分量は特に指定しないが、目的、内容、考察の各項目を十分に説明するには、それぞれ、数百字程度は必要であると期待される。また、要素技術についても、それぞれの技術項目について数百字程度は期待される。

著作権 (Copyright) やその他の知的財産権 (Intellectual Property), の処理については、十分に注意をすること。具体的には、著作者の許可を得ずに本・教科書・Web ページ等の記述・図・表の転載などを行わないようにすること。

また、倫理上の問題やコンプライアンス（法令、政令等の法律や国の規定、通達、学則や「高知工科大学における研究活動の不正行為への対応等に関する規程」等の学内規則、判例等の司法判断、一般的な常識（社会通念上の規範）の遵守）についても十分留意すること。具体的には、不正と認定されるような行為（例えば他人のレポート、文書、本、雑誌、Web からの転載）などを行わないこと。

最初にも述べたが、他の文書（インターネット上の情報や他人のレポート等）から文書の転載は、上記に違反するものであり、Q未試験におけるカンニング等の不正行為と同様のものとして扱う。

情報学群実験第3i・4C

目 次

第I部 情報学群実験第3i・第4C	1
<hr/>	
第1章 オペレーティングシステムの導入とネットワーク設定	3
1.1 目的	3
1.2 実験内容の概要	3
1.3 OS の基本的機能	5
1.4 OS の種類	6
1.5 実験内容 (1)	8
1.6 必要となる知識	9
1.6.1 ハードウェア	9
1.6.2 キーボード・マウス接続	9
1.6.3 キーボードの配列	10
1.6.4 ディスプレイインターフェース	10
1.6.5 ユーザID・グループID	11
1.6.6 デバイスファイル	11
1.6.7 パーティション	12
1.6.8 サーバOSの操作に必要なコマンド	12
1.6.9 ユーザアカウントの作成とパスワードの設定	13
1.7 必要な情報	14
1.7.1 サーバOS	14
1.7.2 クライアントOS	14
1.8 OS のインストール手順	15
1.8.1 Ubuntu	15
1.9 考慮すべき点	18
1.10 実験内容 (2)	19
1.11 必要となる情報	19
1.11.1 各機器のネットワーク情報	19
1.12 LAN の構成機器	21
1.13 必要となる知識	24
1.13.1 サーバのネットワーク設定	26
1.13.2 Netplan によるサーバのネットワーク設定	31
1.14 ネットワーク用ソフトウェアのインストール	33
1.15 SSH でのログイン, SCP でのネットワークファイルコピー	33

1.16 動作確認	34
1.17 考慮すべき点	35
1.18 実験内容 (3)	36
1.19 各 OS のアップデートの方法	36
1.20 考慮すべき点	39
第 2 章 WWW システムと認証および暗号化	41
2.1 WWW システム	41
2.1.1 HTML	42
2.1.2 HTTP	42
2.1.3 URI	42
2.1.4 HTTP におけるリクエストとレスポンス	43
2.1.5 Web サーバの種類	44
2.2 WWW における認証と暗号化	45
2.2.1 Basic 認証と Digest 認証	46
2.2.2 接続元による認証	46
2.2.3 SSL および TLS	46
2.3 暗号方式	47
2.3.1 共通鍵暗号	48
2.3.2 公開鍵暗号	48
2.4 認証手法	48
2.4.1 電子署名	48
2.4.2 証明書と認証局	50
2.5 実験内容 (1)	51
2.6 実験内容 (2)	52
2.6.1 HTTPS サーバの設定	52
2.6.2 HTTPS 通信のモニタ	52
2.6.3 動的コンテンツ	53
2.7 Apache のインストールと各種設定	53
2.7.1 Apache のインストール	53
2.7.2 Apache の動作に関連するディレクトリ	54
2.7.3 コンテンツの公開	55
2.7.4 HTTP 認証 (Basic) の設定	55
2.7.5 IP アドレスによるアクセス制限の設定	56
2.7.6 HTTPS による暗号通信設定 (modssl)	57
2.8 動作確認	57
2.9 考慮すべき点	58
第 3 章 ルータ・スイッチ	61
3.1 目的	61
3.2 TCP/IP ネットワークのルーティング	62
3.2.1 クラスフルルーティング	63
3.2.2 クラスレスルーティング	64
3.2.3 ルーティングテーブルの構築の種類	64

3.3 実験内容 (1)	65
3.4 必要となる知識	67
3.4.1 コンソール	67
3.4.2 Cisco ルータ, スイッチへの接続	69
3.4.3 Cisco IOS ルータの設定方法	69
3.4.4 PC の IP アドレスを変更	73
3.4.5 ルーティングテーブルの登録	74
3.5 ルーティングの設定	74
3.5.1 コンピュータのネットワーク設定の変更	75
3.6 動作確認	75
3.7 考慮すべき点	76
3.8 スイッチを用いる目的	77
3.9 VLAN の種類	78
3.10 実験内容 (2)	79
3.11 設定に必要な情報	81
3.12 作業内容	81
3.12.1 スイッチの設定	81
3.12.2 トランク VLAN (4Cのみ)	86
3.12.3 Spanning Tree 無効化 (4Cのみ)	87
3.13 動作確認	87
3.14 考慮すべき点	87
第 4 章 DNS・メールサービス	89
4.1 目的	89
4.2 DNS とは	89
4.3 実験内容 (1)	90
4.4 必要となる情報	90
4.5 必要となる知識	91
4.6 DNS サービスの構築の手順	96
4.6.1 BIND のインストール	96
4.6.2 BIND の全体設定	96
4.6.3 ドメイン（ゾーン）の設定	98
4.6.4 本実験での BIND ゾーンファイル例	100
4.6.5 DNS サーバの起動と動作確認	100
4.6.6 サーバ OS での参照先 DNS サーバ(リゾルバ)設定	101
4.6.7 クライアント OS のリゾルバ設定	102
4.7 発展：Macbook, ChromeBook の固定 IP 設定	102
4.8 動作確認	102
4.9 考慮すべき点	103
4.10 電子メールの送受信	104
4.11 実験内容 (2)	104
4.12 必要となる情報	105
4.13 必要となる知識	105
4.14 postfix のインストール	108

4.14.1 DNS の設定の追加	110
4.15 MUA の設定, および, postfix, dovecot のテスト	111
4.16 考慮すべき点	112
第 5 章 データベース・Web システム	113
5.1 データベース	113
5.1.1 データ形式	113
5.1.2 ハッシュ (Hash)	114
5.1.3 リレーションナルデータベース (RDB)	114
5.1.4 SQL	115
5.1.5 代表的な RDBMS	115
5.2 Web システム	116
5.2.1 動的コンテンツ	116
5.2.2 Wiki	117
5.2.3 ブログ	117
5.2.4 CMS (Content Management System)	118
5.2.5 SNS (Social Networking System)	118
5.3 実験内容 (1)	119
5.4 必要な知識	121
5.4.1 文字コード・改行コードの種類	121
5.4.2 日本語 UTF-8 を扱うための手順	123
5.4.3 MySQL インストール	124
5.4.4 データベース構築の手順	124
5.4.5 SQL 言語によるデータベース操作	129
5.5 実験内容 (2)	131
5.6 必要となる知識	132
5.6.1 SHELL の環境変数による言語設定	132
5.6.2 PHP の MySQL 拡張のインストール・有効化	132
5.6.3 MediaWiki のインストール	133
5.6.4 WordPress のインストール	134
5.7 デバッグ	136
5.8 動作確認	137
5.9 考慮すべき点	137
5.10 実験内容 (2')	138
5.10.1 HTTPS サーバの設定～認証局による証明書の署名 (これ以降は実験 4C のみ)	138
5.10.2 Apache のサーバ証明書の作成	139
5.10.3 証明書の作成	140
5.10.4 証明書ファイルの配置	141
5.10.5 注意点	141
第 6 章 ネットワークセキュリティ	143
6.1 ネットワークセキュリティ	143
6.1.1 ファイアウォール	143
6.1.2 NAT	145

6.1.3 ポートフォワーディング	147
6.2 実験内容 (1)	148
6.3 必要となる知識	148
6.3.1 IOS によるパケットフィルタリング	148
6.3.2 NAPT の設定	152
6.3.3 NAPT の設定例	152
6.3.4 ポートフォワーディングの設定例	153
6.3.5 設定確認	153
6.4 動作確認	154
6.5 考慮すべき点	155
第 7 章 最終課題	157
7.1 目的	157
7.2 内容	157
7.3 実験内容 (3 回分)	158
7.3.1 仕様	158
7.3.2 AWS EC2 インスタンスの準備	159
7.3.3 AWS EC2 インスタンスにログイン	159
7.3.4 パスワード等の注意	159
7.3.5 LAMP のインストール	159
7.3.6 セキュリティ (ファイアウォール) の設定	160
7.3.7 DNS サーバの設定	160
7.3.8 LAMP 設定の続きと WordPress	162

第 II 部 付 錄 編 163

付録 A 章 vi エディタの使い方	165
A.1 vi の起動	165
A.2 vi のモード	165
A.3 コマンドモード	165
A.3.1 カーソルの移動	166
A.3.2 削除, 複写, 移動	166
A.3.3 検索, 置換	167
A.3.4 コマンドの取り消しと再実行	167
A.3.5 ファイルの読み書き	167
A.3.6 vi の終了	167
A.4 モードの切替え	168
付録 B 章 UNIX コマンド集	169
B.1 chgrp (CHange GRouP)	169
B.2 chmod (CHange MODE)	170

B.3 chown (CHange OWNer)	172
B.4 df (Disk Free) / du (Disk Usage)	173
B.5 grep (Global Regular Expression Print)	174
B.6 more, less, lv	175
B.7 tail	177
B.8 ip, ifconfig (InterFace Configuration) , ipconfig	178
B.9 netstat (NETSTATUs)	180
B.10 nslookup	181
B.11 dig	182
B.12 ping	185
B.13 traceroute, tracert	186
B.14 route(ROUTE)	187
B.15 shutdown(SHUTDOWN)	188
B.16 kill	190
B.17 ps (Process Status)	192
B.18 mail	194
B.19 man	196
B.20 telnet	197
B.21 tar (Tape ARchive)	198
B.22 make	199
B.23 patch	200
B.24 su(Switch User)	201

付録 C 章 Cisco IOS コマンド集 203

C.1 IOS の基本操作	203
C.2 ip route	208
C.3 show ip route	209
C.4 show mac-address-table	211
C.5 show arp	212
C.6 show vlan	213
C.7 show interface status	214
C.8 show ip interface brief	216
C.9 show ip nat translations, ip nat	217

第I部

情報学群実験第3i・第4C

第1章

オペレーティングシステムの導入とネットワーク設定

1.1 目的

基本ソフトウェア（オペレーティングシステム（Operating System）：OS）は、様々なアプリケーションソフトウェアで共通に必要とされる処理を提供するプログラムである。現在では、ネットワークサービスを含む全ての応用ソフトウェア（アプリケーションソフトウェア、Application Software）¹は、何らかの基本ソフトウェア上で動作するよう開発されているので、これらを用いる場合は、そのアプリケーションソフトウェアを動作させることのできる適切なOSが必要がある。

のことから、ネットワークサービスを構築する際は、まず構築するサービスに適したOSを選定し、適切にインストール（導入作業）を行い、OSが提供するネットワーク機能（プロトコルスタック）を適切に設定した後、必要なネットワークサービスを提供するアプリケーションソフトウェアをインストールするという手順を踏む必要がある。

1.2 実験内容の概要

下記を構築する。

- ネットワークサーバを構築するための基盤
- オンプレミス（実機）²でサーバOSをクリーンインストール
- クラウド³（仮想）上で同様に基盤をデプロイ⁴

ネットワークサービスに関する応用ソフトウェアを実行する環境を構築するためにOSをクリーンインストールし、ネットワーク設定を行う。クリーンインストールとは、インストールするコンピュータのストレージ（補助記憶装置、ハードディスクドライブ（HDD）やソリッドステートディスク（SSD）のこと）を完全に初期化された状態から、OSの開発元が提供するインストール用のメディア（CD-ROM/DVD-ROM/USBメモリ）を用いてOSをインストールすることである。クリーンインストールではない方法として、既に購

¹ 応用ソフトウェアは、コンピュータ利用者が必要とする具体的な処理を行うものである。例えば、文書の作成、編集を行うワードプロセッサ、表形式のデータに対してデータ間の計算や集計を行う表計算ソフト、Webサービスを提供するWebサーバなどである。

² オンプレミス（On-premises）：店内、学内、構内に導入すること

³ Cloud：インターネットを空の雲の中に見立てて、雲のどこにあるかは意識せずに使いたいときに、使いたいだけ使えるようにするコンピュータ（のリソース（資源））

⁴ デプロイ（deploy）：配置=使えるようにすること

入時にインストールされていた OS や古い OS に対してアップグレードや消去せずに上書きする形でインストールするものがある。

まず、OS をクリーンインストールするために、下記を行う。

- (1) OS インストール用メディアの準備およびブート
- (2) ハードディスクの初期化
- (3) ユーザアカウントの作成
- (4) その他のユーザ登録情報の設定

USB メディアからの起動 (ブート:boot) クリーンインストールを行うためには、ハードディスクの初期化が必要になるために、インストール作業中はハードディスクを利用できないため、インストールおよびブート用のメディアを別途準備し、それを用いてブートする。

パーティション分割 (partitioning)・フォーマット (format) ハードディスクの初期化とは、具体的にはハードディスクをいくつかの領域に用途別に分割する設定、および OS がどのパーティションを使用するかの指定を行い、そのパーティションに OS が用いるファイルシステムを作成する。ファイルシステムとは OS のハードディスクを利用するためのフォーマット (形式) であり、その形式で使えるようにする初期化作業をフォーマットと言う。

ユーザアカウントは、最低1つの管理者用アカウントが必要であり、更に使用するユーザ毎にアカウントを設定する必要がある。アカウント設定を行うことで、複数のユーザが同じ端末を共有することができる(電子メールやブラウザのブックマーク等の設定を個々で行える)。

次に、TCP/IP ネットワーク設定を行うために、次のことを行う。

- ネットワークインターフェースの設定

ネットワークインターフェースの設定に先だって、ネットワークインターフェースハードウェアの存在が OS から認識されていない場合は、そのハードウェアのメーカーが提供するデバイスドライバをインストールする。

ネットワークインターフェースの確認は以下のコマンドを使う。

Linux の場合

```
# ip address (ip a と省略可)
```

古い Linux、その他の UNIX (macOS 含む) の場合

```
# ifconfig -a
```

以上のコマンドで、「eth0」や「enp2s0」、「en0」(macOS)など、インターフェース名と呼ばれる文字列から始まる行があれば良い。「e」は有線 LAN の Ethernet からきている。なお、「lo」などは、ループバックインターフェース (loopback) で、すべての TCP/IP に対応した OS が持つ仮想的なインターフェースであり、それがあっても直ちにネットワークが有効であるとは判断できない(IP アドレスは、IPv4 は 127.0.0.1, IPv6 が ::1)。

その後、ネットワークインターフェースに対して、IP アドレス、ネットマスク、デフォルトゲートウェイ、DNS サーバアドレス、プロキシサーバのアドレスとポートを指定する。

最後に、OS のアップデート作業をネットワーク経由で行う。OS は、不具合やセキュリティ上の問題が現れた場合、メーカからインターネット経由での更新プログラムが提供される。OS のインストール時点で、

様々な不具合やセキュリティ上に問題が明らかになっていることも多いため、インストール後、すぐに最新の状態まで更新することが重要である。

1.3 OS の基本的機能

OS の役割は、基本的な機能をアプリケーションに提供することである。更に現在ではマルチタスク OS と呼ばれ、複数のアプリケーションを同時に動作させるので、キーボードやディスク、画面などのデバイス（ハードウェア）資源（リソース）を管理し、複数のアプリケーションソフトウェアの間のデバイスが共用できるよう、その利用の調停を行うことも OS の役割である。

現代の OS では、特に重要なデバイス管理として下記の 3 つが挙げられる。

- メモリ（主記憶）管理（実メモリとスワップファイルによる仮想メモリ）
- プロセス管理（マルチタスク TSS）
- ストレージ（補助記憶）管理（ファイルシステム）

加えて、現在の OS は複数のユーザが同時に使うものであるので、ユーザ管理（権限管理）がある。これはセキュリティにも密接に関連し、セキュリティインシデントは、しばしばこの権限管理が想定外になった場合に引き起こされる。

- ユーザ管理（マルチユーザ）

さらに、それに加えて、下記のネットワーク接続、すなわち LAN⁵ および TCP/IP⁶ 接続のサポートも、近年の OS では重要な機能の一つである。OS のうちネットワークプロトコルを提供する部分をプロトコルスタックと呼ぶ。

- ネットワーク管理

その他、ディスプレイやプリンタ、キーボードやマウス、タッチパネルなどの、入出力装置の管理も行う。クライアントユーザが主な利用者となるクライアント OS では、利用者の利便性向上のための統一的な GUI (Graphical User Interface) デスクトップ、ビデオ・グラフィックス・サウンド機能の提供などがある。Apple 社の Mac OS X や iOS、Microsoft 社の Windows、Google の Android などは、GUI 部分も OS ベンダが提供し、別の独自 GUI を適用することはできないが、Linux や BSD 系 OS などは、OS 本体には GUI 機能は含まれておらず、GUI 用のソフトウェアを用意することで使うことができる。GUI 用ソフトウェアにもいくつかの種類があり、X Window System が標準的に用いられる。X Window System には、基本的なウィンドウ管理機能しか含まれておらず、デスクトップ用途の機能（例えば、アイコン表示や、ダブルクリックで起動する機能、タスクバーやドラッグアンドドロップ）は、さらに別のソフトウェアを用いる。よく用いられるものは、GNOME と KDE であるが、それ以外にもウィンドウマネージャと呼ばれるソフト群が数多く存在し、ユーザの好みで切り替えることができる。

つまり、アプリケーションに共通に必要とされる、資源管理、入出力、ネットワークなどの基本機能を提供するのが OS の役割である。

⁵ Local Area Network

⁶ Transmission Control Protocol/Internet Protocol

1.4 OSの種類

本節では、より具体的にOSの種類、機能について述べる。

OSは、コンピュータのアーキテクチャ(CPUやメモリ、周辺装置の構造、駆動方法)に依存して様々なものがある。たとえば、図1.1に示すように、コンピュータにはそれぞれに適したOSが用いられている。

表1.1 コンピュータとOS

コンピュータ	OS
パーソナルコンピュータ (Apple Macintosh)	macOS
パーソナルコンピュータ (Intel CPU PC/AT)	Microsoft Windows, Linux, UNIX系OS, ChromeOS
スーパーコンピュータ (Intelまたは独自CPU)	Linux, UNIX系OS, 独自OS
スマートフォン	Android(Linuxベース), iOS, 独自OS

以下にパーソナルコンピュータやネットワークサーバでよく用いられるOSを挙げる。

- UNIX系OS

 - Linux

Linuxは、Linus Torvaldsがフィンランドの大学院生であった頃に開発されたOSで、当時有償であったUNIXと同じ機能を持つ互換OSとして作成された。OSの基本的機能を提供するカーネル部分のみが開発され、それ以外の、シェル・コマンド・ライブラリ・コンパイラ等は、MITのGNUプロジェクトのものを用いる。カーネル以外の部分の組み合わせ方は、OSをインストールするものが様々なものを選択し、アプリケーションソフトウェアも様々であるので、それらをパッケージにまとめた、全体のインストールパッケージが、100種類以上ある。これをディストリビューションと呼ぶ。大きく、Debian系、RedHat系、Slackware系ディストリビューションに分かれる。ディストリビューション間の互換性があるものもあるが、ない場合も多い。

 - * Ubuntu

UbuntuはUNIX系OSの一つであるLinuxのディストリビューション⁷の一つで、Debian系⁸のディストリビューションである。

 - * CentOS

Cent OSは、RedHat系ディストリビューションの一つで、サーバのプラットフォームとしてよく用いられる有償のRedHatに互換のものを、無償のオープンソースで作成しているものであり、多くのサーバのプラットフォームとして用いられる。

高性能コンピュータを超並列に配置したスーパーコンピュータのOSとして、現在ではしばしば用いられるほか、スマートフォン用OSのAndroidのベースとして用いられたり、Googleのサービスに用いられるなど、大規模なものから組み込みまで、様々な規模の計算に用いられている。

⁷ Distribution: Linux OSの配布形態には数多くの種類があり、Ubuntuはその一つである

⁸ Linux系、Debian系、Slackware系があり、それぞれrpm(RedHat Package Manager)、deb、tgzという別の仕組みのパッケージ管理システムを採用している。

- FreeBSD

FreeBSD は UNIX 系 OS の一つで、現在のサーバにしばしば用いられる OS の一つである。 FreeBSD には、定期的に一般に公開されるバージョンである RELEASE と 2 つの開発用のバージョン (CURRENT と STABLE) とがある。元来、UNIX には、BSD 系と SystemV 系との 2 つの系統があり、それぞれ、カリフォルニア大学バークレイ校、AT&T (アメリカの電話会社) で開発された。BSD は、TCP/IP の開発・実装に用いられ、現在のインターネットの基盤技術の開発に大きく貢献した。現在でも、ネットワークの技術開発によく用いられる他、Linux と並んで大企業や有名ウェブサイトなどのサービス運営に用いられるほか、組み込み機器などにも使われる。

- macOS

macOS は、アメリカのコンピュータメーカーの Apple 社が販売するコンピュータ「Macintosh」に用いられている OS である。元々、Mac OS 1, 2, 3 と続き、9 の次にリリースされた MacOS X は「マックオーエステン」と呼称し、アーキテクチャが大きく更新された（互換性がなくなった）。X には、10 番目という意味がある。Mac OS 9 までと異なり、MacOS X は、全く新規に UNIX (FreeBSD, Mach 等) および NextSTEP をベースに構築された OS である。The Open Group の認証を受けた正式な UNIX OS である。2020 年には、Apple M1 CPU が搭載され、それまでのものとは大きく異なる仕様に変更された。

- Windows

Windows は Microsoft 社が販売している OS であり、1980 年代から開発が行われ、日本では 1993 年に Windows 3.1 が発売されたものが最初である。当時は、Microsoft 社の低機能の CUI (Character User Interface) である MS-DOS 上で動作する OS であり、16 ビット CPU である 8086 系 CPU で動作するものであった。その後、16 ビット CPU 系 OS の後継として、Windows 95, Windows 98, Windows ME (Millennium Edition) が、32 ビット CPU 系 OS として WindowsNT, Windows2000, WindowsXP, Windows Vista が使われた。WindowsXP 以降は、64 ビット CPU 向け OS も発売されるようになり、32 ビット版と 64 ビット版の両方の CPU アーキテクチャ向けに発売されている。Windows Vista, Window 7, Windows 8, 8.1 がリリースされ、2021 年時点では、Windows 10 が最新バージョンである。コンピュータのメモリ容量が 2GB 以上の場合には 64 ビット版を、それより小さい場合は 32 ビット版を用いるのが効率的であるが、使用するアプリケーションソフトやデバイスドライバにより対応状況が異なるため、どちらかを使わなければならない場合もあり、使用用途を良く考え選択する必要がある。

これまで多くの OS で、32 ビット版、64 ビット版がリリースされていたが、2020 年からは、4GB メモリを下回るコンピュータはほとんどなくなり、新規にリリースされる OS は 64 ビットのみであることがほとんどとなっている。

1.5 実験内容(1)

ハードウェアのセットアップと、各コンピュータへのOSのインストールを行った後、ネットワークの設定を行う。

ハードウェアの設定の流れを、下記に示す。

- 端末となるコンピュータのハードウェアセッティング
 - キーボードの接続
 - モニタの接続
 - マウスの接続（サーバ用コンピュータには必須ではない）
- 各装置の電源の接続
- 各端末へのOSのインストール
- OSインストール後、各端末において必要となる初期設定
- ユーザの追加

各端末のハードウェアと使用OSの対応は表1.2である。

表1.2 使用OSと対象コンピュータ

OS	対象コンピュータ
Ubuntu Linux または別のUNIX系OS	サーバ用コンピュータ ASUS RS-100
Ubuntu Linux	クライアント用 UNIX系コンピュータ Dell Power Edge T110
Windows 10 (Education 64ビット版)	クライアント用 Microsoft Windows コンピュータ HP ノートPC
macOS	クライアント用 Macintosh コンピュータ ノートPC MacBook Air M1 2020
ChromeOS	クライアント用 ChromeBook

使用するハードウェアは以下に挙げる条件で設定を行う。

- デスクトップPCのキーボード接続はUSB接続を用いる。
- サーバには、HHK(Happy Hacking Keyboard)と呼ばれる、英語のテンキーのないキーボードを用いる。
- Dell製タワーPCには、Dell製英語キーボードを用いる。
- デスクトップPCのディスプレイ接続は、D-SUB 15ピン、アナログRGB接続を用いる（“D”の字を横にしたような、端子のピンが3列15個あるもの）。ただし、MacにはDisplay Port変換アダプタ、Windowsデスクトップには、DVI-I（ディジタル・アナログ混在端子）とRGBの変換アダプタが必要になる。

- 端子のピンは細く、折れ曲がりやすいので、無理に差し込むなどはしないこと。

日本語キーボードと英語キーボードは、記号類のキーの位置や、制御用のキーの位置が異なる。設定を間違うと、キートップに印字されている記号と、実際に入力される記号が異なるので、注意する。

1.6 必要となる知識

1.6.1 ハードウェア

現在、一般的に汎用 OS で用いられるコンピュータのハードウェアは、インテル製 CPU かその互換 CPU (例えば、AMD 製など) を搭載し、メモリが数 GB、ディスク装置が数 100GB、数 TB のものが一般的である。Microsoft Windows, Linux, FreeBSD など、様々な OS をインストールして、異なる用途に用いることができる。一部の製品では、Windows をインストールすることを念頭に開発され、その他の OS で用いることが難しい、あるいは、Windows 以外では一部のハードウェア機能を用いることできないなどの問題が生じる場合がある⁹。

1.6.2 キーボード・マウス接続

キーボード・マウス接続には、大きく分けて、下記の 2 通りがある。

- PS/2
- USB

PS/2 接続は、USB が開発される以前から存在した接続で、丸い端子で、キーボードは紫色、マウスは緑色で示されている。現在では、新製品にはあまり用いられていない。一部のサーバ向けハードウェアでは、互換性を重視する観点から、PS/2 端子を備えているものがあるが現在は、ほぼ USB に統一されている。

USB は、Universal Serial Bus の略称で、それまでキーボードやマウス、その他のデバイス接続において、デバイス毎に個別の接続端子が必要であったものを、全ての接続デバイスで統一的に (universal) 用いることができるシリアル接続端子 (Serial Bus) を構築することを目的に開発された。

90 年代後半の USB 1.1 以来、速度の向上がはかられており、現在では下記のような規格がある。

USB 1.1 1.5Mbps と 12Mbps の帯域幅が存在。マウスやキーボードなどの低速デバイス向け。電源供給は、5V, 500mA (2.5W) まであり、ポータブル HDD など電力が大きいものは、セルフパワーと呼ばれ、機器側でコンセントから電源供給する必要がある。

USB 2.0 480Mbps の帯域幅で、外部ハードディスクやカメラ、マイクをはじめ幅広く用いられる。

USB 3.2 Gen 1x1 (USB3.0) 5Gbps でハードディスクやハイビジョン品質映像キャプチャデバイスなどの高帯域幅が必要なものに用いられる。青色など、それまでの USB 2.0 と異なる色で示されることが多い。電源供給能力も向上し、4.5W (5V, 900mA) まで供給可能である。

⁹ 低価格のもの、新機能が多用されている新製品、ノート型 PC などは Windows 以外では機能制限される場合が多く、古い（枯れた）ハードウェアやサーバ用途製品では Linux などで容易に用いることができる場合が多い。

USB 3.2 Gen 2x1 (USB 3.1) 伝送速度を 10Gbps に高速化した規格。電源供給能力も強化され (USB-PD)，100W まで供給できる。USB-C と呼ばれる新しい端子も採用され，小型化され端子の向きも裏表どちらでも差し込めるようになり，PC 側・機器側の端子の形状も同じとなった。Apple の 2015 年の MacBook では，電源端子も含め外部接続ポートは USB-C 1 ポートに統一され，2016 年の MacBook Pro では，USB-C ポート (Apple Thunderbolt 3 端子を兼用) が 4 ポートとなった。

USB 3.2 Gen 2x2 端子が USB-C のみとなり，最大 20Gbps。

USB4 Apple Thunderbolt 4 と統合され，画面出力等幅広く使われる。まだ Apple 製品以外では採用されていない。

本実験では，全て，USB 接続のマウス・キーボードとなっている。

1.6.3 キーボードの配列

日本では，キーボード配列には，下記のように大きく 2 通りの配列がある。

日本語配列 キーボードのキートップに，日本語のカナが印字されているもの。キーボードの数から，106 配列，109 配列，JIS 配列などとも呼ばれる。

英語配列 キーボードのキートップに英語のみ印字されているもので，全世界的に英語圏を中心に使われているもの。101 配列，104 配列，US 配列などとも呼ばれる。

両者の配列では，アルファベットや数字の位置は変わらないが，記号の位置が異なるので，接続するキーボードと設定を誤らないようにする必要がある。誤った場合は，キーボードに表示されている記号と実際に入力される記号が異なって入力され，場合によっては一部の記号は入力できない。

1.6.4 ディスプレイインターフェース

ディスプレイの接続規格には，下記のようなものがある。

Dsub 15 ピン RGB アナログインターフェースであり，20 年以上用いられてきており，現在でも，ディスプレイやプロジェクタなどにしばしば用いられる。赤 (R)，緑 (G)，青 (B) の 3 原色の量をアナログ (電圧) で表現した信号であり，垂直周波数，水平周波数などが合っていないと正しく接続できない。RGB，VGA，Dsub，Dsub 15 ピン，アナログなどと呼ばれる。

DVI デジタルに対応したインターフェースである。アナログ RGB との互換性も取られており，アナログ信号とデジタル信号を混在して送ることができるが，一方がデジタルのみ，あるいはアナログのみにか対応していない場合は，接続しても正しく表示できない。デジタルのみの DVI-D，アナログのみの DVI-A，両者を同時に送ることができる DVI-I があり，形状が異なる。DVI-A はアナログ RGB と，DVI-D は HDMI と，変換ケーブルを使って接続できる。

HDMI 近年，普及したデジタル専用インターフェースで，コンピュータの他，テレビなど，デジタルデバイスに広く用いられている。音声も同時に送ることが可能な他，3D やディープカラーなど先進的な送信にも対応している。

Display Port Mac や小型ノート PC などで用いられる端子。

Thunderbolt Mac で用いられる端子。

1.6.5 ユーザ ID・グループ ID

UNIX 系 OS をはじめ、現代のオペレーティングシステムでは、マルチユーザ環境を実現しているが、ユーザの識別に用いられる数値をユーザ ID(UID) と呼んでいる。ユーザ ID は、16 ビット程度であることが多く、その場合は、32767（符号つき）や 65535（符号なし）の値が上限である。UNIX 系 OS では、UID 0 は常にスーパーユーザ（ユーザ名 root)¹⁰である。

また、1つ以上のユーザに対してまとめてファイルやプロセスの権限付与操作を行うためのグループの概念がある。このグループも、グループごとに一意に数値を持ち、これをグループ ID (GID) と呼ぶ。GID 0 は、BSD 系では wheel と呼ばれる、root ヘユーザ権限を変更できるユーザのグループとして定義され、管理者グループであるユーザを、wheel グループに登録しておく。

多くの OS のユーザアカウント作成では、UID や GID は重複のない番号が自動的に割り振られるので、通常はこれを用いれば良い。

これらの値をはじめとするユーザアカウント情報は、/etc/passwd, /etc/shadow¹¹, /etc/group に登録されている。passwd ファイルや shadow は、実際はファイルの内容が別のファイルにデータベース化されてからオペレーティングシステムで使用されるので、vi エディタなどで直接編集せず、vipw コマンドで編集を行う（ファイルそのものはテキストファイルなので閲覧は可能）。

```
# vipw
```

また、wheel にユーザを追加した場合、システムが正常に終了（shutdown）できることも確かめておくとよい。

1.6.6 デバイスファイル

UNIX 系サーバでは、コンピュータに接続される全てのデバイスは、デバイスファイルとして、ファイルシステムの一部に現れている。通常、/dev ディレクトリ以下に全てのデバイスファイルがある。

このうち、よく用いるディスクデバイスについて述べておく。ディスクデバイスは、ディスク（やその一部のパーティション）の先頭セクタ（ブロック）から順に一本の長いデータとして表されている。このディスクのブロックの集まりを、raw ファイル（生ファイル）などと呼ぶ。通常は、OS のファイルシステムを介してディスクの読み書きを行うので、raw ファイルを用いることはない。万が一、raw ファイルを誤って破壊すると、ファイルシステムが壊れ、ファイルが正常に読み書きできなくなる。

表 1.3 ディスクデバイスファイル

/dev/sda (, sdb, sdc,...)	HDD, USB メモリ等 (, 2 台目, 3 台目, ...)
/dev/sda1 (, sda2, ...)	1 番目のディスクのパーティション 1 (, 2, ...)

パーティション操作は、fdisk, gpart コマンドなどで行える。

¹⁰ UNIX における特権ユーザであり管理者権限を持つユーザである

¹¹ シャドウパスワードと呼ばれるパスワード保護の仕組みのある UNIX のみ存在する

1.6.7 パーティション

パーティションは、一つの物理ディスクを複数の領域に分けて、それぞれを論理的に一つのディスクとして扱うというものである。パーティションを分割している場合は、複数のディスク（領域）がOSから見えるようになる。OS本体のプログラムとデータを異なるパーティションにおくこともできる。また、異なるOSを別々のパーティションにインストールし、起動時（ブート時）にどのOSを起動するか選択することもできる。

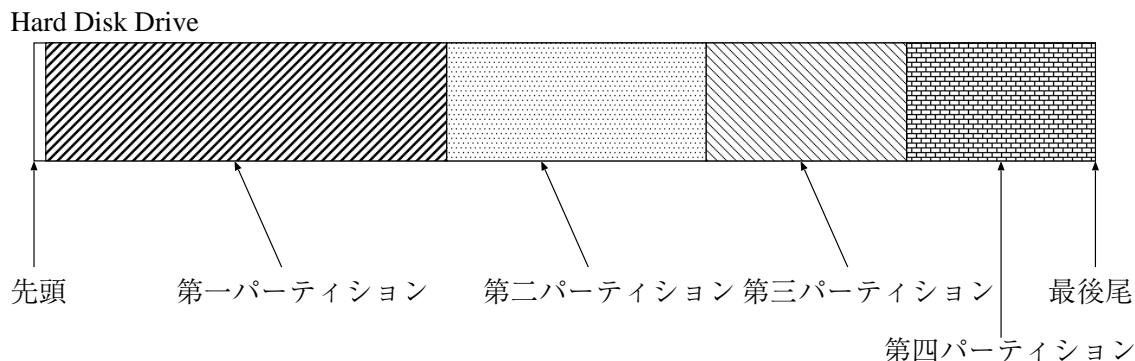


図1.1 パーティション

パーティション情報は、具体的には、ディスクの先頭部分に、各パーティションがどこからどこまでであるかを書き込むことで構成し、全てのOSはブート時およびブート後に、その領域情報に基づいて必要な領域のみにデータの読み書き操作を行う。

ディスクの先頭部分を、MBR (Master Boot Record) と呼ぶ。従来のパーソナルコンピュータでは、このMBRにブートローダ（起動プログラム）とパーティション情報をパーティションテーブルに書き込んでおり、このブートローダがパーティションテーブルを参照し、起動するパーティションからOSを読み込む。MBR形式パーティションの設定は、通常fdiskコマンドが使われる（Windows, Mac以外のUNIX系）。

MBRは2TB以上の容量には対応できないため、最近は、GPT (GUID Partition Table) 形式でパーティション情報を保存するものが増えてきている（MacOS X, Windows 10, Linux, FreeBSD 9.0）。GPTにはブートローダは入っておらず、パーティションテーブルのみがある。ブートローダは、コンピュータのマザーボード上のチップセットに格納されているEFI（またはUEFI: Unified Extensible Firmware Interface）に入っている。UNIX系OSでは、gpartあるいはgdiskコマンドで操作を行う。Windows 7など最新でないOSではGPTでは不都合がある場合もあるが、最新のOSはほとんどGPT・EFIに対応し、今後はこちらが主流になる。

1.6.8 サーバOSの操作に必要なコマンド

サーバOSを操作するコマンドを表1.4に挙げる。サーバOSは、GUI(Graphical User Interface)を用いずに、CUI(Character User Interface)によるCLI(Command Line Interface)で、コマンドによる操作のみで運用を行うことが多い。このため、CUI操作で必要なUNIXコマンドをよく調べておくと、今後の操作が円滑に行える。

また、エディタとして、viやemacsに慣れておくと良い。特に、viはUNIX系OSでは標準的に備わっているので、最低限の操作には慣れておくこと。

表 1.4 UNIX 系 OS 基本コマンド

シャットダウン	shutdown -h now
再起動	reboot
ログアウト	exit
ユーザアカウント作成	useradd
アカウント情報変更	chsh
アカウント情報変更 (root のみ)	vipw
ディスクマウント状況確認	mount
ディスク容量, パーティション確認	df -k
ディレクトリ以下のファイル総容量	du -k
ファイル操作	cp, mv, ls, cd
ページャ	more, less
メッセージ記録	script ファイル名
ファイル検索	find ディレクトリ -name 'ファイル名' -print
ファイル内容検索	grep キーワード ファイル1 ファイル2 ...
パーティション操作	gpart サブコマンド HDD のデバイスファイル
ディスク初期化 (format)	newfs パーティションのデバイスファイル

コマンドには、様々なオプションが付いており、これらをよく理解し使いこなすことで、操作性を上げることができる。どのようなコマンドがあるかは、man コマンドでオンラインマニュアルを調べることができる。例えば、man コマンドのマニュアルを読むには、下記のようにすれば良い。

```
# man man
```

1.6.9 ユーザアカウントの作成とパスワードの設定

Linux におけるユーザアカウントの追加を useradd コマンドで行う。

man コマンドで詳しい使い方を閲覧できる。

```
# man useradd

SYNOPSIS
    useradd [options] LOGIN

    useradd -D

    useradd -D [options]
```

```
# useradd -c 'フルネームなど' -m ユーザ名
```

-c はローマ字で本名などを書く。空白を含む場合は、シングルクォーテーションで囲む。-m はホームディレクトリも自動的に作成する。ホームディレクトリは、/home ディレクトリ以下に作成される。

ユーザ名は、8 文字以下の英数字で、英語大文字は使わない（小文字のみ）かつ 1 文字目は数字を使わない。

パスワードの設定は passwd コマンドを用いる。

```
# passwd ユーザ名
```

ユーザの作成を終えたら、ログイン情報が記録されたファイルを調べる。

/etc/passwd ファイルがそうであり、1行に1アカウント分の情報が、コロン：で区切られて記載されている。このファイルは全てのユーザが閲覧できる必要がある。以前は、ここにパスワードもハッシュ化されて記載されていたが、実際のパスワード情報は、別のシャドウファイル(shadow)に置かれており、root ユーザしか閲覧できないようになっている。編集、およびパスワード(ハッシュ化されているので、パスワードそのものはたとえスーパーユーザでも分からぬ)の閲覧は、vipw を用いることで行う。グループの所属情報は、/etc/group に記録されている。

ls コマンドでユーザディレクトリが作成されているかも確認する。

```
# ls -l /home/
```

1.7 必要な情報

1.7.1 サーバOS

ホスト名：serverX (Xの部分は、グループ番号で置き換えること)。

その他、下記の要領で設定を行う。

- ディスクパーティションの設定については任意の設定を行って良いが、ルートパーティションに十分な容量(数10GB)を割り当てること。
- ユーザ“exp”をインストール時に作成する。
- exp ユーザのパスワードは、“root00”とする(この項目は、実験実施上必要な措置であり、実運用においては十分長く、推測されにくく、辞書などにないパスワードを設定すること)。
- グループ全員分のユーザアカウントを作成する。

1.7.2 クライアントOS

クライアントOSについては、下記の通り設定を行う。

- 管理者ユーザアカウントを1つ作成する(可能な限り管理者ユーザ名は、“exp”とする)。管理者ユーザのパスワードは、“root00”にする。
- ユーザ登録画面などが出た場合は、氏名は「KUT GroupX」(Xはグループ名)、住所等は、大学のものを登録する。
- Chromebookでは、Googleアカウントが必要なため、kutexpG@gmail.com (Gはグループ番号)というアカウントを作成し、これを使う。またパスワードは強固なものとし、TAに伝える。
- ネットワーク設定は後に行うので、特に設定しなくて良い。

- ディスクのパーティション設定は、オペレーティングシステムのための領域をあまり小さくしそぎないこと。これは、オペレーティングシステムの運用が始まると、スワップファイルや一時ファイルを作成するので、これらのための領域が足りないと動作に支障を及ぼすためである。

クライアント OS のホスト名は、表 1.5 の通りとする。ただし、表中 X はグループ番号を表す。図 1.2 も参照のこと。

表 1.5 クライアント OS のホスト名

コンピュータ・OS	ホスト名
Desktop Linux	linux
Windows 10 ノート	win
Apple Macbook	mac
Chromebook	chrome

1.8 OS のインストール手順

1.8.1 Ubuntu

- インストールメディアの挿入

Ubuntu はオープンソース OS であり、インターネットに公開されている。インターネット上から、最新バージョン（または所望するバージョン）のイメージをダウンロードする。イメージとは、記録メディア（CD, DVD, HDD, SSD など）に記録されている情報を、先頭ブロックから最後まで順番に記録したファイルであり、イメージライタと呼ばれるソフトウェアで、メディアに書き込むことで、インストール等が行えるメディアが作成できる。CD や DVD 等のメディアは、ファイルシステムの形式が ISO 規格¹²で定められたものであり (ISO 9660 など)、CD や DVD 向けのイメージファイルは、iso 形式などと呼ばれる。

Ubuntu 20.04 LTS server 64 ビット版のインストールメディアを用いてインストールする。

- 起動

DVD を挿入したら、リセットボタンまたは、電源を押し直すなどして再起動する。DVD ドライブから起動しない場合は、[F8] を押すとブート選択メニューが出るので、光学ドライブを選択する。

- インストール言語、キーボード配列

Language (インストール言語) は、「English」とする。

- Install

「Install Ubuntu Server」を選択する。

¹² 国際標準化機構、本部スイス・ジュネーブ

(インストール時の言語を再度聞かれるが、English にする)

- 国・地域設定

時計などの国の設定を行う。

「other」→「Asia」→「Japan」を選択する。

「locale」(ロケール) 設定は、ここでは、「en_US.UTF-8」を選択。

- キーボード

キーボードは英語版を選択する（通常は、US で良い。UK にはしない）。

ここで間違えると、キーボードのキートップの文字と入力される文字が異なることになり、使いにくくなる。

- ネットワーク設定

ホスト名は、「serverX」(X はグループ番号) とする。

(入力したら、TAB を押して「Continue」までいって Enter を押す)。

- 一般ユーザ設定

ユーザ「exp」を作成する（Full name もアカウントのユーザ名も exp で良い）。

パスワードは、「root00」とする。

弱いパスワードを用いるか確認されるが、これを用いることを選択する。

ホームディレクトリの暗号化も聞かれるが、ここでは行わないこととする。

- 時計設定

タイムゾーンを確認する (Asia/Tokyo)。

この設定を正しく行わないと時刻がずれ、メールやファイル共有で困る場合がある。

- partition

パーティションの設定を行う。Guided はガイダンスつき、Manual が完全に操作者が自由に設定可能な手動モードである。

Entire disk を選択して、ディスク全体を使うよう設定する。

なお、LVM (Logical Volume Manager : 論理ボリュームマネージャ) は、ディスク利用の自由度を大きく向上させるが、設定が難しくなる面もあり、ここでは用いないこととする。

また暗号化は行わない。

sda = 1 台目のディスクを選択する。

ディスクに変更を書き込むか聞かれるので、書き込むことを確認する。

パーティション設定は、既存のディスク情報（データや OS、その他の配置情報）をすべて上書きしてしまうので、既存のデータには基本的にアクセスできなくなる（消去されることと同じ）。パーティション等のディスク操作は、注意して操作する習慣をつける。

このため、ここでは、操作を決定する前に本当に操作を行って良いか問われる。ディスク sda のパーティション情報 (partition 1, 5 等)、それぞれのフォーマット情報 (ext4 や swap など) が表示されるので、問題なければ、実行する。

- パッケージマネージャのプロキシ

パッケージマネージャ（ソフトウェアのインストールツール）のプロキシを尋ねられるがここでは、空白としておく（ネットワークの設定後に行う）。

- アップデートの設定

アップデートは、セキュリティ上重要だが、サーバではアップデートの影響を慎重に考える必要があり、ここでは自動アップデートは行わないようにする。

- 初期ソフトウェアの設定

最初からチェックの入っているものはそのままとし、それ以外からは、OpenSSH server を、リモートログインでの作業とファイルコピーを行うためにインストールする。

- ブートローダ GRUB

ブートローダをマスタブートレコード (MBR) に導入しないと OS が起動しないため、MBR に GRUB のインストールを行う。

以上で終了である。Continue を選択し再起動する。

- 再起動

再起動は、DVD-RW メディアを抜いて行う。

- ログイン

login プロンプトが現れるので、exp ユーザでログインする。

その後、sudo su コマンドで管理者ユーザ root となる。

```
$ sudo su  
#
```

管理者となると、プロンプトが \$ から # に変わる。

- ユーザの追加

ログインが無事行えたら、グループ内全員分のアカウントを作成する。

1.9 考慮すべき点

OS の必要性 現在のコンピュータシステム、ネットワークシステムを構築していくのに、なぜ OS が必要かを考える。適切な OS を各コンピュータに導入する必要性は何かを考える。また、現在の OS にはどのような機能が必要とされるか、あるいは備えているか。将来は、どのようなものが求められるかを考える。

ユーザ権限 ユーザの権限や管理者権限の必要性について考える。

1.10 実験内容 (2)

様々なサーバソフトウェアを動作させるためには、オペレーティングシステムに対して適切なネットワーク設定を行う必要がある。サーバは通常アプリケーション層（あるいはセッション層以上）で動作するため、必要なネットワーク設定は、物理層からネットワーク層（実質はトランスポート層）までの設定である。

そこで、インストールされたOSに対して、イーサネットとIPネットワークの設定を行う。

- (1) もしコンピュータにネットワークインターフェースが認識されていない場合は、ネットワークインターフェースのハードウェアの種類と、使用するOSのバージョンに合致したデバイスドライバをインストールする。
- (2) ネットワークケーブルを用意し、レイヤー2スイッチに適切に接続する。UTPカテゴリ5の場合は、適切な長さのケーブルを作成することができる。
- (3) ネットワークインターフェースにIPアドレス・サブネットマスク・デフォルトゲートウェイ、DNSサーバを設定する。
- (4) インターネットへ接続ができるようにする
- (5) 下記ネットワーク管理に有用なソフトウェアをインストールする
 - Linux: net-tools, telnet, traceroute
 - Windows: Putty, WinSCP
 - ChromeOS: Linux (ping, ssh等) を使えるようにする。Androidアプリが使えるようにする。「Wifi Analyzer」をインストール
- (6) 全ての端末から、全てのユーザ名で、sshでログインできるようにする。

まず、図1.2を参照し、各グループに必要なLANケーブルを準備する。作成したLANケーブルで各機器を正しく接続した後、IPアドレス、サブネットマスク、デフォルトゲートウェイ、DNSサーバなどの設定を行い、各機器が通信を行える環境を構築する。また、外部Webサイトを閲覧できるように、各クライアントPCのブラウザにHTTP Proxyサーバの設定を行う。

1.11 必要となる情報

1.11.1 各機器のネットワーク情報

各グループのネットワーク構成は図1.2で示す。

機器の設定一覧を表1.6に示す。なお、Xは各グループ番号である。また、サブネットマスク、デフォルトゲートウェイ、DNSサーバは表1.7のように設定する。DNSサーバについては、後の章で各グループに構築されるが、ここではメインサーバを用いる。

さらにメインサーバ192.168.0.1のポート7999にはHTTP Proxyサーバが稼動している。

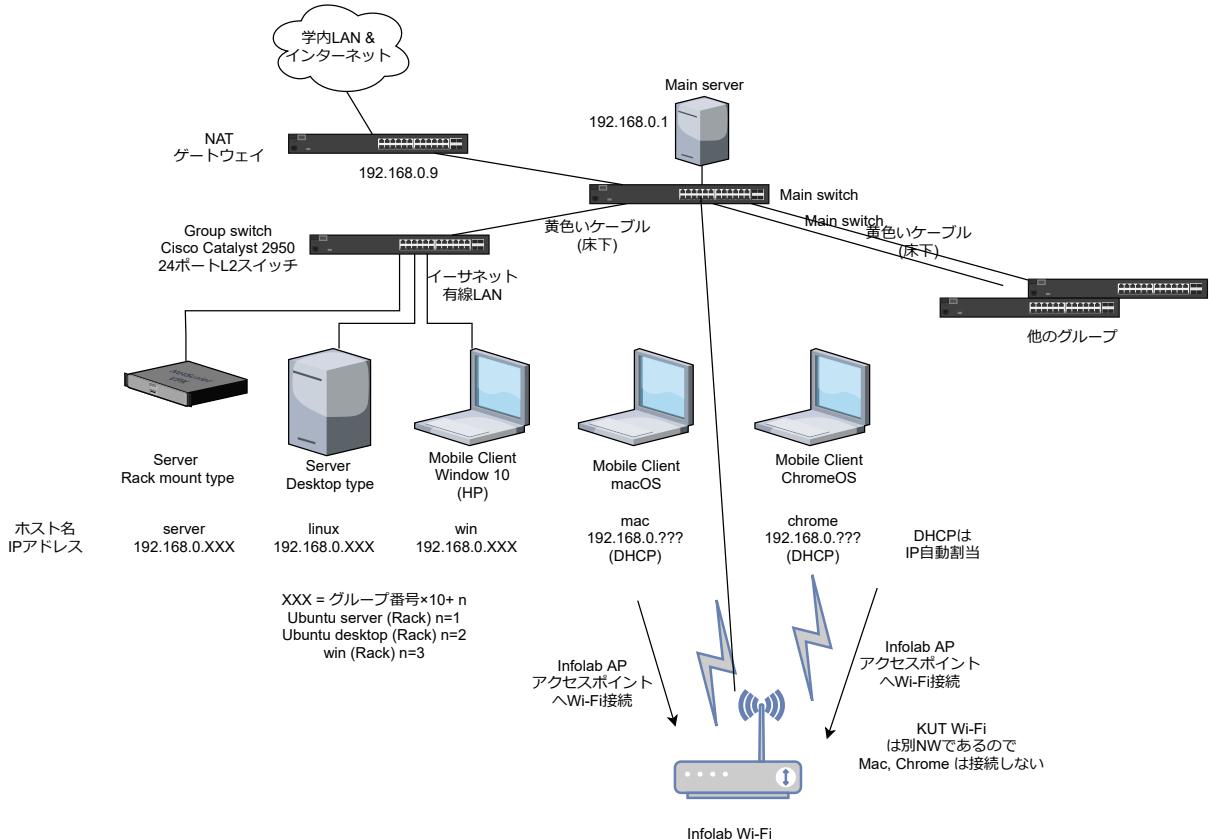


図 1.2 各グループのネットワーク接続図と A260 全体ネットワーク

なお、同じ LAN 内のサーバにアクセスする際は、プロキシの設定を無効にしなければならない場合があるので、注意する。

表 1.6 各コンピュータと IP アドレスの対応

コンピュータ	IP アドレス
サーバ	192.168.0.X1 (有線 LAN イーサネット, I/F: enp3s0)
Ubuntu Desktop Linux	192.168.0.X2 (有線 LAN イーサネット)
Windows 10 HP ノート	192.168.0.X3 (有線 LAN イーサネット)
Macbook macOS ノート	Wi-Fi DHCP 自動割当
Chromebook ChromeOS ノート	Wi-Fi DHCP 自動割当

表 1.7 ネットワーク設定情報

サブネットマスク	255.255.255.0
デフォルトゲートウェイ	192.168.0.9
DNS サーバ	172.30.0.2

1.12 LAN の構成機器

通常、単体のコンピュータはスタンドアロン (Stand Alone) の形で利用することもできるが、現代では、インターネット通信や携帯端末をはじめとするコンピュータ通信技術が一般社会にも広く普及しており、計算機間を接続するネットワーク技術は、現代のコンピュータにとって標準的な機能になっている。

計算機ネットワークを構成するには、まず、LAN 技術を用いて相互に接続を行う。LAN は、Local Area Network の略で、「狭い」範囲のコンピュータネットワークを意味する。「狭い」の定義は場合によって異なる。本来、一つの組織の（大学や会社、学群や部署、研究室など）中ののみの接続を意味しているが、ルータを用いずに通信する範囲を意味することも多い。典型的には、研究室や自宅の中の接続などがこれに該当するが、より広い範囲に適用されることも多い。

現在、LAN 技術としては、IEEE¹³ 802.3 の規格で定められているイーサネット (Ethernet) が事実上の世界標準 (defacto standard) である。イーサネットは、OSI 7 階層モデルにおける、Layer 1 物理層、Layer 2 データリンク層の通信を担う。Layer 1 では、ケーブルやコネクタ形状、電気信号での電圧・電流・インピーダンス¹⁴や、光信号での波長、ビットの表現（ライン符号化）が規定されている。イーサネットでは、これらの違いにより様々な規格があるので、知識として知っておくことは重要である。Layer 2 では、イーサネットで用いられる伝送単位であるフレーム¹⁵や、フレームの送信元・宛先アドレス（MAC¹⁶アドレスと呼ぶ）などが決められている。

これらのうち、運用現場で特に重要な情報は、イーサネット規格のうちのどの規格を用いるかと、使用するケーブルやコネクタに関する情報である。

イーサネット LAN を構成する機器は、下記の通りである。

- コンピュータ
- ネットワーク (TCP/IP) 対応のオペレーティングシステム
- ネットワークインターフェース (カード) (I/F または NIC)
- ケーブル
- ネットワーク (イーサネット) デバイスドライバ
- スイッチ（基本機能のみのものは、スイッチングハブあるいは単にハブとも呼ばれる）
- (ルータ (Router) : ここではまだ用いない)

¹³ The Institute of Electrical and Electronics Engineers

¹⁴ 交流での抵抗=電流の通りやすさのようなもの

¹⁵ パケットと呼んでも間違ひではないが、IP など他のレイヤーのパケットと混同しやすいので注意する。

¹⁶ Media Access Control

以下では、その中からケーブルとネットワークインターフェース、およびスイッチングハブについて説明を行う。

ケーブル

表 1.8 LAN の種類とメタルケーブル

LAN の種類	10BASE-T	100BASE-TX	1000BASE-T
使用ケーブル	UTP カテゴリ 3 以上	UTP カテゴリ 5 以上	UTP カテゴリ 5 エンハンスト以上
伝送速度	10Mbps	100Mbps	1Gbps
セグメント長	100m	100m	100m
リピータ最大段	4	1	原則用いない

リピータは、L1 の接続機器であり、ケーブル長に制限がある場合にリピータを用いて延長することができる。例えば、10BASE-T であれば、ケーブル長が最大 100m で、リピータを 4 段¹⁷まで接続できるので、500m 離れた機器同士を接続することができる。

リピータの役割は、長いケーブルで減衰やノイズの重畳がされた信号を、Layer 1 レベル、すなわちビットレベルで、本来の信号の電圧値に增幅し、信号波形やタイミングの整形を行う。ビットレベルの整形のみを行うので、Layer 2 のイーサネットフレームとしての認識は行わないため、コリジョン (Collision: 衝突) 信号などのイーサネットとしては意味をなさない信号も传送する。

ただし、リピータは現在の高速化したイーサネットではハブとして用いられることはない。現在は、10BASE 等の低速イーサネット用でもリピータハブは販売されておらず、過去の機器であると考えて良いが、その仕組みは理解しておく必要がある。また、L1 機器としては、他にトランシーバ（光信号と電気信号に変換して機器に入力するデバイス。GBIC や SFP など）やリピータ型メディアコンバータ（光と電気の変換）などがある。

非常に古いタイプのハブ¹⁸はリピータハブの可能性も全く無いわけではないが、現在は、ほぼ完全にハブとしては、Layer 2 のスイッチングハブ (Ethernet スイッチ、ブリッジ) が用いられる。

表 1.9 LAN の種類とケーブル（光）

LAN の種類	1000BASE-SX	1000BASE-LX
使用ケーブル	光ファイバ 主にマルチモード (GI)	光ファイバ シングルモード
伝送速度	1Gbps	1Gbps
セグメント長	550m	5-10Km

¹⁷ ネットワーク全体としては 5 個以上リピータ存在しても良いが、全ての機器同士の通信において、経路上に 5 回以上リピータを通過してはいけない。この通過回数を「段」と呼んでいる。

¹⁸ 2000 年以前の 10BASE 専用ハブ等

表 1.10 LAN の種類とケーブルの一覧 (10G)

LAN の種類	10GBASE-SR	10GBASE-LR	10GBASE-ER
使用ケーブル	光ファイバ 主にマルチモード (GI)	光ファイバ シングルモード	光ファイバ シングルモード
伝送速度	10Gbps	10Gbps	10Gbps
セグメント長	26m～300m	10Km	40Km

表 1.11 UTP の分類

分類	適用	対応速度の目安
カテゴリ 1, 2	音声や低速のデータ伝送に用いられる。 LAN 配線システムの規格外	—
カテゴリ 3	一般に音声や 10Mbps までのデータ伝送に用いられる。 IEEE802.3 の 10BASE-T や IEEE802.5 のトーカンリング (4Mbps)	1～10Mbps
カテゴリ 4	カテゴリ 3 の用途と 16Mbps (IEEE802.5 のトーカンリングまで) のデータ伝送	4～16Mbps
カテゴリ 5	カテゴリ 4 までの用途, 音声や 100Mbps (100BASE-TX), 更には 156Mbps の ATM-LAN にも用いられる。	10～100Mbps
カテゴリ 5e	カテゴリ 5 までの用途, Gigabit Ethernet (1000BASE-T) 現在売られている LAN ケーブルは殆どがこのカテゴリ 5e 以上である。	1Gbps
カテゴリ 6	カテゴリ 5e までの用途, Gigabit Ethernet (1000BASE-TX), 10GBASE-T, ATM (622Mbps) ATM (1.2Gbps)	1～10Gbps
カテゴリ 6a	カテゴリ 6 までの用途, Gigabit Ethernet (1000BASE-TX), 10GBASE-T	1～10Gbps
カテゴリ 7	カテゴリ 6a までの用途, Gigabit Ethernet (1000BASE-TX), 10GBASE-T	1～10Gbps

ネットワークインターフェース

LAN ネットワークを構築する際に、それぞれのコンピュータをケーブルで接続する必要がある。コンピュータとケーブルを接続するインターフェースのことを、ネットワークインターフェース (I/F または Network Interface Card, NIC)，あるいは **LAN 端子** 等と呼ぶ。

レイヤー2スイッチまたはハブ（Hub）

LANにおけるスイッチ（レイヤー2スイッチ）またはハブは、各PC（のNIC）に接続されたケーブルをお互いに接続するための集線装置のことを意味する。現在のイーサネットでは、ハブのポートとPCのI/Fとを一对一で接続する。Cisco社のCatalystスイッチなどが代表的な製品の1つである。

ハブには、OSIネットワーク階層モデルでのレイヤ1（物理層）で接続するリピータハブと、レイヤ2（データリンク層）で接続するスイッチングハブ（イーサネットスイッチ、単にスイッチ、あるいはブリッジなどとも呼ばれる）とがあるが、現在では、100BASE-TX以上の規格ではスイッチングハブのみが使われている。このため、単にハブと言う時、事実上(L2)スイッチを指すことがほとんどである。

なお、ハブ同士を接続（カスケード接続とも言う）し、さらに大きなLANを構築することも可能である。ただし、ハブ同士の接続の注意点として、下記のような点に注意する必要がある。

- リピータの場合は、最大4段（4個ではない）以上接続できない。スイッチ（ブリッジ）では制限はない。
- ハブ同士の接続は、10BASE-Tまたは100BASE-Tではクロスケーブルが必要か、片方をハブの特別なポート（カスケードポート、MDI-Xポート）に接続する必要がある。近年のスイッチ、PCのインターフェースは、AUTO-MDI/MDI-Xと呼び、ストレートケーブルのみで、自動的に対抗がどのような機器でも接続できるように内部配線が変更される。そのためクロスケーブルはほとんど必要ない（古い10/100BASE機器でのみ必要となる）。
- 1000BASE-Tでは、リピータは存在せず、1000BASE-T規格ではAUTO MDI/MDI-XによるMDI・MDI-X自動認識が必須であるためストレートケーブルのみで接続する。

なお、ネットワーク管理機能、接続状況やパケット流量（フロー）の計測、VLAN等の高度な機能を備えたスイッチングハブは、インテリジェントスイッチとも呼ばれる。

1.13 必要となる知識

カテゴリ5エンハンストケーブルの作成方法

(1) ケーブルの準備

ケーブル両端の外被（一番外側の皮膜）を皮むき器（ケーブルストリッパ）、あるいはニッパなどを用いて先端から1cmほどむき、中の芯線を出す。この時、芯線の外被まで切ったり傷つけたりしてケーブルに障害を与えないよう注意する。各芯線そのものを覆っている外被を剥く必要はない。外被を剥くとツイストペアケーブルは芯線が2本づつよじれていることが確認できる。

次にモジュラジャックにケーブルを差し込むため、各芯線のヨリをときほぐし、芯線をまっすぐ伸ばし、8本が平面になるように揃える。芯線の長さも揃えておいておく（はさみやニッパで切りそろえる）。

(2) ケーブルとモジュラジャックの圧着

ケーブルを作る前にケーブルの芯線の色と線の番号の対応を知る必要がある。ケーブルの配色と番号の対応を表 1.12 に示す。

この番号は、モジュラジャックのツメ（フック）がある側を上にして、奥をケーブル側、手前を端子側とし左から順番に 1, 2, …, 8 となる。色との対応は、EIA/TIA 568B にて規格化されているので、ストレートケーブルだからといって、勝手に変更しないようにする¹⁹。

表 1.12 ケーブルの芯線の色と番号の対応

1	2	3	4	5	6	7	8
白／橙	橙	白／緑	青	白／青	緑	白／茶	茶

UTP には、導線の配置がケーブルの両端で一致しているストレートケーブル（単にストレートともいう）と、導線の一部が交差するように配置するクロスケーブル（単にクロスともいう）の二種類の配線がある。ストレートケーブルは、端末とハブとの接続やルータとハブとの接続に使われ、クロスケーブルは、端末同士あるいはハブ同士の接続に使われる。なお、表 1.12 の 1-3, 2-6 をそれぞれ結線させれば、クロスケーブルとなる。

ケーブルの各芯線が、対応したモジュラジャック先端の電極に接触するように、ケーブルをモジュラジャックに差し込む。この時、モジュラジャックを横から見て、電極の一番奥（つまり図 1.3 においてはモジュラジャックの最上端）まで芯線が届くようにする。もし届かなければ接触不良で信号が通らないので、ニッパーなどを用いて芯線の長さを整えるなどの処理を施す。

次に、圧着器にモジュラジャックを差し込み、ケーブルと圧着する。一度圧着したモジュラジャックは二度と使えなくなるので、圧着する前に、再度芯線の順序を確認すること、反対側の端子についても同様である。

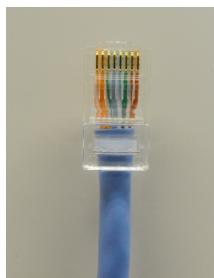


図 1.4 圧着器にジャックを差し込んだ場合

図 1.3 ジャックを下から見た場合

¹⁹ ノイズやインピーダンス、クロストークの問題が発生するため、ケーブルチェックなどで警告が出される他、通信障害の原因になる

1.13.1 サーバのネットワーク設定

サーバOSではあらかじめネットワークの設定を行う必要がある。各項目に対する与える値の意味は今後実験を通じ解説を行うが、とりあえず以下の通り設定を行う。

- ifconfig を用いた IP アドレスの設定

この方法は、一時的に設定するときに使うもので、一般的なサーバネットワーク設定では使わない。後述の netplan 設定を使う。

しかし、Macを含めたUNIX系OSにおいては、ネットワークインターフェースの設定（IPアドレス・ネットマスク・データリンク層）は、伝統的にifconfigコマンドが使われてきており、知っておくと様々なサーバ設定で役に立つ場面がある。

なお、Linuxでは、2018年頃からの新しいLinuxでは、ipコマンドを用いるよう変更されている(ifconfigも存在し、aptコマンドをタイプするとパッケージシステム追加インストールするよう指示される)。

GUIなどの設定ツールも、内部でip, ifconfigコマンドを内部で呼び出している。

また、OS起動時のIPアドレスの自動設定も、起動スクリプトがファイルに保存されているIPアドレス情報を読みだし、ip, ifconfigコマンドを実行してIPアドレスの付与を行っている。

ipコマンドでは、addressオプションを付けて実行することで、現在のインターフェース設定を確認することができる。

ifconfigコマンドでは-aオプションを付けて実行する。-1オプションなしであれば、設定の一部情報のみの確認ができる。

ipコマンド実行例：(新しいlinux)

(現在のIPアドレスの下記九人)

```
exp@server:~$ ip address   (<-- ip a と省略できる)
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group de...
    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: enp3s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN...
    link/ether e0:cb:4e:87:4f:12 brd ff:ff:ff:ff:ff:ff
3: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP gro...
    link/ether e0:cb:4e:87:4e:b7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.150/24 brd 192.168.0.255 scope global dynamic enp2s0
        valid_lft 65985sec preferred_lft 65985sec
    inet6 fe80::e2cb:4eff:fe87:4eb7/64 scope link
        valid_lft forever preferred_lft forever
```

OSからは2つのインターフェースが見えていて、1つ目がループバック（自分自身）、その他、

イーサネットが、 enp2s0 と enp3s0 の 2つがある。

link/ether の項目には、 データリンク層アドレス (MAC アドレス) が確認できる。

その他の項目は、

NO-CARRIER → ケーブルが接続されていない

UP → 有効 (管理設定の状態)

state → 状態 (実際の状態), DOWN でダウン, UP でアップ

それぞれの PC では、 メーカなどに応じて別の名前が付くため (enp?? や eth? 等も), 確認する。

I/F 名がどちらがどちらかは、 実際に差して、 有効になった方の I/F 名を確認する。

ifconfig コマンド実行例 : (mac, linux)

```
$ ifconfig -a
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:6d:11:70
          inet addr:172.21.39.110  Bcast:172.21.39.255  Mask:255.255.255.0
          inet6 addr: fe80::7508:7196:26f1:3aae/64 Scope:Link
                    UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
                    RX packets:65936905 errors:0 dropped:14 overruns:0 frame:0
                    TX packets:26458399 errors:0 dropped:0 overruns:0 carrier:0
                    collisions:0 txqueuelen:1000
                    RX bytes:79994773867 (79.9 GB)  TX bytes:53642932020 (53.6 GB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
                    UP LOOPBACK RUNNING  MTU:65536  Metric:1
                    RX packets:1928 errors:0 dropped:0 overruns:0 frame:0
                    TX packets:1928 errors:0 dropped:0 overruns:0 carrier:0
                    collisions:0 txqueuelen:1
                    RX bytes:224307 (224.3 KB)  TX bytes:224307 (224.3 KB)
```

ipconfig コマンド実行例 : (Windows)

```
PS C:\Users\user> ipconfig
```

Windows IP 構成

イーサネット アダプター イーサネット:

```
接続固有の DNS サフィックス . . . . .:  
IPv6 アドレス . . . . . : 2404:1b00:1:39:cd2b:34d4:a0d2:1290  
一時 IPv6 アドレス . . . . . : 2404:1b00:1:39:352b:a09e:f9f2:3d71  
リンクローカル IPv6 アドレス . . . . . : fe80::cd2b:34d4:a0d2:1290%17  
IPv4 アドレス . . . . . : 172.21.39.22  
サブネット マスク . . . . . : 255.255.255.0  
IPv4 アドレス . . . . . : 192.168.1.2  
サブネット マスク . . . . . : 255.255.255.0  
デフォルト ゲートウェイ . . . . . : fe80::212:e2ff:fe6e:79db%17  
172.21.39.7
```

ip の実行例では、インターフェース enp71s0 の IP アドレスが、222.229.69.82 であることがわかる。ifconfig の実行例は、ネットワークインターフェース enp0s3 の IP アドレスは、172.21.39.110 であることが確認できる。

lo というインターフェースは、ループバックと呼ばれコンピュータの内部間の通信に使われる。IP アドレスは、常に 127.0.0.1 であり、自分自身としか通信できない。

その他の主な項目についての説明を表 1.13 に示す。

表 1.13 ifconfig コマンドの主な表示内容

Flag	UP: インターフェース有効, BROADCAST: ブロードキャストを行うリンク, RUNNING: 動作中, PROMISC: プリモジックモード
MTU	Maximum Transfer Unit 最大のペイロード長 (1 フレームの データ量) (オクテット)
HWaddr	NIC の MAC アドレス (L2 アドレス)
inet addr, Mask, Bcast	IP アドレス, ネットマスク, ブロードキャストアドレス (設定されていれば)

下記のコマンドは、メディア(光ファイバーやツイストペア)、リンク速度や Duplex(全二重か半二重)などを調べることができる。

```

Supported pause frame use: No
Supports auto-negotiation: Yes
Supported FEC modes: Not reported
Advertised link modes: 10baseT/Half 10baseT/Full
                        100baseT/Half 100baseT/Full
                        1000baseT/Half 1000baseT/Full
Advertised pause frame use: Symmetric
Advertised auto-negotiation: Yes
Advertised FEC modes: Not reported
Link partner advertised link modes: 10baseT/Half 10baseT/Full
                                    100baseT/Half 100baseT/Full
Link partner advertised pause frame use: No
Link partner advertised auto-negotiation: Yes
Link partner advertised FEC modes: Not reported
Speed: 100Mb/s
Duplex: Full
Port: Twisted Pair
PHYAD: 1
Transceiver: internal
Auto-negotiation: on
MDI-X: off

```

Speed は 100Mbps で、ツイストペアケーブル、Full Duplex (全二重) で接続されている例。

次に、ifconfig で IP アドレス設定方法について説明する。

この方法は、OS に対して確実に IP アドレスを付与することができるが、再起動した際には、再度同様のコマンドを発行しなければならないため、通常は、一時的に IP アドレスを付与・変更するために用いる。

インターフェースへの IP アドレス設定は、スーパーユーザしか行うことができないので、root になる。

下記のように、付与したいインターフェース名を指定して、inet に続き IP アドレスを、netmask に続きネットマスクを入力する。

そしてインターフェースを有効にするために、up と記述する。

```
# ifconfig IF 名 inet IP アドレス netmask ネットマスク up
(↑臨時に IP を変更するとき以外はあまり用いない)
```

この後、正しい UP アドレス・netmask が設定されていることを確認する。

また、ケーブルの接続を行い、インターフェースが、UP 状態で active となっているか確認する。

- デフォルトゲートウェイの設定

デフォルトゲートウェイの確認を行うには、`netstat` コマンドを用いる。-rn オプションをつけて用いる。以下は使用例である。

```
# netstat -rn
Routing tables

または,

# ip route (← ip r と省略可)
```

ここで表示されるものは、ルーティングテーブル（Routing Table 経路表）であり宛先 IP ネットワーク毎に、次に転送するルータ（Next Hop Router）が示されている。

ルーティングテーブルでは、ネットマスクがプレフィックス表記（/24 等）になっているので、適宜読み変えること。

宛先ネットワーク（destination）が default と書かれた行がデフォルトゲートウェイであり、この例では 192.168.0.1 となっている。

宛先ネットワークが、0.0.0.0/0 になっている場合もあるが、デフォルトネットワークと同様の意味である。

設定・変更する場合は、`route` コマンドを発行する。

```
#route delete default (既に default が設定れている場合は、削除する)
#route add default gw X.Y.Z.W ← X.Y.Z.W はデフォルトゲートウェイの IP アドレス
(↑臨時に IP を変更するとき以外はあまり用いない)
```

設定後は、ルーティングテーブルを確認する。

- ホスト名の設定・確認

UNIX ではホスト名の確認・設定は、`hostname` コマンドで行う。

```
# hostname serverX (設定)
```

```
# hostname (確認)
serverX
```

更に、OS に標準の名前解決システムである Hosts ファイルに、自分の名前を登録する。

```
# vi /etc/hosts
下記の 1 行を追記
192.168.0.X2      serverX
```

以上で L3 の設定は完了であるが、この状態では再起動後に設定がなくなってしまうので、次節の netplan にて設定をしておく必要がある。

1.13.2 Netplan によるサーバのネットワーク設定

管理者権限

```
$ sudo su
```

netplan ディレクトリへ移動

```
# cd /etc/netplan
```

```
# ls
```

YAML 形式の設定ファイルがあるため、ファイル名を確認

00-installer-config.yaml あるいは 01-network-manager-all.yaml 等

vi エディタで編集

```
# vi 上記の YAML ファイル
```

まず、中身を確認

```
# This is the network config written by 'subiquity'  
network:  
  ethernets:  
    enp2s0:  
      dhcp4: true  
    enp3s0:  
      dhcp4: true  
  version: 2
```

この場合、enp2s0 と enp3s0 の 2つがある。

このうち enp3s0 に IP アドレスを固定で割り振る。

(物理的に、どちらのポートが enp2s0, enp3s0 なのかは、差してみて確認)

以下のようにファイルを書き換える（インデントが意味を持つので注意）

```
# This is the network config written by 'subiquity'  
network:  
  version: 2  
  ethernets:  
    enp2s0:  
      dhcp4: true  
    enp3s0:  
      addresses:  
        - IP アドレス/プレフィックス
```

```

gateway4: デフォルトゲートウェイ
nameservers:
  addresses:
    - DNS サーバ IP アドレス
dhcp4: no

```

Ubuntu Desktop の方も `ethernets:` 以下の設定を同様に記述する。

以上の書き込みができたら、`vi` で保存し、下記コマンドを実行。

`vi` で YAML ファイルを書き込み後、下記コマンドで設定を「適用」する。

```
# netplan apply
```

- ネットワーク接続の確認

TCP/IP ネットワークの接続確認には、以下に示す `ping` コマンドを利用する。

```

# ping X.Y.Z.W ←相手の IP アドレス
PING X.Y.Z.W (X.Y.Z.W): 56 data bytes
64 bytes from X.Y.Z.W: icmp_seq=0 ttl=64 time=0.451 ms
:
(何行が表示されたら適宜 Ctrl-C で止める)

```

`ping` コマンドは、指定した IP アドレスの端末に、パケットを送り、相手から返信のパケットが届いた場合は、送信から受信までにかかった時間を表示する。エラーが出る場合は、送信に失敗したり、受信が行えないなど、正常に送受信ができなかった場合であり、トラブルシューティングを行う。

`ping` が正常に行えた場合は、Layer1, Layer2, Layer3 では問題なく通信が行えていることを示す。

クライアント OS のネットワーク設定

- Windows 10 の場合

Windows 10 では、左下スタートボタンを、右クリック→「設定」→「ネットワークとインターネット」→「アダプターのオプションを変更する」→「イーサネット」右クリック→「プロパティ」→「インターネットプロトコルバージョン 4 (TCP/IPv4)」選択→「プロパティ」にて、IP アドレス、サブネットマスク、デフォルトゲートウェイ、DNS サーバーを設定する。

「OK」をクリックし、「閉じる」をクリックする (OK と閉じるをして変更設定が有効になる)。

1.14 ネットワーク用ソフトウェアのインストール

Linux は、 apt コマンドでソフトウェアをインストールする。

```
$sudo su

まずソフトウェア情報を最新状態に更新
# apt update

インストール
#apt install ソフトウェアパッケージ名
(または、 apt-get install)

削除（アンインストール方法は各自で確認）
```

Windows はインストーラをダウンロード、実行し、インストールする方法と、「Microsoft Store」からのストアアプリの2通りがあり、ソフトウェアにより配布方法が異なる。

Mac は、インストーラをダウンロード、実行する方法と、アップルストアからインストールする方法とがあり、ソフトウェアにより配布方法が異なる。

Chrome は、Chrome アプリ、Android アプリ、Linux アプリが使えるが、Chrome はウェブストア、Android は Play ストア、Linux は apt コマンドでインストールする。

なお、上記以外のインストール方法もある（ZIP や tar で圧縮されたファイルを展開して、あるいは配布された実行ファイルをそのまま実行する方法（インストールしないで実行する方法）や、C/C++等のソースコードを make などでコンパイルしてインストール、実行する方法などがあり、Linux、Windows、Mac とも、このような方法あるが、後者は開発ツール（コンパイラや開発環境 gcc, make, Visual Studio, Xcode 等）が必要である。

1.15 SSH でのログイン、SCP でのネットワークファイルコピー

SSH (Secure SHELL) は、リモートログインを行ってサーバでシェル操作を行うためのソフトウェアである。

ssh コマンドを使って、リモート接続する。

```
$ ssh ユーザ名@IP アドレス
```

SCP (Secure Copy) でファイルコピーも行える。

```
ローカルからリモートへ
```

```
$ scp ファイル名(パス) ユーザ名@IP アドレス: サーバのファイルパス
```

例：カレントディレクトリにある file.txt をサーバの/etc/file.txt にコピーする

```
$ scp file.txt exp@192.168.0.5:/etc/file.txt
```

リモートからローカルへ

```
$ scp ユーザ名@IP アドレス: サーバのファイルパス ファイル名
```

カレントディレクトリに同じファイル名でコピーする場合は、

```
$ scp ユーザ名@IP アドレス: サーバのファイルパス .
```

でも良い

Windows は、PowerShell から、ssh, scp コマンドを呼び出しても良いが、Putty (ssh の代替) や WinSCP (scp の代替) を使うことが多い。

1.16 動作確認

- ケーブルチェックにて、正しくケーブルが作成されているか確認する。
- また、端末同士がネットワーク接続されているか確認を行う。
- ハブや NIC のリンクが正しく（グリーンランプ）点灯するか確認する。
- ip/ifconfig/ipconfig コマンドで、UP/Active/Running になっているか確認する
- ping コマンドで宛先コンピュータから正しくパケットが返信されるか確認する。ping はグループ内の各コンピュータ間、およびメインサーバ 192.168.0.1、Yahoo Japan www.yahoo.co.jp と行う。
- Windows は、デフォルトでは、ping に応答しないよう、Windows Firewall が設定されているため、Windows Firewall を無効にして確認する (Firewall を設定したままでも、ARP で確認することもできるが、適用場面はローカルネット内に限定される)。

1.17 考慮すべき点

今回の実験を行うにあたり、以下のようなことについて考慮する必要がある。

イーサネット規格 イーサネットには様々な規格があるが、それぞれの規格の比較（用いるケーブルの種類や利点・欠点など）を行い、どのような場合にはどのような規格が有効であるかを考える。10BASE の頃用いられた、共有バス型ネットワークと CSMA/CD の役割や、100BASE まで用いられたクロスケーブルの役割についても考えてみる。

ネットワーク機器の種類 OS と同様に、ネットワーク機器にもさまざまな種類があり、それぞれの用途や特徴が異なっている。どのようなネットワーク機器がどのような動きをするのか把握し、目的に応じて正しいネットワーク構築をする必要がある。また、それぞれのネットワーク機器には、どのような設定を必要かを考える。

ネットワークを構成するための情報 コンピュータ・端末を TCP/IP ネットワークに接続する際に設定しなければならない情報と、その役割を考える。

構築時・障害時の注意 構築する際、あるいはうまく稼働していないネットワークに対し、どのような手順で、どのような点を確認していくのが良いか、その考え方や理由について考えてみる。

1.18 実験内容 (3)

ネットワークに接続したサーバ以外のコンピュータについて、下記のことを行う。

- (1) プロキシの設定
- (2) OS のアップデート
- (3) 必要なソフトウェアのインストール
 - ftp, telnet, traceroute (tracert) を行えるようにしておく。

1.19 各OSのアップデートの方法

Windows

Microsoft 社の OS である Windows のアップデートを Windows Update と呼ぶ。手動で行う場合は「すべてのプログラム」から Windows Update を選択して行う。Windows は大規模なシステムで、数多くのアップデートが実施されており、リリースから年月の経たものでは、修正のためのファイルが 1GB 以上になることもある。こうしたことから、OS のインストールから最新版までの間に、いくつかのアップデートをまとめて実施するサービスパックが準備されることがある。また、サービスパックには、新機能が搭載されたり、性能向上がはかられるなど利便性が向上していることが多い。その反面、大きな変更がなされているので、業務に影響をおよぼす可能性が多く、その適用可否は十分に検討する必要がある。また、Windows Update は通常、再起動(時間がかかるものもある)が必要であり、さらには、重ねて Update をする必要がある場合は、数回の Windows Update および再起動が必要な場合もあり、長時間の業務停止が必要になる場合がある。

Windows Update を自動的に行うか否かの設定はコントロールパネルの Windows Update の項目から設定することができる。

Proxy 環境下の Windows Update

Windows Update は、HTTP (TCP ポート 80) および HTTPS (TCP ポート 443) を用いるが、Proxy 下の環境の場合、Internet Explorer 等のブラウザの Proxy を操作しても、Windows Update の Proxy には反映されない。

Windows Update の Proxy 設定は、管理者権限のコマンドプロンプト²⁰から、下記のコマンドを実行する。

```
netsh winhttp set proxy proxy-server="IP アドレス: ポート"
```

IP アドレスとポートは、1.11.1 節を参照しクライアント OS の設定と同じにする。

なお、Proxy の解除は netsh winhttp reset proxy コマンドで行える。

²⁰ コマンドプロンプトのアイコンを右クリックし、「管理者として実行」をクリックする。

Windows 7 のアップデート効率化

Windows 7 に、先に述べたサービスパックを適用し、オフラインでダウンロードしたアップデートファイルをまとめて適用する方法を述べる。

下記のメインサーバの FTP サイトの /pub/Windows ディレクトリの中に、サービスパックおよびアップデートファイルが ZIP 圧縮された形で置いてあるので、Explorer や Internet Explorer, コマンドプロンプトの ftp コマンドを使ってこれをダウンロードし、展開してからファイルにあるテキストファイル README.txt を見ながら実行して行く。

- X86 (32bit) 版 Windows 7 の場合の FTP ファイル URL
ftp://192.168.0.1/pub/Windows/Windows7update-x86.zip
- AMD64(64bit) 版 Windows 7 の場合の FTP ファイル URL
ftp://192.168.0.1/pub/Windows/Windows7update-x64.zip

数回の再起動を伴う作業が最後まで終了したら、Windows Update を行う。

MacOS X

MacOS X のアップデートは、りんごメニュー中の「ソフトウェア・アップデート」から行う。

Linux

Linux では、OS に固有の共通的なアップデートの仕組みはない。そのかわり、アプリケーションも含めたディストリビューション単位でのアップデートの仕組みが用意されており、ディストリビューションごとに、その内容や操作、アップデートに使用するソフトウェアなどが異なる²¹。

CentOS, Fedora Core などの RedHat 系 Linux では、標準のパッケージ管理ソフトウェアとして、yum が用いられている。Cent OS の「ソフトウェアの更新」も yum の GUI フロントエンドである。yum は、リポジトリと呼ばれるソフトウェアやデータの集積サイトに接続し、必要な更新があればダウンロードするようになっている。

また、Ubuntu 等の Debian 系 Linux では、apt update にて最新の更新情報を問い合わせ、apt upgrade を行うことで、更新がダウンロード・適用される。

```
# apt update (更新情報の入手)
# apt upgrade (OS・ソフトウェアの更新←行う場合は慎重に)
```

クライアント OS の HTTP Proxy サーバ設定

Web ページの閲覧などを行う場合、ブラウザにおいて HTTP Proxy サーバ（代理サーバとも呼ばれる、外部 network との接続を行うサーバのこと）を設定する必要がある。本節では、各クライアント OS において、HTTP Proxy サーバの設定方法を述べる。プロキシの設定を行うか否かは、その状況により異なるのでいつでも設定・解除が行えるように、その操作に習熟しておく必要がある。

²¹ CentOS の GNOME デスクトップの場合、「システム」→「管理」→「ソフトウェアの更新」にて、更新処理を行う。内部にて、yum コマンドが呼び出される。

一般には、プロキシの設定はアプリケーション（Web ブラウザなど）ごとに行う設定であるが、近年の Linux や Mac, Windows のデスクトップ環境では、その環境全体を行い、その環境に対応しているアプリケーションは、その設定に従うよう設計されていることが多い。

なお、本実験室の環境では、コンピュータは直接インターネットと接続されておらず、プロキシが下記のホストで提供されている。

Proxy の IP アドレス	192.168.0.1
Proxy の ポート番号	7999
Proxy ソフトウェア	Delegate
Proxy 対応プロトコル	HTTP, HTTPS, FTP

高知工科大学では、内部ホストの HTTP, HTTPS サービス向けに、proxy.noc.kochi-tech.ac.jp というプロキシサーバ（ポート 3128）が、提供されており、学内からのインターネット/学内アクセスはこれを利用することが原則となっている。

各 OS のプロキシの設定手順は、以下を参照のこと。²²

• Windows の場合

- Windows Internet Explorer（青い e のアイコン）を起動する。Windows 10 の場合は、「スタートボタン」→「Windows アクセサリ」→「Internet Explorer」（水色の e に黄色の線が入っているアイコン、濃い青色で黄色の線がない e のアイコンの edge は別のブラウザ）である。
- もし、ここで「インターネットエクスプローラ 8へようこそ」のウィンドウが表示されたら、「後で確認」を選択する。
- 「ツール」（「Alt」「T」のキーを順番に押すと現れる）→「インターネットオプション」→「接続」→「ローカルエリアネットワーク (LAN) の設定」の「LAN の設定」を選択する。
- その画面上で「設定を自動的に検出する」のチェックを「OFF」にし、「LAN にプロキシサーバを適用する」を選択し、HTTP Proxy サーバの IP アドレスとポート番号を入力する。
- もし、内部ネットワーク上のアドレス宛の通信に対し、HTTP Proxy サーバを利用したくない場合は「ローカルアドレスにはプロキシサーバを適用しない」を選択する。

Firefox や Chrome など、その他のアプリケーションも、IE の設定に従うよう設定することも可能である（Chrome のデフォルトはそのようになっている）。

• Macintosh の場合

- IP アドレスなどの設定時と同じく、左上のりんごアイコン→「システム環境設定」→「インターネットとワイヤレス」の「ネットワーク」を選択する。
- 次に「Ethernet」を選択し、「詳細」→「プロキシ」から「Web プロキシ」のチェックを入れ、「Web プロキシサーバ」に HTTP Proxy サーバの IP アドレスを入力し、コロン（“.”）の後の欄にポート番号を入力する。

²² プロキシは本来はブラウザの機能であるため、ブラウザ毎に設定を行いブラウザによって設定手順は異なる。

- CentOS(Linux) の場合²³

- 「Applications」 → 「System Tools」 → 「Settings」 → 「Network」
- 「Network proxy」を選択し, Method を「Manual」に設定する.
- HTTP, HTTPS, FTP 等, 必要とするプロキシのみ, プロンキサーバの IP アドレスまたはホスト名とポート番号の設定を行う (プロキシサーバで提供されていないサービスは設定しない方が問題が生じない).
- 最後に、「閉じる」.
- Ignore Hosts には, プロキシを適用しない (すべきでない) ホストを書いておくと, 例外的にこれらはプロキシを用いずに直接アクセスするようになる.

- Ubuntu Linux を含む多くの UNIX 一般のコマンドライン (シェル) 環境の場合

- 下記の環境変数 http_proxy, https_proxy を設定する.

```
# export http_proxy=http://192.168.0.1:7999  
# export https_proxy=http://192.168.0.1:7999
```

- 上記の設定は, シェルをログアウトするたびに消去されるので, ログインのたびに毎回必要である.
- 每回ログインするたびに, 同じコマンドを入力することを自動化したい場合は, 上記コマンドをシェルの初期化ファイル `/.bashrc` の最後に書いておくことで, 每回実行される. `.bashrc` は, そのユーザのホームディレクトリの直下にある.

1.20 考慮すべき点

OS のアップデートの適用可否 OS のアップデートについては, 常に無条件に適用をすべきか否かは, 考慮が必要である. まず, OS のアップデートは多くの場合システムの一時的な停止を伴う. 例えば, 適用後に再起動を要求されるなどである. サーバなどでは, 再起動のみであっても, 短くても数分, 大規模システムなど長い場合では 10 分以上かかる場合もあり, 再起動以外にサービスを停止する場合はさらに停止時間が伸びる.

次に, OS のアップデートにより, 様々な機能のデフォルトの設定や, ソフトウェアから OS の機能を使う際の条件, ソフトウェアが用いているライブラリの動作条件などが変更されるなど, 仕様が変更される場合がある. こうした場合は, 新たな OS に合うようアプリケーションソフトウェアも変更 (アップデート, 変更, あるいは再インストール) を行う必要があり, 大規模なサーバなどでは作業に数時間から数日かかる場合もある.

また, アップデートそのものに相性等の不具合や安全上の問題が発生する, あるいはアプリケーションソフトウェアとの相性により新たな問題が発生することもありうる. こうしたことは, アップデートによりシステムのサービスが停止する時間が増加し, サーバなど, システム停止が大きな影響を与えるような

²³ 正確には Linux で GNOME デスクトップ環境を用いている場合

重要なシステムでは、通常の業務の停滞などの経営上の問題を引き起こすことになる。こうしたことから、緊急を要するような、例えば非常に深刻なセキュリティホールが明らかになり、かつその問題点を用いるウイルスが急激に増加しているような場合などは別であるが、適用すべきか否かを慎重に判断する必要がある。具体的には、アップデートを行うに先立ち、そのアップデートはどのような問題点を修正し、どのような機能改善が行われるものかを確認し、運用中のシステムに必要なものか否かを判断する、他の情報を参照するなどし、不具合の報告などがないかを確認する、影響の少ない別のシステムに適用し試験的に運用を行ってみるなどして、適用の可否を判断する。

クライアントコンピュータにおいても、業務に毎日用いているコンピュータの場合、アップデートによる不具合の発生は、業務に大きな影響を及ぼす危険がある。このため、まず影響の少ないコンピュータで試験的にアップデートを実施し、ソフトウェアの動作やネットワーク接続に問題がないか確認するなどの作業を行う必要がある。

第 2 章

WWW システムと認証および暗号化

WWW は情報の公開を目的とした TCP/IP 上のサービスである。近年は、単なる情報公開にとどまらず、情報検索、電子メール、チャット、地図やルート、時刻表の検索、ショッピングやオークション、ホテルやチケットの予約など、さまざまなサービスの基盤としても使用されている。そのため、今日ではインターネット上のサービスのインフラの一つとなっている。WWW を介して提供されるサービスには纖細な情報を扱うものがあり、正当性を確認するための認証技術や第三者に内容が分からないように情報を変換する暗号化技術が併用される機会も増えている。暗号化技術は認証技術の基盤としても用いられ、WWW 上のセキュリティ確保のための重要な役割を果たしている。

2.1 WWW システム

World Wide Web (WWW, W3, the Web) は、インターネットにおいて利用されているサービスの一つである。

WWW はインターネット上で情報を公開・流通・閲覧する方法の一つであり、その根幹をなす概念にハイパーテキスト (hypertext) がある。ハイパーテキストは、文書を表現するテキストの中の語句に対して、関連する他の文書へ関連付けを定義し、ユーザはその関連をたどることで、関連する情報を閲覧することができる。この文書間の関連付けをハイパーリンク (hyperlink) と呼ぶ。現在の WWW では、テキストのみならず、画像、音声、映像など、多くのコンテンツを表現することができる。また、静的な情報だけでなく、動的な情報、インタラクティブなコンテンツの流通基盤としても用いられている。

現在の WWW は、1990 年、CERN¹ の Tim Berners-Lee により開発され、CERN の研究所内の論文・データ閲覧の手段として用いられた。この時、Web サーバ (web server, WWW サーバ, HTTP サーバ) および Web ブラウザ (web browser, Web クライアント, HTTP クライアント) として用いられたコンピュータは NeXT コンピュータであった。1993 年、NCSA² の Marc Andreessen³ により Web サーバである NCSA http サーバ、Web ブラウザである Mosaic が開発された。1994 年には、Tim Berners-Lee により World Wide Web Consortium (W3C) が設立され、WWW の仕様の標準化などを行うようになった。

現在では、多くの Web ページが公開されており、Web サーバ、Web ブラウザとともに多くのプログラムが提供されるようになっている。

Web サービスを用いた情報公開により、情報公開者はインターネットに情報を公開することが可能となり、全世界へ広く情報を発信することができる。インターネットの利用者は、それら世界中で発信された情報を自由に閲覧することができる。

一方で、インターネットが社会一般に広まったことにより、利用者層や利用者の考え方も様々なものとなっている。それらの中には、悪意を持つ利用者も存在する。こうした現在のインターネットの置かれてい

¹ the European Organization for Nuclear Research: 欧州合同素粒子原子核研究機構

² National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, イリノイ大学付属米国立スーパーコンピュータ応用研究所

³ その後、Netscape 社を創業

る状況下では、適切な情報の発信が重要になる。すなわち、公開しても良い情報なのか、公開すべきではない情報なのかを、個々の情報発信者が気を付ける必要がある。一度インターネットへ公開された情報は、インターネット上へ広く拡散することになり、仮に、公開すべきでない情報を公開してしまった場合、この情報を回収することは困難なものとなる。

また、全世界へ公開するのではなく、一部の利用者のみに限定した情報を公開を行いたい場合がある。このような場合には、Webサーバのアクセス制限や認証の機能を用いることで、一部の利用者のみに情報を公開することが可能となる。

WWWを構成する技術として、データ形式を定義するHTMLと、そのデータのやりとりを行うプロトコルであるHTTPがある。

2.1.1 HTML

WWWでよく用いられるデータは、HTML(Hyper Text Markup Language)と呼ばれる記述形式で書かれた文書である。HTMLは、SGML(Standard Generalized Markup Language)の書式を踏襲したマークアップ言語である。

HTMLは、ハイパーテキストを記述するためのマークアップ言語であり、ハイパーリンクを用いて他の情報への関連付けを行う。また、通常のテキストに対して、タグと呼ばれる記法で文書の構造やハイパーリンクなどの要素に関する意味づけを行う。

ハイパーリンクは、他のHTML文書だけでなく、JPEGなどの画像情報や、音声、動画像、テキストファイル、PDFファイルなど、コンピュータが取り扱う様々な形式のファイルに関連づけることができる。このためマルチメディア情報を取り扱うことにも適している。

2.1.2 HTTP

WWWサービスを実現するために用いられるプロトコルが、HTTP(Hyper Text Transfer Protocol)である。HTTPは、WebサーバとWebブラウザの間で、HTMLやその他の様々なファイルをやりとりするための転送手順を決めている、アプリケーション層プロトコルである。

HTTPは1999年に策定されたバージョン1.1において基本的な機能が整備された。HTTP/1.1はRFC2616として規定されたが、現在ではRFC7230からRFC7235として整理された。また、多重化などの拡張が施されたバージョンであるHTTP/2についてもRFC7540として策定された。

また、HTTPではHTML以外にも様々な情報の送受信を行うため、それらの属性を定め、正しく情報の受信側で認識・閲覧・再生できるよう、電子メールでも用いられるMIME(Multipurpose Internet Mail Extension)によりファイルの属性を定義し、送信時に通知している。

2.1.3 URI

WWW上のリソースを一意に表すものとして用いられるのがURI(Uniform Resource Identifier)である。URIはプロトコルに依存せず、インターネット上の任意のホストの任意の公開情報を指定することができる。クライアントは、Webサーバとのコネクションを確立すると、リクエストとして表示したいWWWページのURIを送信する。これに対しWebサーバは、自分が持つそのURIの示す文書をクライアントに送信する。

<スキーム>://(ログイン情報@)<ホスト名>(:ポート)/<絶対パス>
(例: <http://www.kochi-tech.ac.jp/index.html>)

スキームとは、データ取得のために用いる手段であり、具体的には http や ftp, https などのプロトコルを表し、:// の後に、ホスト名⁴ が続き、パスを絶対パスで記述する。⁵ ポート番号は、/etc/services に指定された通りの番号⁶ である場合は、省略する。

URL (Uniform Resource Locator) は URI のサブセットであり、明示的にリソースが存在する場所を指示したい場合に用いられる。WWWにおいては、URI として URL が用いられることが多い。

2.1.4 HTTP におけるリクエストとレスポンス

図 2.1 に示すように、HTTP のメッセージは、Web クライアントから Web サーバへの Request と、Web サーバからの返信である Response のやりとりでデータが交換される。トランスポート層プロトコルとして TCP を用いており、Web サーバのポート番号は 80 が一般に用いられる。

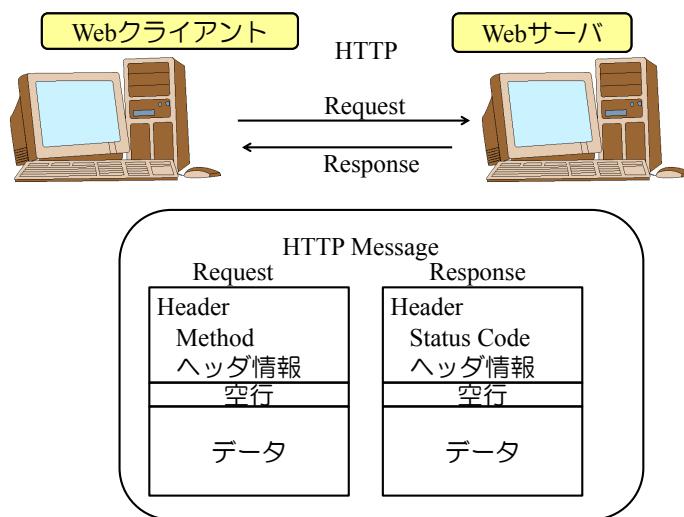


図 2.1 HTTP のプロトコル概要

まず、Web ブラウザは、HTTP Request を Web サーバへ送信する。HTTP Request は、ヘッダに続き、空行を 1 行送信した後、送信データがあればデータを送る。HTTP Request の冒頭にはメソッドと呼ばれるどのような情報交換をのぞむかの指定が含まれる。よく用いられるものは、GET（サーバからのデータ取得要求）などである。それぞれのメソッドでは、データの場所を指定する URI を記述する。たとえば、Web ブラウザは入力された URL を

```
GET <絶対パス> <メソッド> Host:<ホスト>
(例 GET /index.html HTTP/1.1 Host www.kochi-tech.ac.jp)
```

という GET メソッド（ここでは GET だが、他には PUT や POST 等がある）に変換し、Web サーバへリクエストを送る。

この GET メソッド以外のメソッドには以下のようなものがある。

⁴ インターネットでは FQDN (Fully Qualified Domain Name: DNS システムを用いた命名規則による世界唯一のホスト名) を用いる

⁵ 必ずしもサーバ上のファイルシステムのファイルのパスとは限らず、情報 (resource) が一意に (uniformly) 特定 (identify) できさえすれば、何でも良い

⁶ IANA で規定されている番号。HTTP は 80、FTP は 21、HTTPS は 443 など

HEAD

識別した URL から HTTP ヘッダ情報を検索(取得)する。

POST

指定した URL にデータを送る。

PUT

指定した URL に POST されたデータを入れて、既存のデータを置き換える。

HTTP Request を受信したサーバは、HTTP Response を送信する。HTTP Response もヘッダとデータからなり、空行1行で隔てられる。ヘッダには、冒頭に Status Code が記述され、Request に対する結果(状態)を番号で返す。例えば、200 は要求は受理され実行されること、404 は URI で指定されたデータが存在しないこと、500 はサーバ内でエラーが生じたことなどを示す。ヘッダには、その他、データの長さ、データの種類(テキスト、映像、etc.)や、テキストの文字コード、エンコーディング、データのタイムスタンプなどが記述され、空行の後、データ本体が続く。

たとえば、Web サーバは、

```
<HEAD>
<TITLE> ..... </TITLE>
:
</BODY>
```

というような、HTML のデータを返す(今述べた「GET /index.html」は、URL として該当する HTML データを要求していることを表している)。Web ブラウザは受け取った HTML のデータを解析し、表示する。

このような Web サーバとクライアントとの通信の様子は、telnet⁷ を使用して簡単に確認することができる(一般に HTTP のポート番号は 80 番を使用する)。

Cookie

HTTP はステートレスなプロトコルであり、サーバからは、個々のリクエストについて同一のユーザか否かを判定することができない(IP アドレスだけではクライアントを識別する情報としては不足していることに注意する)。Cookie は、このような問題を解決するために定義されている仕組みであり、サーバからブラウザに対して識別情報を送り、ブラウザはその情報を受け入れると、次回の同一サーバに対する接続から、リクエストにその情報を付与して送る。この情報を Cookie と呼ぶ。

2.1.5 Web サーバの種類

Web サーバはそもそも学術目的の文献を交換することを目的として開発された。最初の Web サーバはスイスの欧洲粒子物理学研究所(CERN)で開発された CERN httpd である。

現在、Web サーバのソフトウェアには Microsoft の IIS(Internet Information Server), Sun Microsystems 社の Java で書かれた Java Web Server, Igor Sysoev による nginx, そして、Apache などがある。

⁷ ネットワーク経由の仮想端末ソフトウェア

Apache

Apache とは NCSA⁸ httpd Ver.1.3 をベースに、機能拡張が図られた Web サーバである。その特徴は、機能が豊富で高性能であり、多くのプラットフォームに対応している。

Apache の構成は、モジュールの追加によって行っているので、必要な機能だけ使用する事ができ、最新の機能もモジュールを追加する事により簡単に使用することができる。また、メモリ管理がしっかりしており、速さ・メモリ共に効率的である。Apache は Solaris, Linux などの UNIX はもちろん、Windows (Win32 版), OS/2 などほとんどの OS で使用することができる。

Apache は HTTP Apache Server Projectにおいて開発が行われておりグループのメンバーは、世界中のボランティアで構成されている。ホームページは <http://www.apache.org/> である。

Apache の持つ代表的な機能を以下に示す。

Proxy サーバ

Proxy の機能を提供する。

CGI (Common Gateway Interface)

Web サーバのバックエンドでサーバアプリケーションを動かすとき、Web サーバからサーバアプリケーションを呼び出す標準的なインターフェース。

SSI (Server Side Includes), XSS (eXtended Server Side Includes)

Web サーバの機能の一つで、HTML ファイル中に別のファイルの内容やプログラムの実行結果を挿入する機能。XSSI は SSI の拡張機能。

バーチャルホスト

サービスホストが複数の IP アドレスをもったマルチホームホストである場合や、複数のドメインホスト名を持っている場合に、リクエストを受けた IP アドレスごと、あるいは HTTP リクエストの Host : フィールドの値ごとに異なる設定のサービスを行う機能。

認証機能

ホスト名、ユーザ名、あるいはパスワードによる認証機能によりアクセス制限を行う。

ハンドラ機能

特定の MIME タイプ、あるいはハンドラタイプが呼び出されたときの動作を定義したコード。

2.2 WWW における認証と暗号化

Web サーバ・クライアント間において用いられる認証技術として

- Basic 認証と Digest 認証
- 接続元による認証
- SSL/TLS によるサーバ認証・クライアント認証

が挙げられる。

⁸ National Center Supercomputing

2.2.1 Basic 認証と Digest 認証

Basic 認証は HTTP が備える認証手法であり、ユーザ名とパスワードを Base64 というエンコード方式によって変換して送信する。Base64 によるエンコードは暗号化ではないため盗聴や改ざんのリスクが大きい。一方、Digest 認証は、ユーザ名とパスワードを MD5 によってハッシュ値へと変換して送信する。

2.2.2 接続元による認証

HTTP Request の送信元 IP アドレスによる認証である。接続を受け付ける、あるいは拒否する IP アドレスに関するリストやルールを作成し、それに照らし合わせることで接続の可否を判断する。

2.2.3 SSL および TLS

まず、SSL (Secure Socket Layer) は、トランスポート層（ソケット）の暗号化を意図して作成されたプロトコルである。WWW が普及はじめた1994年に、Web ブラウザ・サーバソフトウェア開発の大手企業であった Netscape Communications 社⁹が、WWW の通信の安全性を高め商用利用を普及させることを目的に、SSL を開発した¹⁰。現在は、RFC 2246, 4346, 5246 において TLS (Transport Layer Security) として規格化されているが、現在でも一般に SSL の呼称で呼ばれることも多い。HTTP との組み合わせによる HTTPS (HTTP over SSL) がよく用いられるが、TCP を用いるトランスポート層プロトコルであれば、原理的にはほぼすべてのプロトコルにおいて、SSL による暗号化が可能である。電子メールに用いられる SMTP, POP, IMAP の SSL 化 (STARTTLS, POP over SSL, IMAP over SSL) もしばしば用いられる他、TELNET, FTP の SSL 化も存在する。stunnel という任意のプロトコルを SSL 上で通信させるためのソフトウェアもある。

SSL 上で通信を行う場合は、元の非 SSL プロトコルとは互換性がないため、サーバとクライアントとで同時に SSL 化を行う。公開サーバのように、クライアントが不特定多数で、SSL 化したプロトコルと元のプロトコルを併用する場合は、ポート番号を分け、異なるソケットで LISTEN させて用いる。例えば、HTTP が TCP 80 を用いるのに対して、HTTPS は TCP 443 を用いる。これについては、/etc/services や <http://www.iana.org/assignments/port-numbers> を参照すると良い。

OpenSSL は、SSL/TLS 標準の通信を実現するための、オープンソースの SSL 通信環境構築ソフトウェア・ライブラリである。最近の OS では、通信のセキュリティは重要な機能であり、Linux, BSD をはじめ、多くの OS で標準配布されている。

⁹ 世界初の画像表示機能を持った Web ブラウザである NCSA Mosaic を開発した Marc Andreessen と画像処理の高いグラフィックスワークステーションを開発する Silicon Graphics Inc. 社を率いていた James Clark が、高性能 Web ブラウザ Netscape Navigator を開発・配布・販売していた会社。Netscape Navigator はテーブル表示機能、フレーム、クッキー、SSL, JavaScript など、その後の Web サービスの発展を促すための先進的機能を多数備え、その後の Mozilla, Firefox へ発展する。Mozilla は Netscape のコードネームであり、Mosaic に代わるものという意図があったといわれる。NCSA とはイリノイ大学内のアメリカ国立スーパーコンピューティングセンターのこと、NCSA HTTPd という HTTP サーバも公開していた。NCSA HTTPd は後の Apache へと発展する。

¹⁰ それまで、TCP/IP における暗号通信は一般的ではなく、インターネットで通信の秘匿性を確保したい場合は、アプリケーションにおいてデータの暗号化を行う必要があった。

2.3 暗号方式

よく知られた暗号方式として、共通鍵暗号方式と公開鍵暗号方式がある。

共通鍵暗号方式は、情報の送信者と受信者が同じ鍵を用いる方式で、古くから用いられてきた方式である。例えば、古代ローマのシーザー暗号などが該当し、送信者と受信者は同じ鍵を、いわば合言葉のように秘密で共有し、これを知るもののみが暗号文を復号できる。

一方、公開鍵暗号方式は、1970年代に Diffie, Hellmann らにより考案され、Rivest, Shamir, Adleman による RSA 暗号で実装され、暗号鍵と復号鍵を別々に分けることで、暗号鍵は自由に配達することができ、誰も見られても問題ない。この鍵は世界に公開することができるので公開鍵と呼び、復号に用いる鍵は、受信者以外が持つてはいけないので（第三者に復号されることは、すなわち暗号が破られることになる）、秘密鍵と呼ぶ。

実際のアルゴリズムの設計においては、共通鍵暗号はビットの置換操作を行うことが処理の基本になるので、暗号・復号処理に必要な処理能力は公開鍵暗号に比較すると低い。一方、公開鍵暗号は、体の演算（多項式演算、すなわち加減乗除演算）を行うので、比較的高い処理能力が必要になる。

通信においては、鍵の配達の問題から、通信の開始時に公開鍵暗号方式で一度通信路を暗号化し、その通信路を用いて送受信者の間で共通鍵暗号に使う鍵を、その場で生成しありに共有し、その後、実際のデータ通信を高速に行うため、共有した共通鍵を用いて、共通鍵暗号方式で実際の通信を行う。

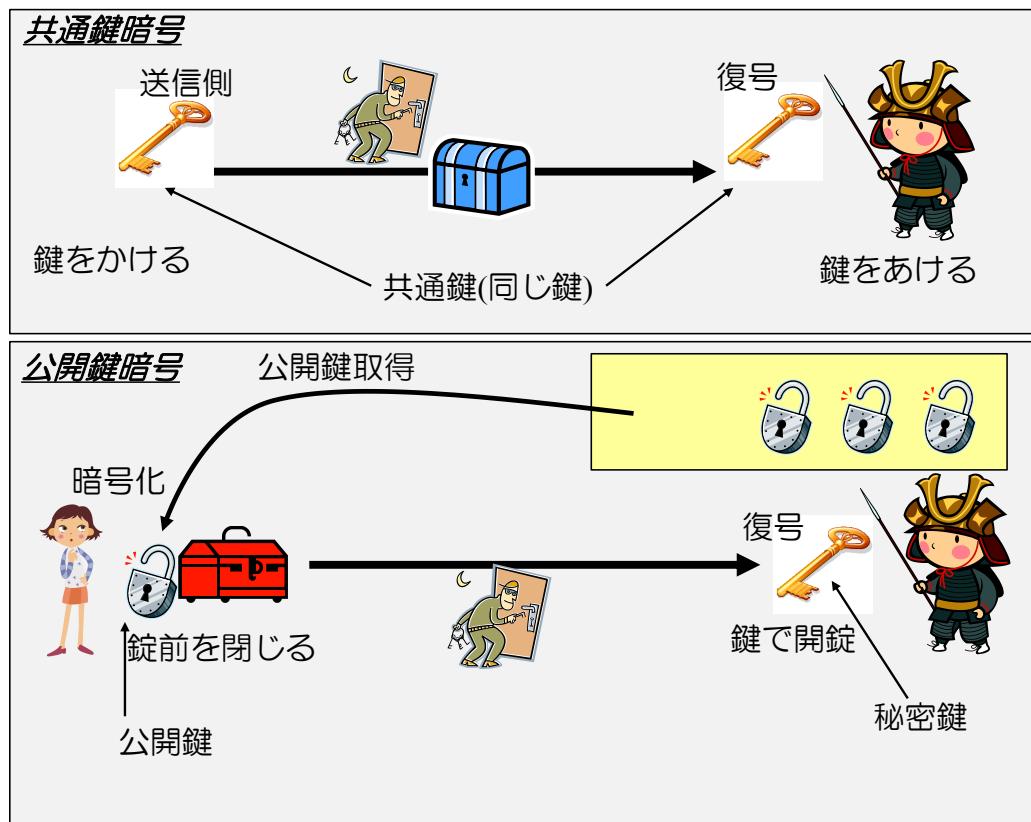


図 2.2 共通鍵暗号と公開鍵暗号

2.3.1 共通鍵暗号

AES

Advanced Encryption Standard の略で、アメリカ政府が策定した標準的な暗号方式である。策定に際して、世界の暗号研究者への公募が行われ、ベルギーの Daemen, Rijmen による Rijndael と呼ばれた暗号が採用されたので、Rijndael と呼ばれる場合もある。ヨーロッパ、日本でも標準に採用され、世界的に最も広く使われている暗号方式である。鍵長は、128, 192, 256 ビットから選択できる。

DES

Data Encryption Standard の略で、AES よりも以前にアメリカで標準だった暗号方式である。設計が古く鍵長が 56 ビットと短いこともあり、AES に比較すると安全性は低い。また、総当たり以外の解読方法も既に知られている。UNIX OS のパスワードのハッシュ化に用いられたが、現在の UNIX のパスワードのハッシュ化には MD5 を使うことが多い。DES を 3 回、異なる鍵で暗号化するトリプル DES (3DES, DES3) という派生方式も一時期使われた。

RC4

Rivest が開発した暗号方式で、無線 LAN の WEP, WPA, Windows 標準の PPTP 方式 VPN, リモートデスクトッププロトコル、Skype, PDF の暗号化など、広く用いられている。40 ビット (56 ビットと呼ばれることもある) のものと、104 ビット (128 ビットと呼ばれることもある) のものがあるが、40 ビットのものは鍵長が短いので現在は使用に適さない。また、128 ビットのものでも、総当たり以外の解読方法が知られている。暗号・復号の速度の点から広く使われているが、AES や DES に比較すると安全性は低い。

2.3.2 公開鍵暗号

RSA

Rivest, Shamir, Adleman による公開鍵暗号方式の初の実装で、現在でも主流の方式で広く使われている。発明した三者はチューリング賞を受賞している。特許により保護されていたので、一時期は他の方式 (PGP や DSA) も用いられたが、現在は RSA が一般的である。

PGP

Zimmermann による公開鍵暗号方式で、フリーソフトウェアとして広く配布され、電子メールの暗号化や、ファイルの暗号化などに用いられることがある。

2.4 認証手法

2.4.1 電子署名

公開鍵暗号は、通信路暗号化に用いる際は、公開鍵で暗号化したものを受け取ったときに秘密鍵で復号するが、逆に秘密鍵で暗号化したものを受け取ったときに公開鍵で復号することもできる。

これをを利用して、電子署名や認証が実現される。具体的には、ある公開鍵で復号化できるような暗号を作れる者は、対応する秘密鍵を持つ者のみであるので、公開鍵に対応する秘密鍵を持つ者が特定できるとい

うことから、送信者を特定することができる。第三者が送信者になりすまして、情報を偽造することはできない。

認証や電子署名で用いられる公開鍵のことを、証明書と呼ぶ。

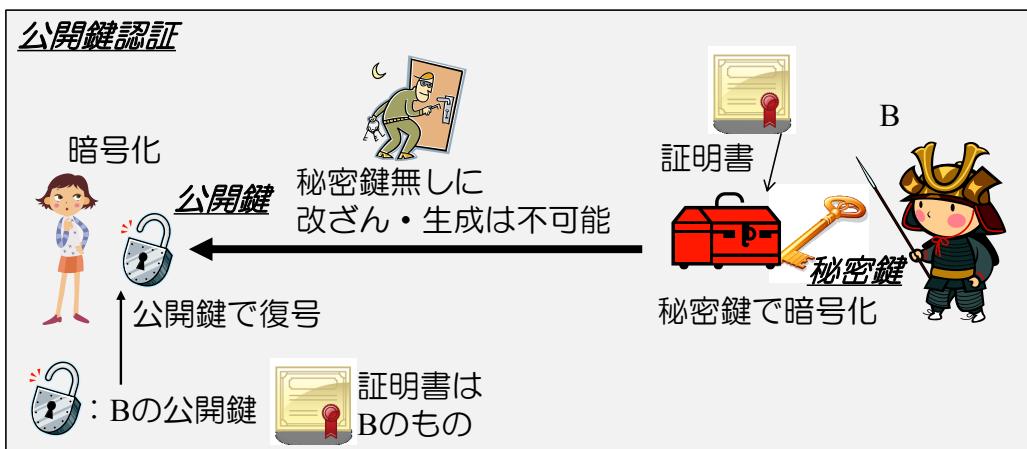


図 2.3 公開鍵暗号認証

しかし、証明書は確実にある認証対象の送信者であることを保証しなければならない。このためには、ある証明書に対して、対応する秘密鍵を持つ者が誰なのかを保証するための仕組みが別途必要である。これをPKI (Public Key Infrastructure) 公開鍵認証基盤と呼ぶ。

一般には、下記のようなものが用いられる。

- (1) 別途、安全な手段で証明書(公開鍵)を取得しておく
- (2) 信頼できる第三者により電子署名された公開鍵を取得する
- (3) 送信されてきた証明書を、自己申告のまま信頼する

3番目の方法は、実際には個人を認証として用いることはできず、暗号化通信のみを意図してサーバの公開鍵をそのまま受け入れる、サーバ証明書で使われる程度である。SSHやHTTPSのサーバ証明書では、簡易運用の際にこの方法もしばしば用いられる。しかし、問題点としては、悪意を持つ者がサーバのかたつたとしても、その証明書を受け入れてしまうことになり、サーバと認識しながら実は本来のサーバでない者へ情報を送信することになる。具体的にはログイン・パスワード情報を誤って送信してしまうなどがある。これをフィッシング(Phishing)と呼ぶ。このため、インターネットへ一般公開するサーバなどで用いることは望ましくない。

そこで、認証を行う際は、証明書の作成者から確実・安全に証明書を送付してもらう必要がある。SSLでは、送信者・受信者ともに確実に信頼できる第三者（その公開鍵=証明書は既知）が、その秘密鍵で証明書を暗号化し、証明書の受信者は、信頼できる第三者の公開鍵でこの証明書を復号し、復号できた場合は信頼する仕組みがある。これを公開鍵基盤と呼ぶ。信頼できる第三者の公開鍵は、あらかじめ全ての通信者が知っている。信頼できる第三者のことを、認証局(Certificate Agency: CA)と呼ぶ。具体的には、CAは、ペリサイン社などの公開鍵の認証を請け負う機関であり、それらの機関の証明書は、Webブラウザとともに配布されている。CAの秘密鍵による、未知の証明書への暗号化を、署名(Sign)と呼ぶ。ブラウザとともに

にも配布されすべてのWebクライアントが既知である証明書のことをルート証明書と呼び、ルート証明書を持つ機関を、ルート認証局あるいは、ルートCAと呼ぶ。

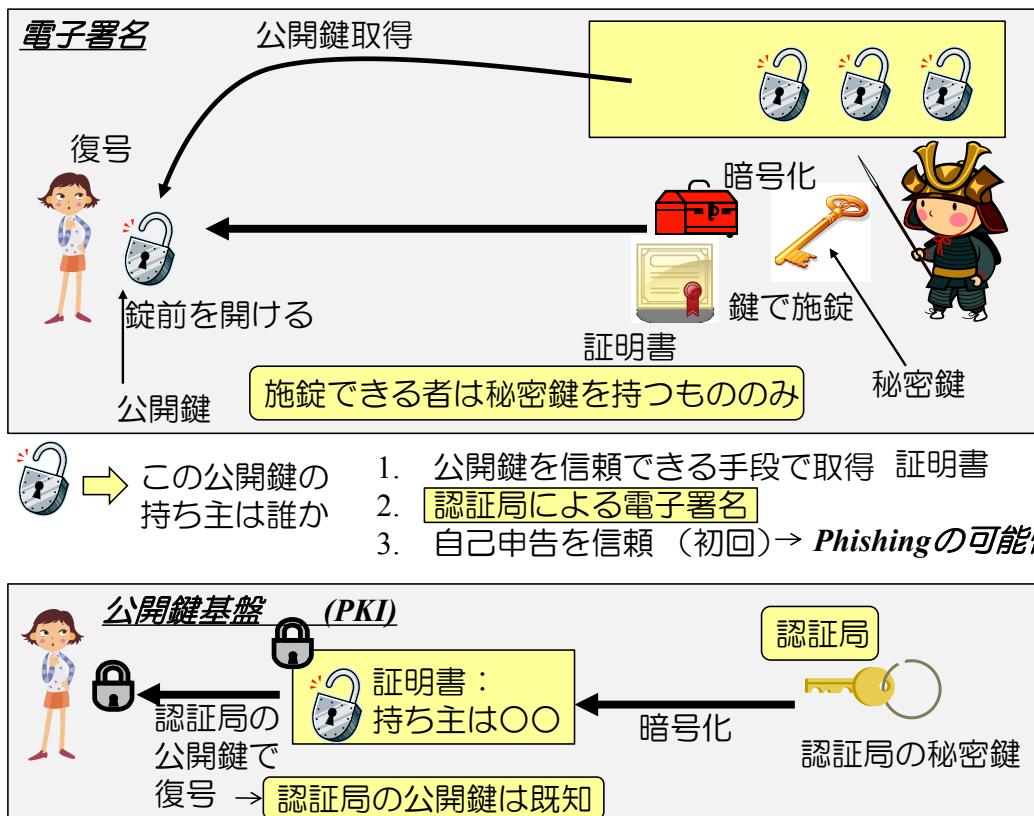


図 2.4 認証局による証明書への署名

なお、SSHの公開鍵認証では、認証局による署名の仕組みではなく、安全で信頼性が確保できる手段であらかじめ公開鍵を送ることを求められる。具体的には、サーバの管理者宛に、自分の公開鍵を電子メールで送付し、サーバの管理者はその文面ややりとり、電話などの別の手段での確認などを経て、サーバの所定の保存場所に、認定された公開鍵として保存される。

2.4.2 証明書と認証局

公開鍵暗号において、ある公開鍵で暗号化したものは特定の秘密鍵を持つ者にしか復号できないが、公開鍵だけを与えられても受信者が本当の受信者か、本来の受信者になりました者かは判定できない。このため、公開鍵で本当に安全な通信を行うためには、公開鍵を特定の受信者のものであることを証明する仕組みが必要である。具体的には、ある公開鍵に対して、送信者が信頼できる者が、この鍵が本来の受信者であることを証明する必要がある。

これに用いられるものが、電子署名である。つまり、世界中から信頼される者が、ある公開鍵に対して、確かに特定の受信者のものであることを証明する。具体的には、信頼された者（すなわち、公開鍵が分かれている）が、送信者の公開鍵と、送信者の情報（名前や所属機関など、個人を特定する情報、あるいはホスト名など、特定のホストであることを特定する情報）の組を、秘密鍵で暗号化する。信頼された者の公開鍵はわかっているので、これ復号した内容には偽造や改ざんの心配はない。このため、復号した内容も正し

い。これを公開鍵に対する電子署名（あるいは、単に署名）と呼び、これを保証する内容を証明書と呼ぶ。「信頼された者」とは、具体的には、認証局サイト (CA) を差し、VeriSign 社をはじめ、数多くの機関がある。これらの機関の公開鍵と証明書ははじめからブラウザなどにインストールされている。この証明書の信頼関係は、親子関係を作ることができるので、信頼された証明書を持つ機関は、別の機関の証明書を発行することができる。

なお、認証局は、各証明書の申請に対して、メールでのやりとりの他、電話や郵便、クレジットカードの引き落としなどを通して、証明書の保有者の身元を確認する。

2.5 実験内容 (1)

Web サーバの構築

今回の実験では、世界中で使われており、高機能で安定した動作をする Apache を用いて Web サーバを構築する。Server 用のコンピュータに、パッケージシステムを用いて Apache (Apache2) をインストールする。

コンテンツの作成と公開

HTML によるコンテンツを作成し、公開用ディレクトリへ配置する。その後、Web ブラウザ上でそのコンテンツを表示できることを確認する。

Apache2 の設定ファイルにて `http://192.168.0.121/` のように、ファイル名を省略して URI が指定されたときは、`index.html` ファイルが表示されるようになっており、これを変更することもできる（省略時には別のファイルを表示する、または、ファイル名省略時はエラーを表示する等）。ここでは、`index.html` を表示するようにしておく。

- Apache2 インストール直後に、`/var/www/html` の下に `index.html` としてデフォルトのものがあるので、これが表示できるか確認する
- デフォルトの `index.html` を `mv` コマンドで別名にリネームする
- グループ独自のコンテンツファイルを作成し、`index.html` として公開する

Web サーバでの認証設定

2種類のアクセス制限（送信元 IP アドレス、ユーザ認証）を施したコンテンツを作成する。

- コンテンツ公開ディレクトリに、2つのサブディレクトリを作成し、それぞれにコンテンツを作成
- 一方のコンテンツディレクトリには HTTP クライアントの接続元の IP アドレスによるアクセス制限
- もう一方のコンテンツディレクトリには Basic 認証によるユーザ認証

それぞれのアクセス制限の仕様は、下記の通り。

- 送信元IPアドレス制限によるものは、グループ内の固定IPアドレスのPC3台のみから閲覧可能にし、それ以外は閲覧できないようにする。
- ユーザ認証によるアクセス制限は、HTTP Basic認証により、限定されたユーザ名とパスワードを入力した場合のみ閲覧できるようにする（グループで決めたユーザ名・パスワードで良い）。

注意！ 認証用パスワードは、実験内で平文（ひらぶん）が表示される場所があるので、普段使用しているものとは異なるものにすること。

Wiresharkによるパケットモニタ

2種類のアクセス制限の設定を行い、その通信をモニタする。

構築したシステムが正しく意図した動作しているかをパケットモニタで確認する。

Wiresharkは高機能パケットキャプチャ（捕捉）・アナライザ（解析）である。スニファ（嗅ぐ）などとも呼ばれる。LANのパケットを調査する用途に用いるが、クラッキングツールにも悪用できるものであり、取り扱いには注意する。

プログラムはインターネットからもダウンロードできる。Linuxでは、aptコマンドなどからもインストール可能である。

これを用いて、HTTPのリクエスト・レスポンスの過程や、Basic認証の際のパスワードのパケット内の取り扱いを確認する。

- Wiresharkをクライアントにインストールし、サーバへの通信のモニタを行う。
- WindowsはメインサーバのFTPに、Cent OSはソフトウェアの画面からインストールすれば良い。
- HTTP接続にてBasic認証を行う通信のモニタし、パスワードの情報などを確認する。

2.6 実験内容(2)

2.6.1 HTTPSサーバの設定

HTTP通信の暗号化をSSL/TLSを用いて行う。

- SSL/TLS通信(HTTPS)対応設定

2.6.2 HTTPS通信のモニタ

WiresharkによってWindowsからサーバへのHTTPS通信をモニタする。

- HTTPS接続によるBasic認証を行う通信のモニタ

HTTP通信がどのようにになっているか、Basic認証の認証情報がどのようにになっているかを確認する。

2.6.3 動的コンテンツ

動的コンテンツを生成するための設定と動的コンテンツの配置を行う。

- Apache にて SSI, CGI が実行できる環境の設定
- 簡単な動的コンテンツの作成
- PHP の動作環境をサーバに作成

簡単な動的なコンテンツの作成として、時刻を表示するプログラムをそれぞれ下記の技術を用いて作成し、それぞれの技術的違いを考えてみよ。

- SSI
- CGI (用いる言語は何でも良いが、本テキストではシェルスクリプトを提供)
- PHP

次に、CGI でカウンタプログラムを作成せよ。カウンタプログラムはユーザからリクエストを受けるたびに、カウンタの値を 1 つずつ増加させ、そのウェブページに何回のリクエストがあったかを表示するものである。プログラムやカウンタを記録するデータファイルの書き込み権限に注意すること。

2.7 Apache のインストールと各種設定

2.7.1 Apache のインストール

Ubuntu への Apache インストール方法について述べる。

スーパーユーザになって下記を実行する。

```
# apt install apache2
```

以上で、インストールは終了し起動している。ps コマンドでプロセスを確認する。

```
# ps auxww
→ apache プロセスの行があるかを確認

(行が長くて流れる場合は、パイプ「|」で出力内容を grep コマンドに渡し、
grep コマンドで「apache」の行が残るところのみを抜き出し表示する)

# ps auxww | grep apache
```

apache2 プロセスがいくつか動作していることを確認する。もしない場合は、インストールに失敗しているか、実行に失敗している。

```
# ps auxww | grep apache2
root      4873 ... 起動時刻 /usr/sbin/apache2 -k start
www-data  4876 ... 起動時刻 /usr/sbin/apache2 -k start
www-data  4877 ... 起動時刻 /usr/sbin/apache2 -k start
```

systemctl コマンドでも下記のように確認できる。

```
# systemctl | grep apache2
apache2.service loaded active running The Apache web server

# systemctl status apache2

Active: active (running) の行を確認する
```

ps コマンドによるプロセス確認が最も信頼できる確認手段であるので、ps コマンドでプロセスが起動していることを、今後も全ての実験内容で確認するのが良い。

起動と終了は、下記で行う。

```
起動
# systemctl start apache2

停止
# systemctl stop apache2

再起動
# systemctl restart apache2
```

2.7.2 Apache の動作に関連するディレクトリ

下記に、Ubuntu で配置される主なディレクトリとそこにどのようなファイルが置かれるかを示す。

/etc/apache2

設定ファイルがある。apache2.conf が起動時に読み込まれる。

/etc/apache2/apache2.conf

起動時に読み込まれる設定ファイル。この設定ファイルには全体の構成のみが書かれ、動作する（LISTEN する）ポートや、機能の有効化・無効化などは別のファイルを読み込むようになっている。全体のデフォルトはここで設定できるので、セキュリティ情報等を他のファイルに引き継ぐよう設定できる。

/etc/apache2/ports.conf

動作するポート。デフォルトは TCP 80、HTTPS は TCP 443。

/etc/apache2/sites-available/000-default.conf

/var/www/html 以下の公開設定などを行う。実際には、apache2.conf の末尾にて sites-enabled のあるファイルが番号順に読み込まれるようになっているが、それらは site-available へのシンボリックリンクとなっている。

/var/www/html

公開用ファイル (HTML ファイルや画像ファイル, CGI 等のプログラムファイル) が置かれる。外部からアクセスされるディレクトリである。

/etc/apache2/envvars

Apache が動作するユーザ権限や、ログファイルの位置などを設定する。

/var/log/apache2

ログファイルがある。アクセスログ・エラーログには常に注意する。

なお、余談であるが、このような配置（デプロイ）は、Debian (Ubuntu の元となるディストリビューション) 独特のものであり、Debian のパッケージシステムで、モジュール単位での自動でのコンフィギュレーション、インストール、アンインストールに対応するためにこのようになっていると、apache2.conf ファイルの冒頭で説明されいる。

Apache をソースコードからコンパイル・インストールすると、デフォルトでは /usr/local/apache2 以下に全てのプログラム・設定・公開用コンテンツが配置され、httpd.conf にその設定をすべて記述するようになっている。

2.7.3 コンテンツの公開

/var/www/html 以下に、HTML ファイルを置くことで公開できる。

index.html

index.html は、URI でファイル名が省略されディレクトリ名のみの場合 (URI が / で終わる場合) に、デフォルトで表示される HTML ファイルである。

通常は、index.html にそのサイトの入り口（ポータル=表玄関）を記述する。

index.html ファイルが無い場合、Apache 設定の Directory /var/www/html の Options 設定で、Indexes の項目があれば、そのディレクトリのファイル一覧が表示される（しばしばこの設定は誤って情報漏洩する原因となるので注意）。この項目が無ければ、「403 Forbidden」エラーとなる。

2.7.4 HTTP 認証 (Basic) の設定

ユーザ・パスワードを入力しての認証つきアクセスの設定は下記のようを行う。

まずは、htpasswd でユーザ・パスワードの設定を行う。下記は、ファイル.htpasswd にユーザ情報を格納する例である。

```
# htpasswd -c /etc/apache2/.htpasswd testuser
```

-c は .htpasswd がまだ無い時に、作成 (create) するオプションで、既に存在していれば必要ない。“testuser”というユーザを作成し、そのパスワードを尋ねてくるので入力する。

/var/www/html の下に basic というディレクトリを作成し、ここにアクセス制限をかける場合の設定例を示す。

```
# vi /etc/apache2/sites-available/000-default.conf

(下記を <VirtualHost *:80> ディレクティブ内に追加)

<Directory /var/www/html/basic>
    AuthType Basic
    AuthUserFile /etc/apache2/.htpasswd
    AuthName "basic auth"

    Satisfy any
    Order deny,allow
    Deny from all
    Require valid-user
</Directory>
```

2.7.5 IPアドレスによるアクセス制限の設定

社内専用ページや研究グループ内のみ閲覧可能な内部専用ページの作成には、IPアドレスによる制限などが用いられる。

/var/www/html の下に inside というディレクトリを作成し、ここにアクセス制限をかける場合の設定例を示す。

```
# vi /etc/apache2/sites-available/000-default.conf

(下記を <VirtualHost *:80> ディレクティブ内に追加)

<Directory /var/www/html/inside>
    Order deny,allow
    Deny from all
    Allow from サブネット/プレフィックス
</Directory>
```

サブネット/プレフィックスには、192.168.100.0/24 のようにアクセスを許可する IP アドレスをサブネットおよびサブネットマスクのプレフィックス表記で設定する。この例では、192.168.100.0 から 192.168.0.255 までの IP アドレスが指定される。

1つしかない場合は、プレフィックスを指定しない。また、2つ以上を指定する場合は、Allow from の行を複数書く。

2.7.6 HTTPS による暗号通信設定 (modssl)

Ubuntu の Apache で SSL/TLS を有効にするために必要なモジュールを使用するための設定を行う。

- モジュール mod_ssl を有効にする。
- conf ファイルで、SSL を有効化する。
- Apache の再起動

```
# a2enmod ssl
# a2ensite default-ssl
# systemctl restart apache2
```

HTTPS で通信できることを確認する。

ただし、通信そのものは暗号化されているが、証明書は乱数で作成しただけの真正のものではないため（俗にオレオレ証明書などとも呼ばれる）、コミュニティ内ののみの通信であればまだ良いが、このまま公開運用するのは問題がある。

SSL 設定ファイル /etc/apache2/sites-available/default-ssl.conf の下記の 2 行が、それぞれ、証明書（署名された公開鍵）、秘密鍵となっていて、現在は、make-ssl-cert generate-default-snakeoil コマンドが自動的に実行されて作成された、PEM、KEY ファイルが指定されている。

```
SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
```

ブラウザからアクセスした際に、認証されていない証明書である旨の警告が出る（正規のサイトを詐称しているフィッシングサイトと区別がつかない）。

なお、サイトのコンテンツ (DocumentRoot) 等の設定内容は、000-default.conf とは別のファイルになるので、内容をあらためて確認し、必要があれば設定を行う¹¹。

2.8 動作確認

Web サーバ

全ての設定を終了後、起動を行い、クライアントコンピュータからブラウザを起動し、ホームページが表示できるか確認する。

また下記を確認する。

- URI は設定したドメイン名・ホスト名を使用した URL とする。
- 意図したコンテンツが表示されることを確認すること。
- アクセス制限（ユーザ認証、IP アクセス制限）をかけたものは、各グループで別途 URI を指定し、確認を行う。ただしくアクセス制限が行えているか、TA などの他のコンピュータなどからのアクセスも確認する。

¹¹ default-ssl.conf

- また、Apacheのログも確認し、アクセスの痕跡、アクセスが成功したか否かなどを全てチェックし、作業記録に残すこと。意図しないアクセスなどがあればこれも確認する。
- HTTPS通信も確認する。
- この時点で、自己署名証明書のため、ブラウザからエラーが出ること、またそのエラーの内容を理解しておく。
- WiresharkにてBasic通信の認証内容（パスワード等）が、HTTPでは表示され、HTTPSでは暗号化されていることを確認する。
- httpsでもアクセス制限が同様にかかっているか。

2.9 考慮すべき点

WWW・Webサーバ

- HTTPが広く用いられる理由。
- データ公開・転送にHTTPを用いる利点・欠点。
- バーチャルホストはどのように用いられるか。
- 自己署名証明書では、どのようなことは守られ、どのような危険があるか。

これまでの確認

サーバの環境変数の設定

環境変数や良く用いるシェルのコマンドなどを、あらかじめ初期化ファイルに記述することで、ログイン時に自動的に設定することができる。

初期化ファイルは、Linux 標準シェルの Bash であれば `/.bashrc`, BSD 系標準の Csh であれば `/.cshrc`, 商用 UNIX や古い Bsh 系であれば `/.profile` である。

例えば、`http_proxy`. `https_proxy` を root の環境変数に設定しておきたい場合は、下記のようとする。

(2021 年度の情報実験室ではプロキシは必要ない)

```
$ sudo su    (システムによっては su)
# vi ~/.bashrc
```

下記を末尾に追記。

```
export http_proxy=http://proxy.noc.kochi-tech.ac.jp:3128
export https_proxy=http://proxy.noc.kochi-tech.ac.jp:3128
(KUT の場合 https のプロキシも http で指定する)
```

以上で、毎回入力する必要がなくなる。

第3章

ルータ・スイッチ

3.1 目的

中規模以上のネットワークを構築する際は、ネットワークをある程度の大きさごとに分割する方が良い。この理由は下記の通りである。

IP アドレスは、ネットワーク部とホスト部に分かれており、同じネットワーク部を有する IP アドレスの端末は、同じ LAN¹ を構成する。同一 LAN のコンピュータ同士で通信を行う場合は、ARP (Address Resolution Protocol) を用いて、宛先 IP アドレスから、宛先端末の MAC アドレスを取得し、イーサネットフレームを送信してパケットを送る²。

異なる組織に属する端末は、お互い異なるネットワーク部を持つ IP アドレスを持つので、1台以上のルータを介して通信を行う。しかし、大きな組織においては、下記のような理由で、組織内で更に LAN を分割する方が良い。

- (1) 管理上の理由によりネットワークを分割したい場合。例えば、セキュリティなどの管理ポリシーが異なる場合や、部署間で情報のやりとりに制限を設けたいため、通信トラフィックを分けたい場合。
- (2) 物理的な距離の制限で、データリンク層に用いるプロトコルでは、距離が届かない場合（現在は、光ファイバやイーサネットのスイッチにより、データリンク層の距離制限は事実上無い）。
- (3) SMB などのブロードキャストを用いるプロトコルが、大きな LAN 全体に送信されることが望ましくない場合。

特に近年は、上記の(2)はイーサネットスイッチと長距離光イーサネットの普及で必要がなくなり、1Gbps や 10Gbps の普及で帯域上の理由もあまりない。むしろ、(1) のように異なる組織の通信がなるべく干渉しないようにしたい、組織外からのあるいはセキュリティを強化したいなどの理由でサブネット分割を行う。例えば、大きなネットワークで(3)を許すと、Windows などでは数百以上の PC やプリンタ、ファイルサーバなどが一度に見えるようになってしまい、管理上好ましくない。

このような場合に、IP アドレスのホスト部のうち上位ビットの一部分を、ネットワーク部とみなし、複数の組織間でネットワーク部の異なる IP アドレスを割り当てることで、複数の IP ネットワークを作り、それらのネットワークをルータで接続することで、複数のネットワークからなる IP ネットワークを構築する。もとのネットワークアドレスから、複数のネットワークアドレスを作ることをサブネット分割という。

インターネットは、ルータを介して異なるネットワーク部を持つ LAN を接続していったものであり、正しくルーティングの設定を行うことで、世界中の全ての LAN と接続され、世界中の全ての端末と相互に通信が可能になる。

¹ 「LAN」の定義については世界中でコンセンサスのとれている厳密な定義が存在するわけではなく、文脈によって意味が多少異なるが、ここでの「LAN」は、一つのデータリンク層ネットワーク、例えばイーサネットで接続されたネットワークという意味で用いる。すなわち、ブロードキャスト（より正確には、L2 のブロードキャストフレーム、あるいは L3 のリミテッド（ローカル）ブロードキャストパケット）が届く範囲を意味する。ちなみに、ブロードキャストが届く範囲をブロードキャストドメインと呼ぶ。ルータ（ルーティング）を用いずに直接通信できる範囲のネットワークとも言える。

² データリンク層にイーサネットを用いない場合は、必ずしも ARP を行わない

表3.1 ホスト部のうちネットワーク部に使われるビット数とサブネットの大きさ(7ビット以降もホスト部のビットが2ビットになるまで続く)

ビット数	サブネット数	ホスト台数(元のネットワークに対する比率)
1	2	1/2
2	4	1/4
3	8	1/8
4	16	1/16
5	32	1/32
6	64	1/64

3.2 TCP/IPネットワークのルーティング

既に説明してきたように、IPネットワークではIPアドレスを全世界のコンピュータにユニークに割り振り、IPアドレスを送信元コンピュータから宛先コンピュータまで、(宛先の)IPアドレスのみを用いて転送していくことが基本である。IPネットワークでは、データはIPパケットに分割され、このパケットが多くの中継機器を介して、宛先IPアドレスまで転送されていく。IPパケットは、ヘッダとデータ(ペイロード)に分けられる。ヘッダは、郵便における宛名・差出人・消印のようなもので、送信元から宛先まで転送される過程で用いられる情報である。ヘッダの後に、実際に転送したいデータが含まれる。パケットが宛先まで無事に到着すると、ヘッダは破棄され、データの部分がオペレーティングシステムに渡される。

途中のパケット中継機器では、隣接する中継機器から送信されたパケットを受信し、宛先IPアドレスを見て、次に転送すべき中継機器(次ホップ:Next Hop)を隣接する中継機器から選択し、その機器へ送信する。この経路選択およびパケットの転送動作をルーティングと呼び、ルーティングを行う中継機器をルータと呼ぶ。

ルータが、IPアドレスから対応する次ホップを選択する際、IPのアドレス長は、IPv4で32ビット、IPv6では128ビットあるため、全てのIPアドレスと対応する次ホップを記憶することは現実的ではない。このため、IPアドレスは、ネットワークアドレス部とホストアドレス部に分けて構成され、一つのまとまった組織は、一つの共通するネットワークアドレスを持ったIPアドレス群を取得し、その組織内の個々の端末に対して、ホストアドレスを変えながらIPアドレスを割り振る。このようにすることで、パケットを中継するルータでは、IPアドレス全てを見る必要はなく、ネットワークアドレス部のみを見て次ホップを選択できる。ルータが記憶しているネットワークアドレス部とそれに対応する次ホップの表を、ルーティングテーブルと呼ぶ。

表 3.2 IP アドレスのクラス（現在はクラスレスに移行しており、クラスの概念は歴史的なもので、現在はクラスの囚われずにネットワークが割り当てられる）

クラス	第1オクテットの値	ネットワーク部	ホスト部	
クラス A	0-127	第1オクテット	第2~4オクテット	
クラス B	128-191	第1・2オクテット	第3・4オクテット	
クラス C	192-223	第1~3オクテット	第4オクテット	
クラス D	224-239	N/A	第1~4オクテット	マルチキャストアドレス
クラス E	240-255	N/A	第1~4オクテット	未割り当てる

表 3.3 （旧来のクラスフルネットワークにおける）各クラスのネットワーク数・収容可能ホスト数

クラス	最大ネットワーク数	1 ネットワーク内の最大ホスト数
クラス A	128	16,777,214
クラス B	16,384	65,534
クラス C	2,097,152	254

3.2.1 クラスフルルーティング

IPv4 では、IP アドレスのネットワーク部とホスト部は、32 ビットからなっている。これを 1 オクテット (8 ビット) ずつに分け、第1オクテットから第4オクテットまでをドットで区切り、それぞれのオクテットを 10 進数で表す (dotted decimal notation)。IP アドレスの第1オクテットの値により、IP アドレスは 5 つのクラスに分けられる。クラスにより、IP アドレスのネットワーク部とホスト部の長さが決まる。

IP アドレスのうちホスト部のビットがすべて 0 であるような IP アドレスをネットワークアドレスと呼ぶ。ネットワークアドレスは、ネットワークそのものを示すアドレスであり、通常ホストやルータのインターフェイスに付与することはない。

クラスに基づいて、ネットワーク部を決定しルーティングテーブルを構築する方法をクラスフルルーティングと呼ぶ。

ルータのルーティング動作は以下の手順で行われる。

- (1) ルータのいずれかのインターフェイスからパケットが受信される
- (2) 受信したパケットの宛先 IP アドレスを読み出す
- (3) 宛先 IP アドレスとルーティングテーブルのあるエントリのネットマスクとを AND 演算し、その結果がそのエントリのネットワークアドレスと一致するか調べる
- (4) 前項の操作をすべてのエントリについて行う
- (5) 一致したエントリのうち、最も長いネットマスクのエントリの経路を採用する。このエントリに書かれている、次ホップルータの IP アドレスを経路として採用する (Longest Match と呼ぶ)
- (6) 前項の操作で、等しいネットマスク長の経路が複数マッチした場合は、最も小さいメトリック (距離、またはコスト) の経路を採用する
- (7) 採用した経路の次ホップルータへパケットを送信する
- (8) 一致するエントリが 1 つもない場合は、すなわちパケットの宛先 IP アドレスへの経路をルータが知らない場合、ルータはパケットは破棄し、パケットの送信元アドレスに宛てて、ICMP(type 3: Destination Unreachable) パケットを送信する。

例えば、ルーティングテーブル上で、

- 宛先ネットワークアドレス : 192.168.1.0
- 宛先ネットマスク 255.255.255.0
- 次ホップルータ 10.1.2.3

というエントリがあった時、192.168.1.13 を宛先 IP アドレスとするパケットが受信されたら、ネットマスク 255.255.255.0 とその宛先 IP アドレスとを AND 演算すると 192.168.1.0 となり、宛先ネットワークアドレスに一致するので、他に一致する経路が無ければ、この経路が採用される。

ネットワークアドレス 0.0.0.0, ネットマスク 0.0.0.0 のエントリには、全ての IP アドレスが一致するが、ネットマスク長が 0 であるため最も優先順位が低いエントリである。このエントリがある場合、他のエントリに該当することがなかった全てのパケットに対して、この経路が採用される。これをデフォルトルート、デフォルトゲートウェイなどと呼ぶ。

逆に、ネットワークアドレスとして例えば 192.168.2.33, ネットマスク 255.255.255.255 のエントリの場合、マスク長は最も長い 32 ビットであるから、宛先 IP アドレスとして 192.168.2.33 を持つパケットはこのエントリが最優先で採用される。これは単一のホスト専用のルートのため、ホストルートと呼ぶ。

このようにして、インターネット上のすべての IP アドレスについて、適切に次ホップルータを選択できるようにルーティングテーブルを構築する作業が、ルータの基本的な運用作業である。

3.2.2 クラスレスルーティング

クラスフルルーティングに対して、ネットワークアドレスとサブネットマスクを用いて、クラスの大きさに依存せずに任意の大きさのサブネットマスクを持ったネットワークの経路情報を、ルーティングテーブルに設定する方法をクラスレスルーティングと呼ぶ。

関連する技術としては、1つの（クラスフルな）ネットワークのネットマスクを延長し、ネットワークを複数に分割するサブネット化、そのネットマスクをネットワーク毎に変化させる VLSM (Variable Length Subnet Mask), 複数のネットワークを束ねる CIDR (Classless Inter-Domain Routing)³などがある。

3.2.3 ルーティングテーブルの構築の種類

ルーティングテーブルの構築には、大きく下記の2種類がある。

静的ルーティング 管理者がルーティングエントリを入力するもので、ルーティングテーブルは管理者が設定した状態のまま運用される。再設定しない限り、経路が変化することはない。「静的 (static)」とは、変化しないという意味である。ルーティング情報は通常、手動 (Manual) で入力する必要がある。

動的ルーティング ルーティングプロトコルを用いて、ルータが近隣のルータとルーティング情報を交換し、隨時ルーティングテーブルの更新を行う。ルーティングテーブルはネットワークの変化に応じて、随時更新され変化していく。すなわちある経路が遮断されたら、ルーティングテーブルの経路が、別のルータを経由するように変更されるなどの処理が行われる。管理者は、近隣のルータの管理者と、用いるルーティングプロトコルや設定を調整し、動的ルーティングプロトコルを用いる設定を行う。近隣のルータ同士で同一のルーティングプロトコルを用いることで、他のルータからルーティング情報を取得し、ルーティングテーブルは自動 (Automatic) で更新する。

³ より小さなネットワークに分割するサブネットに対して、複数の（隣接する）小さな（クラス C）ネットワークをまとめて大きなネットワークにすることからスーパーネットと考えることもできる。ルーティングテーブルに現れるネットワークは大小様々なものとなり、クラスの概念は最早ないことから、クラスレスルーティングと呼ぶ

静的ルーティングは、ネットワーク管理者がルーティングテーブルを構築する方法である。また、ルーティングテーブルが自動的に変更されないため、経路が不安定になることがない。ネットワーク接続の構成が変化せず、管理者が全ての経路情報を管理できる場合は、あらかじめ必要な経路を全て静的に設定することで、安定した通信を行うことができる。欠点として、ネットワーク接続の状態が変化した場合、管理者が新たな経路情報を設定するまで、ルーティングテーブルは変化しないため、通信に障害が出るなどの不具合がある。

ルーティングの登録に必要となる情報は、宛先ネットワークと次ホップルータの IP アドレスである。宛先ネットワークは、ネットワークアドレスとネットマスクで与える。次ホップルータの IP アドレスは、自ルータのインターフェイスのいずれか 1 つと同じネットワークに所属している必要がある。

動的ルーティングでは、管理者がどのルーティングプロトコルを用いるか (OSPF, RIP など⁴) を決め、参加するネットワークをどのようにするかやパラメータなどを設定しておく。

3.3 実験内容 (1)

本章において必要となる作業は下記である。

- 現状のネットワークの確認
 - traceroute コマンドにて、他のグループのサーバや www.yahoo.co.jp などへの経路を確認する (Windows の PowerShell またはコマンドプロンプト cmd なら tracert)
 - Windows のエクスプローラー (explorer) で「ネットワーク」をクリックし他のコンピュータの状況を確認
- ルータの設置による L3 ネットワーク分割。各グループとバックボーンとを接続するルータで、静的にルーティングテーブルを構築し、自グループと他グループおよびバックボーンと通信が行えるようにする。
 - ルータへのコンソール接続
 - * コンソール (CONSOLE) ケーブル (水色) を、片方の Dsub 9 ピンの形状のコネクタをサーバのシリアル端子 (RS-232C 端子, Dsub 9 ピン) に、RJ45 コネクタ (LAN と同一の形状) をルータの CONSOLE 端子に接続する。
 - * サーバから cu コマンドで接続する。
 - * シリアル端子の無いノートパソコン等から接続する場合は、RS-232C-USB 変換コネクタを使いコンソール接続をする（本実験では用いない）。
 - ルータの初期設定
 - サブネット（ネットワークアドレス、ネットマスク）の決定、IP アドレスの決定
 - ルータへの IP アドレス付与
 - ルータでのルーティングテーブルの作成
- ネットワークアドレス変更に伴う各 PC のネットワーク設定の変更

⁴ RIP には、RIPv1, RIPv2 があり、他にも旧来の IS-IS, シスコ社製品でのみ用いる IGRP, EIGRP, インターネット全般の経路制御用の BGP (BGP4) などがある

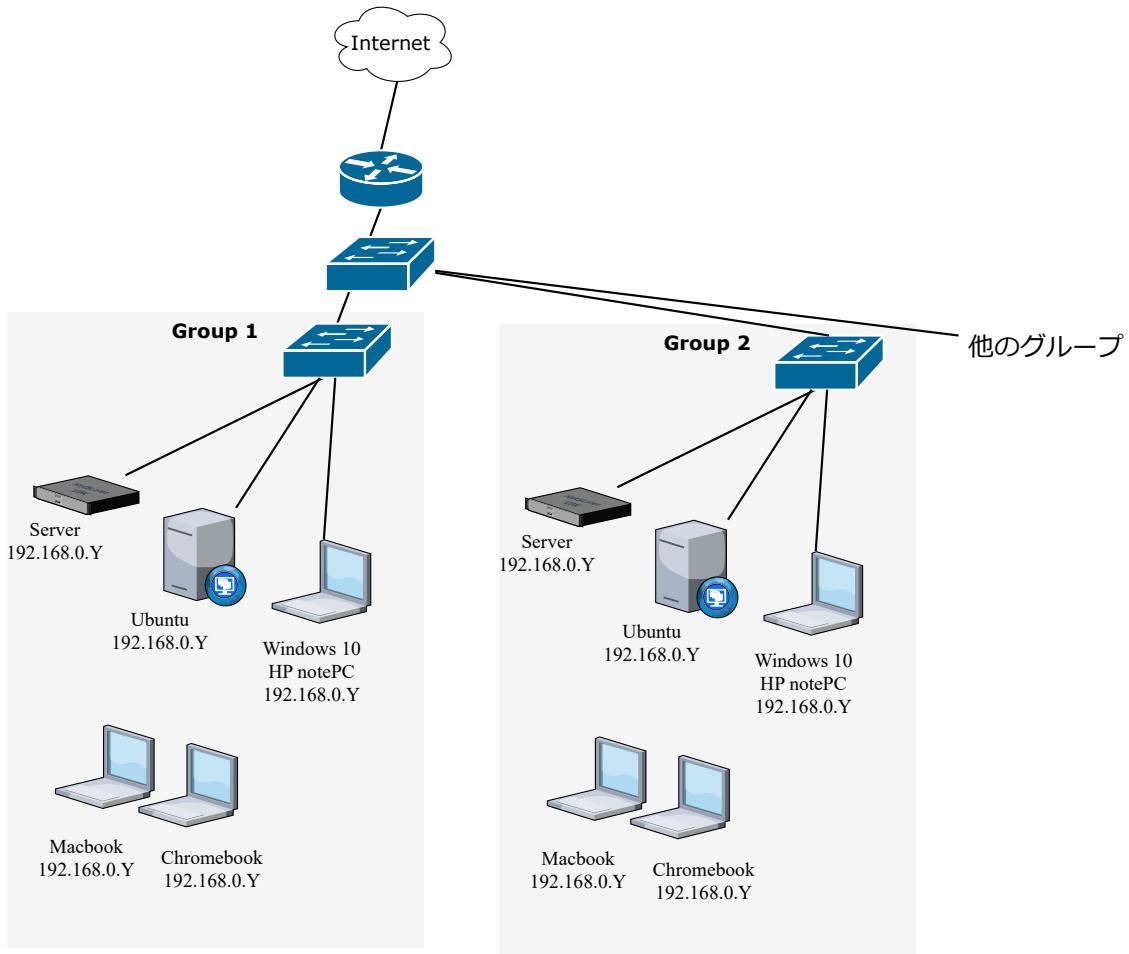


図 3.1 既に構築されているネットワーク。1台のインターネット接続ルータの配下に、L2スイッチのみでネットワークが構築されている。表中のYは、グループ番号が10の位となるアドレス。

- DNS のゾーン設定変更 (IP アドレス変更)
- Apache の IP アドレス制限の変更
- Postfix の mynetworks の変更

現在、全端末が同一ネットワーク (192.168.0.X/24) となっている実験室全体のネットワークに対してサブネット分割を行って、各グループごとにネットワークを分割する。クラスBのアドレスである172.21.0.0/16を256個のサブネットに分割し、表3.4のように、各ネットワークに割り当てる。また、グループ内ネットワークでのIPアドレスは図3.3のように割り当てる。

これらのネットワークを構築し、各ルータで静的ルーティングテーブルの設定を行い、正常に通信が行えるようにする。

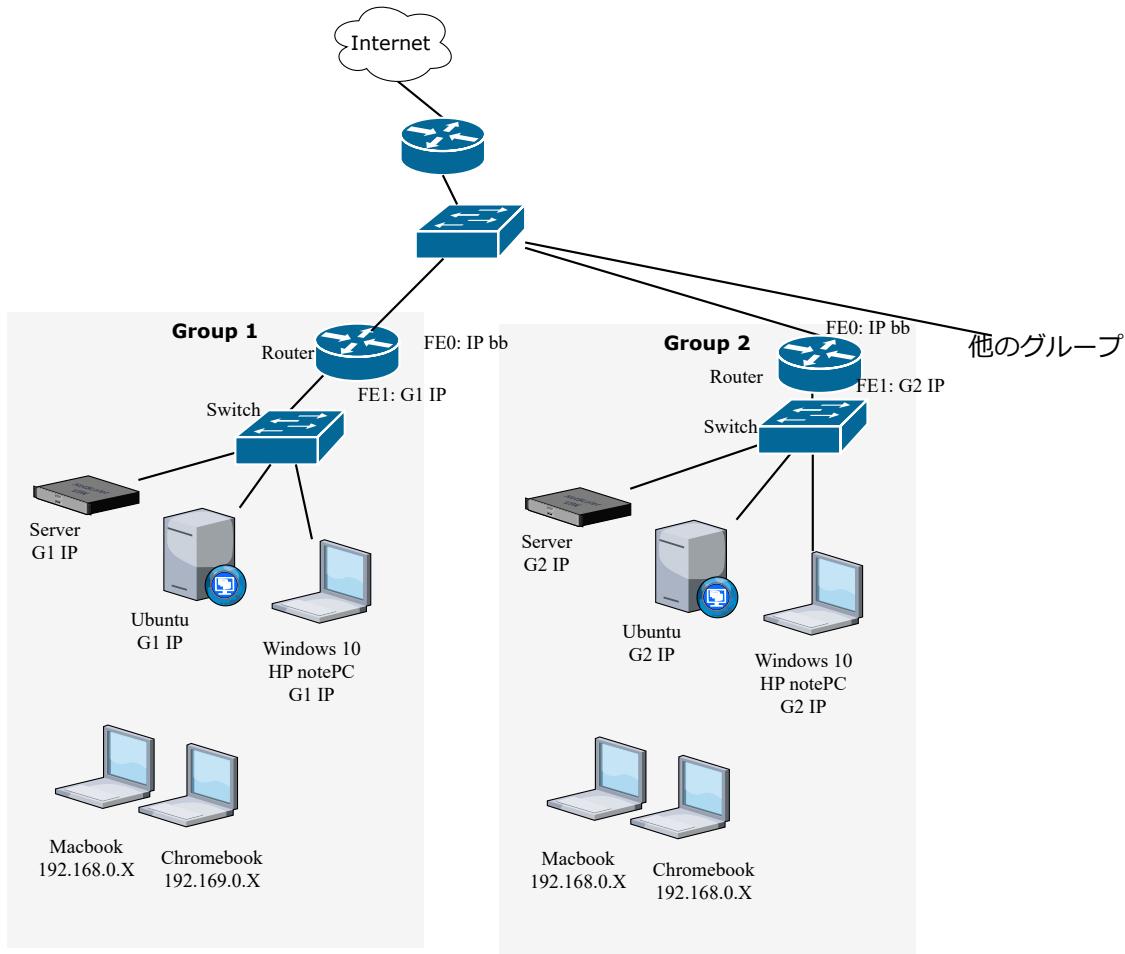


図 3.2 サブネット化後のネットワーク。ルータを各グループごとに配置し、FE0 (インターフェース fasterEthernet0) はバックボーン (外側=インターネット側)、FE1 はグループスイッチに接続する。各グループの IP (GX IP) は、指示されたものを使う。

3.4 必要となる知識

3.4.1 コンソール

ルータ・スイッチの設定は、コンソールを用いて行う。コンソールとは、キーボードやディスプレイなどの、人間との入力・出力をを行うデバイスであるが、ルータやスイッチなどの製品は、わずかなボタンやインジケータしか備えておらず、初期設定をすべてこれらのボタンで行うのは難しい。また、ネットワークの設定が整い、他の端末と通信が行える環境ができるまでは、通信を行うこともできない。このような場面で初期設定を行うのに用いられるものが、コンソールである。

ここでは、シリアルコンソールと呼ばれる、RS-232C シリアル通信 (無手順) を使用したコンソール（文字のみのキーボード入力・画面出力）を用いる。

パソコン用のシリアル端子に接続して、PC の端末（ターミナル）画面をコンソール（画面

表3.4 サブネット割り当ては、172.21.0.0/16 を1024分割したサブネットを下のように各班で用いる。

Group	Subnet
Group 1	分割したサブネットで12番目に小さいもの
Group 2	分割したサブネットで13番目に小さいもの
Group 3	分割したサブネットで14番目に小さいもの
Group 4	分割したサブネットで15番目に小さいもの
Group 5	分割したサブネットで16番目に小さいもの
Group 6	分割したサブネットで17番目に小さいもの
Group 7	分割したサブネットで18番目に小さいもの
Group 8	分割したサブネットで19番目に小さいもの
Group 9	分割したサブネットで20番目に小さいもの
Group 10	分割したサブネットで21番目に小さいもの
Group 11	分割したサブネットで22番目に小さいもの
Group 12	分割したサブネットで23番目に小さいもの

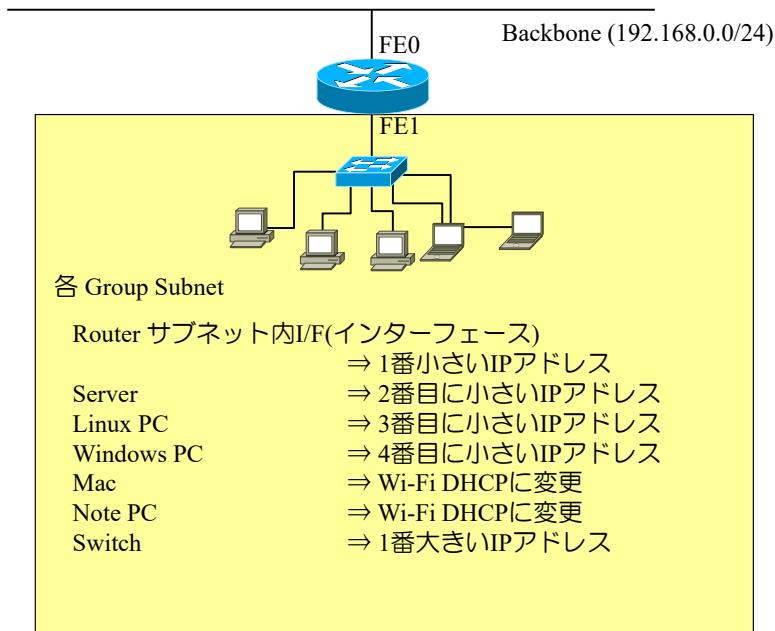


図3.3 サブネット内のIP割り当て

表示先)として用いる。

シリアル端子は、RS-232Cとも呼ばれるが、通信速度は、9600bps程度⁵と、現在の高速コンピュータに比較して遅いため、コンピュータに装備されていないことも多く、その場合は、USBシリアル、あるいはUSB-RS232Cなどの変換ケーブルを使って接続する。Windowsからは「COM」ポートとしてデバイスマネージャから確認でき、「COM1」、「COM2」など番号が付与されている（USBの場合、接続のたびに番号は可変のため、デバイスマネージャで確認する必要がある）。

本実験では、Ubuntu Linux Serverにシリアル端子があるため、これを用いる。

⁵ 8ビットコンピュータ等の遅いものでは、300bps, 2400bps, 早いものでも14400bps, 28800bps, 56Kbps程度

3.4.2 Cisco ルータ、スイッチへの接続

Ubuntu Server Linux のシリアル端子 (Dsub 9 ピン端子=RS-232C 端子) に、コンソールケーブルを接続する（水色のケーブル）。RJ-45 端子 (LAN 端子と同じ形状) の方は、ルータ (Cisco) の CONSOLE ポートに接続する。

次に端末で、root 権限になり、下記のコマンドで接続する。

```
cu コマンドをインストール  
(シリアル通信コマンド)  
  
# apt install cu  
  
シリアルポート（のデバイスファいる）に読み書き権限を付与  
ポート名は ttyS0 であり、ファイル名は /dev/ttyS0  
  
# chmod 777 /dev/ttyS0
```

以上でルータ設定のためのコンソール接続準備が終了。

■ コンソール接続

```
# cu -l ttyS0  
  
ルータの電源を ON  
(既に ON の場合は、Enter を 1-2 回か押してみて接続を確認)
```

■ コンソール接続終了

```
端末で Enter を 1-2 回押してから  
「~.」  
と入力する。  
  
(プロンプトを確認し、linux に戻っていることを確認)
```

3.4.3 Cisco IOS ルータの設定方法

ルータの設定は Cisco IOS 上で行うことができる。IOS (Internetworking Operating System) は、Cisco 社製のルータ・スイッチ製品に用いられている OS であり、現在、多くのネットワーク機器において、IOS に似た設定体系が、設定方法として採用されている。

ルータの初期状態では、パスワード等は設定されていないため、Enter でログインできる。

```
Would you like to enter initial configure dialog?  
→ このメッセージが表示された場合は、  
    初期設定メニュー（ダイアログ）に入るか否かを確認しており、
```

これからマニュアル(手動)で設定するため,
[no] を選択する(「no」と入力し Enter).

: 起動メッセージが表示される

:

Press RETURN to get started!

→ ここで、[Enter]を入力

Router#

(↑プロンプト：入力コマンド待ち状態)

(上記メッセージが表示されず、直接プロンプトが出る場合もある)

ここから、ルータのユーザ名、パスワードを設定し、cisco ユーザの削除、ホスト名の設定等を行うことができる。詳しくは巻末付録の IOS コマンド集や HELP を参照すること。以下は、ユーザとホスト名の管理および IP アドレスの設定例である。

また、それに先だって、Cisco IOS ではデフォルトとなっている DNS の逆引き設定、および、ログ表示とコマンド入力が混乱しないように、以下の 2 つの設定も有効にしておく。

```
Router>
(この状態はユーザモードで設定はできない)

管理者になる（イネーブル）

Router> enable
(TABで補完可。「en」と省略可)

Router#
```

Router# configure terminal ← (設定モードに入る)
(IOSでは、[Tab] 補完が可能。
例えば、「conf」まで入力し [TAB], 「t」だけ入力し [TAB] など。
「c」で [TAB] を入力すると、c で始まるコマンド群の説明が表示される。
候補項目が複数で無ければ、[TAB] 自体も省略可能で、
「conf term」「conf t」などと入力することも可能
途中まで入力し、「?」を入力することで、
その場面で入力できる項目の HELP を表示できる)

```
Router(config)#no ip domain lookup ← (DNSの逆引き設定の無効化)
(通常、コマンドを打ち間違えるとホスト名と認識され、
DNS検索が始まるので、これを無効にする)
```

コンソールでは、入力中にメッセージが表示されると、打っていた文字が流れてしまうのが、そのまま打ち込み続ければ正しく処理される。

しかし、分かりにくいので、メッセージ表示後、再度打ち込んでいた文字を再表示させる設定を行う。

```
Router(config)#line con 0
Router(config-line)#logging synchronous ← (ログ表示とコマンド入力画面の分離)
Router(config-line)#exit ← (console画面の設定の終了)
```

次にユーザ・パスワードを追加

```
Router(config)#username exp password 0 root00
```

次に管理者パスワードを追加

```
Router(config)#enable password 0 root00
```

ホスト名設定

```

Router(config)#hostname routerX (Xはグループ名)

router12(config)#exit (設定確認のため設定モードから抜ける)

router12#show running-config                                ← (現在の設定を確認)
Building configuration...

Current configuration : XXX bytes
:
:
(入力した内容が設定に反映されていることを確認)

```

■ パスワードの暗号化

このままでは、`show run` したときにパスワードが生で見えててしまうため、設定ファイルではハッシュ化して見えなくする。

```

router12#conf t
router12(config)#service password-encryption

show run して、再度設定内容を確認する
(パスワードがハッシュ化されていること)

```

■ IP 設定

FastEthernet0 (FE0)、FastEthernet1 (FE1) に IP アドレス設定
まず、FE0、FE1 にケーブルを接続する。
FE0 にはバックボーン (インターネットへ) のケーブル。
FE1 はグループのスイッチへ接続。

```

router12#conf t

router12(config)#interface fastethernet 0                  ← (I/F の設定)
router12(config-if)#ip address IP アドレス ネットマスク
router12(config-if)#no shutdown      ← インターフェースの有効化
router12(config-if)#exit

router12(config)#interface fastethernet 1                  ← (I/F の設定)
router12(config-if)#ip address IP アドレス ネットマスク
router12(config-if)#no shutdown      ← インターフェースの有効化
router12(config-if)#exit

```

■ 遠隔ログインの設定

CONSOLE だけでなく、TCP/IP 経由でリモートログインできるようにする。telnet と ssh が使えるがここでは、telnet を使う。

```
router12(config)#line vty 0 4
router12(config-line)#password 0 root00
router12(config-line)#login
router12(config-line)#exit
```

3.4.4 PC の IP アドレスを変更

Server, Linux, Windows の IP アドレスを、新しいサブネットのものに変更する。

- Server で cu 接続している場合は、一旦抜ける。
- Linux に telnet コマンドがなければ、apt install telnet にてインストール
- Windows の Powershell、コマンドプロンプトに telnet コマンドがなければ、「設定」→「アプリ」→「プログラムと機能（一番下）」→「Windows の機能の有効化または無効化」→「Telnet Client」のチェックを入れる。
- Windows の端末の telnet ではなく、ここでは Putty を使って接続することを勧める（上記の telnet コマンドはネットワーク管理で必要になることが多いので、入れておく）。Putty では、接続時に、「Connection type:」を「SSH」ではなく「Telnet」を選択する。
- IP アドレス・ネットマスクは新しいもの
- デフォルトゲートウェイはルータのグループ側 IP アドレス
- DNS はそのまま

IP 設定、遠隔ログイン設定が正しくされると、Linux, Mac, Windows の telnet コマンド (Windows は Putty) で、ログインできるようになる。ただし、PC の IP 設定の変更も終えている必要がある。

telnet 接続の例

```
# telnet ルータの IP アドレス
```

これが成功して以降は、cu コマンドでなく、telnet 接続して設定することが可能である。また、最大 5 人 (0-4) まで同時にルータに接続して設定することができる。

3.4.5 ルーティングテーブルの登録

Cisco IOS 上でルーティングテーブルを登録する例を示す。まず、現在のルーティングテーブルを確認する。

```
router12#show ip route
← (ルーティングテーブルを確認)

Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

172.21.0.0/24 is subnetted, 2 subnets
C       192.168.0.0 is directly connected, FastEthernet0
C       172.21.X.0 is directly connected, FastEthernet1
```

経路が“C”と書かれた、直接接続されたネットワークのみになっていることを確認する。これは、そのルータが直接接続されている（ルーティング設定をする必要のない）ネットワークである。

次に、静的ルーティングを手動で追加する。追加は、設定モードに入り、ip route コマンドで追加する。

■ルーティング設定

(他の) グループへの経路を、そのグループのネットワークアドレス、ネットマスク、次ホップルータの情報をルーティングテーブルに登録して設定する。

```
router12# configure terminal
← (設定モードに入る)
Enter configuration commands, one per line. End with CNTL/Z.
router12(config)#ip route ネットワークアドレス サブネットマスク 次ホップルータ
```

これを、自分以外の全てのグループについて設定する。

3.5 ルーティングの設定

Cisco ルータを用いて、サブネット分割されたネットワークを構築し、ルーティングの設定を行う。

コンソール用の PC とルータとをコンソールケーブルで接続し、ルータのユーザ名およびパスワード、ホスト名などの設定を行う。また、図 3.3 や図 3.4 を参照し、静的ルーティングが行うために必要な経路を考え、ルーティングテーブルを登録する。なお、グループは 1 から 12 まである。

3.5.1 コンピュータのネットワーク設定の変更

次に各コンピュータの IP アドレス, ネットマスク, デフォルトゲートウェイ, DNS サーバ等, ネットワーク設定の情報を変更する。

Ubuntu server では, /etc/netplan 以下の YAML ファイルを編集し, 変更後は, 再起動を行う。

なお, 実際にはミッションクリティカルなサーバ運用においては, netplan 編集後にダウンさせずに ifconfig で手動変更も行って, 止めずに運用を続ける場合も多いが, ここでは, 編集ミスなどによる不具合の可能性も考え, 念のため再起動を行い正常に起動するか確認する。

3.6 動作確認

経路の設定および IP アドレスの再設定が終わったら, ping コマンドおよび, traceroute (Windows は tracert) コマンドで実際に宛先ネットワークまで到達できるか確認する。ネットワーク内のコンピュータから, 以下の各ネットワーク内のコンピュータへの到達性を確認する。

- 他のグループのルータ, サーバ, Windows, Linux 等
- 実験室のメインサーバ 192.168.0.1
- インターネット www.yahoo.co.jp
- Mac, Chrome からのグループ内 PC に接続可能であること

上記すべての経路への ping, traceroute が成功すれば, 終了である。もし, ping が失敗する場合は, 原因を考察し設定を修正する必要がある。

Destination Unreachable エラーの場合 Destination Host Unreachable (宛先ホスト到達不可能) あるいは Destination Network Unreachable (宛先ネットワーク到達不可能) エラーが出る場合, そのエラーがどの IP アドレスから来ているかを確認する。その IP アドレスのルータにおいて, 宛先の IP アドレスへの経路がルーティングテーブルに無いことが原因であるので, 適切な経路を設定する。

もし, そのルータが自グループの管轄外である場合, 適切な管理者に必要な経路の設定を要請する。

エラーメッセージが出ない場合 ping のパケットがどこかで消失している場合は, エラーが出ない。この場合は, 明確な原因が分からない。より詳しい原因を知るために traceroute コマンドの使用を検討する。traceroute コマンドは, Linux, Windows, Cisco ルータにはデフォルトでインストールされている。ただし, Windows では tracert というコマンド名である。traceroute コマンドの詳細は付録を参照のこと。この traceroute コマンドを使用することで, ネットワークのどの部分に不具合があるかを推測することができる。

3.7 考慮すべき点

ルーティングは何のために必要であり、ルータはルーティングを行う際にパケットのどのような情報を見るのか、ルーティング先を決定するのにどのような情報を必要とするかを考える。

サブネット化を行い、L3 ネットワーク分割を行うことで、どのようなメリットがあるのか、またどのようなデメリットが生じるかを考え、適切なサブネット化は、どのように行えば良いかを考慮する。

3.8 スイッチを用いる目的

Ethernet による LAN を分割するには、物理的に異なるネットワーク機器およびケーブルを用いる方法がある。例えば、プロジェクトごとに異なる目的でネットワークを利用し、互いの通信を制限させたい場合は、A チーム用の LAN と B チーム用の LAN を、全く異なるスイッチによるネットワークを構成し、A チームの端末は A チーム用のスイッチに A チームのケーブルを用いて接続し、B チームは、B チーム用のスイッチに B チームのケーブルを用いて接続することで、それぞれのチームの端末が互いにチーム内では通信を行い、チーム外との通信は行えなくなる。2つの異なる LAN を接続するためには、Layer 3 の機器であるルータを用いる。

Layer 2において、物理的な分割でなく Layer 2 のプロトコルにおいて分割する方法が、Virtual LAN (VLAN) である。VLAN は、主にスイッチにおいて用いられている技術で、1つのスイッチに複数の異なる LAN の通信環境を実現したり、1本のケーブル上に複数の異なる LAN のパケットを送受信する技術である。本来1台のスイッチやネットワークケーブルは、1つの LAN の通信に用いられるが、VLAN を用いることで、複数の異なる LAN で、機器やケーブルを共用することができる。

現在は、中規模以上のネットワークでは VLAN が広く使用されており、ルータやサーバ用コンピュータなどの端末も、VLAN の使用が可能なものが多い。VLAN を用いることで、管理する LAN の数が増加しても、機器の費用、設置場所、電気等のランニングコスト、ネットワークケーブルの敷設本数、管理の手間や保守のコストなどを抑えることが可能になる。

VLAN が行っていることは、ブロードキャストドメインの分割である。ブロードキャストドメインとは、イーサネットのブロードキャストフレームが到達する範囲である。通常、リピータやブリッジ、スイッチはブロードキャストフレームを全てのポートに転送し、LAN 内の全ての機器に送信する。このようにすることで、ARP などの宛先を探索するフレームを確実に全ホストに転送する。ブロードキャストの届かない端末は、他の端末から探索されなくなるため、必然的に通信が行えなくなり、隔離される。

VLAN に対応したスイッチ・ルータは、各ポートで受信したフレームがどの LAN に所属するのかを判断し、その LAN に所属するポートのみに対して転送を行う。異なる LAN に所属するポートには、ブロードキャストも含め、いかなる別の LAN のフレームも送信しない。このようにすることで、1つのスイッチ内で、LAN を分割する。これはあたかも複数のスイッチを設置した環境と同じような環境である。

図 3.7 は、ポート VLAN を用いて LAN A, LAN B, LAN C の3つの LAN を実現する図である。VLAN を用いない場合は左のように、3つのスイッチを用意する必要があるが、VLAN を用いることで1台のスイッチで Layer 2 のレベルでネットワークを分割することができる。

図 3.5, 図 3.6 は、3つの LAN を異なる場所で接続できるようする図である。図 3.5 の VLAN を用いない場合は、それぞれ3つのスイッチを準備し、さらにそれらの間を3本のネットワークケーブルで接続する必要があるが、図 3.6 のように、ポート VLAN とタグ VLAN を用いることで、それらをまとめることができる。

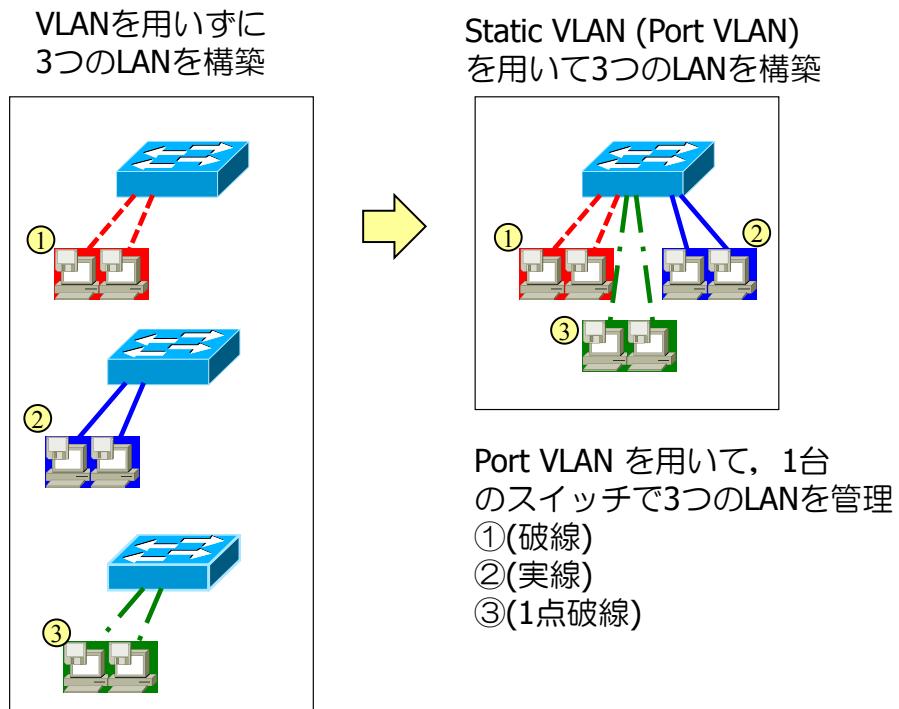


図 3.4 静的 VLAN

3.9 VLAN の種類

VLAN では、受信したフレームがどの LAN に所属するかを判断する情報として、以下のようなものに分けられる。

静的 VLAN スイッチの各ポートが、どの LAN に所属するかを管理者が手動で設定する VLAN。スイッチの物理ポート毎に所属する LAN を決めるため、しばしば、**ポートベース VLAN**、**ポート VLAN**とも呼ばれる。管理者が一度設定すると設定を変更するまでそのポートを送受信するフレームの所属する LAN は固定される。このため静的 VLAN (Static VLAN) と呼ばれる。また、末端の PC のアクセスには後述のタグ VLAN が使わないので、**アクセス VLAN**とも呼ばれる。

動的 VLAN スイッチで受信されたフレームの Mac アドレスや IP アドレス（サブネット）、プロトコル（IP か Appletalk など）で、所属する LAN を決める方法。フレームに含まれる情報のみから、所属する LAN を決めるので、受信するポートには依存することなく LAN が決める。このため、動的 VLAN と呼ばれる。VLAN が意図せずに変わることもあるので、セキュリティなど考慮すべきことは多い。

タグ VLAN スイッチで受信されたフレームにあらかじめ所属する LAN の情報が含まれており、この情報を元に所属する LAN を判断する方法。この情報を、VLAN ID と呼ぶ。VLAN ID の含め方によ

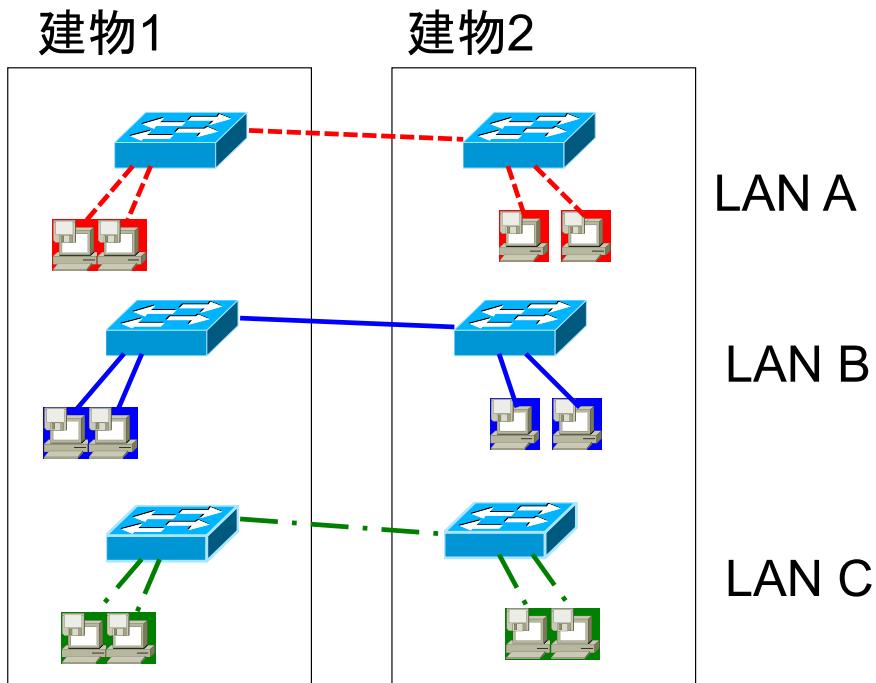


図 3.5 VLAN を用いない場合

り、Cisco ISL VLAN と IEEE 802.1q VLAN に分けられるが、現在では、ほとんど 802.1q が用いられる。略して、dot1q、ドットイチキューなども呼ばれる。スイッチでは、このタグ VLAN (tagged VLAN) のことを、VLAN トランкиング (trunking)、トランク VLAN などと呼ぶ。1つのポート（ケーブル）に複数の VLAN を収容することからこのように呼ばれる。タグ VLAN のポートのことを、トランクポート (trunk port) と呼ぶ。

3.10 実験内容 (2)

本章において必要となる作業は、下記の通りである。

- 現在のスイッチの状況を確認
 - 設定情報
 - ポート（インターフェース）の状態
 - VLAN の状態
- スイッチの初期設定

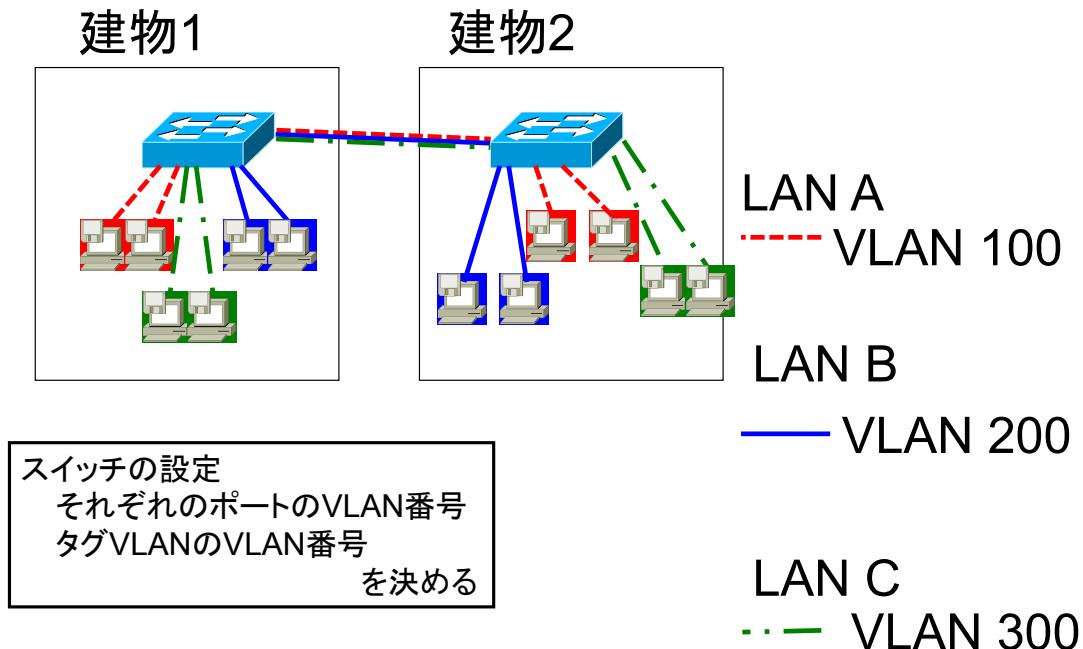


図3.6 静的 VLAN とタグ VLAN を用いる場合

- ログ表示と設定中の文字とが混在しないようにする
- DNS ルックアップを無効にする
- スイッチに、自グループ用の VLAN と、バックボーン用の VLAN を定義する。
 - 自グループの VLAN ID は、グループ番号 + 10.
 - バックボーン用の VLAN ID は、999.
- ポート VLAN (静的 VLAN, アクセス VLAN) を設定
 - 1~16 番ポートは自グループの PC 用のアクセス VLAN 用にポート VLAN (静的 VLAN) を設定.
 - 17~22 番ポートはバックボーン用の VLAN をポート VLAN で設定.
- リモートログインの設定
 - 自グループ VLAN に IP を設定する
- タグ VLAN (トランク VLAN) を設定 (実験 4C のみ, 実験 3i では必要ない)
 - 23, 24 番ポートには、自グループの VLAN とバックボーンの VLAN の両方を, IEEE 802.1q タグ VLAN で接続できるよう設定する (実験 4C のみ, 実験 3i では必要ない).
- ケーブル接続変更

- バックボーンからのケーブルをスイッチのバックボーン用ポートに接続
 - ルータのバックボーン側インターフェースとスイッチのバックボーン用ポートを接続
 - ルータの自グループ LAN 側インターフェースとスイッチの自グループ用ポートを接続
 - 有線 LAN 接続の PC, サーバを適切に接続
- ポート接続設定
 - 23 番ポートは接続できないよう, shutdown する (実験 4C のみ. 実験 3i では必要ない).
 - 1~16 番ポートは, スパニングツリーを無効にする (実験 4C のみ. 実験 3i では必要ない).
 - 設定後のスイッチの状況を確認
 - 設定情報
 - ポート (インターフェース) の状態
 - VLAN の状態
 - MAC アドレステーブルの状態

3.11 設定に必要な情報

- VLAN 番号
 - 自グループ用 VLAN : グループ番号 + 10
 - バックボーン VLAN : 999
- スイッチの設定
 - ポート 1~16 : 自グループ用 ポート VLAN
 - ポート 17~22 : バックボーン用 ポート VLAN
 - ポート 23~24 : 自グループ・バックボーンのタグ VLAN (3i では必要無し)
- スイッチ遠隔用 IP アドレス
 - VLAN は自グループに設定
 - IP アドレス : 自グループサブネットで一番大きい IP アドレス (ブロードキャストを使わないこと)

3.12 作業内容

3.12.1 スイッチの設定

スイッチの現状を確認する.

(1) 設定確認

```

Switch>enable (または en)
Password:
Switch#show running-config (省略形 show run, sh run,)
↑ 現在の設定の確認。スイッチ設定が表示される
"??" キーでのコマンド説明や [TAB] 補完も活用できる。

Switch#show interface status (ポート確認)
Port      Name          Status       Vlan      Duplex   Speed Type
Fa0/1                notconnect   1         auto     auto
10/100BaseTX

(ポート Fa0/1 は VLAN 1(デフォルト VLAN) で未接続)
:
:
Fa0/6                connected    1         a-full   a-100
10/100BaseTX

ポート Fa0/1 は VLAN 1(デフォルト VLAN) で自動認識 (auto negotiation=オートネゴシエーション) の結果 全二重 (full duplex), 100M で接続されている

### VLAN 状態
kut192-168-0-176#show vlan

VLAN Name          Status      Ports
--- -----
1    default        active      Fa0/19, Fa0/20, Fa0/21, Fa0/22, Fa0/23
, Fa0/24
12   group12       active      Fa0/1, Fa0/2, Fa0/3, Fa0/4, Fa0/5,
Fa0/6, Fa0/7, Fa0/8, Fa0/9, Fa0/10, Fa0/11, Fa0/12, Fa0/13
                               Fa0/14, Fa0/15, Fa0/16, Fa0/17, Fa0/18
1002 fddi-default  act/unsup
1003 token-ring-default  act/unsup
1004 fddinet-default  act/unsup
1005 trnet-default   act/unsup

↑ どの VLAN がポートに設定されているか分かる

```

(2) 初期設定

パスワード設定、リモートログイン (telnet)、ログ出力制御、DNS ルックアップなどの設定をする。

```
...> enable

...# configure terminal

### スイッチホスト名の設定
... (config)# hostname switch12

### ユーザ名 exp、パスワードの設定
switch12(config)# username exp password 0 root00

### 管理者パスワード
switch12(config)#enable password 0 root00

### パスワードの暗号化
switch12(config)#service password-encryption

### DNS ルックアップ無効化
switch12(config)#no ip domain-lookup

### コンソール設定
switch12(config)#line con 0

### コンソールでログで文字が流れないように
switch12(config-line)#logging synchronous

switch12(config-line)#exit
```

(3) VLAN 定義

```
switch12#configure terminal

### 自グループ VLAN を定義

switch12(config)#vlan XXX
    ↑ vlan 作成。XXX はグループ番号+10
switch12(config-vlan)#name groupXXX
    ↑ 分かりやすく名前を付ける (XXX グループ番号)
switch12(config-vlan)#exit
```

```
### バックボーン VLAN (YYY の値は本文参照) を定義

switch12(config)#vlan YYY
switch12(config-vlan)#name backbone
switch12(config-vlan)#exit

### 次に interface コマンドでポートのアクセス VLAN 設定

### interface で range を使い、多くのポートを一度に設定

switch12(config)#int range fa0/1-16
  (range 記述 fa0/1-16 をまとめて指定)

switch12(config-if-range)#switchport mode access
  (ポート VLAN 用にポートに設定)

switch12(config-if-range)#switchport access vlan XXX
  (VLAN XXX (XXX は自グループ用 VLAN) に設定)

switch12(config-if-range)#exit

### 次にバックボーン用ポート VLAN の設定。

switch12(config)#int range fa0/17-22

switch12(config-if-range)#switchport mode access

switch12(config-if-range)#switchport access vlan YYY
  (VLAN YYY バックボーン用 VLAN)

switch12(config-if-range)#exit
```

(4) リモートログイン設定

VLAN 設定が終わり遠隔ログインが設定可能になったので設定する。

```
### スイッチに遠隔ログイン用 IP アドレスを設定

switch12(config)#interface vlanXX (XX は自グループ VLAN 番号)
switch12(config-if)#ip address X.Y.Z.W 255.255.255.0

(255.255.255.0 はネットマスクであり異なる場合は
```

適切なものにすること)

```
switch12(config-if)#exit

### telnet 許可設定

switch12(config)line vty 0 4
# 5人まで同時ログイン

switch12(config-line)password 0 root00
# 遠隔用パスワード

switch12(config-line)login
# ログイン許可

switch12(config-line)#logging synchronous
# コンソールと同様、ログで文字が流れないように

switch12(config-line)#exit
```

サーバ、Linux Desktop、Windows Putty などからスイッチに telnet 接続できるか確認

(5) 接続

図 3.7 を参考に物理接続を変更する。

- 自グループの機器（ルータを含む）は、自グループのスイッチの自グループ VLAN へ接続
- バックボーンのケーブル、ルータバックボーン側はバックボーン VLAN へ接続

(6) スイッチの設定確認

下記各コマンドで、設定を確認する

```
show running-config

show vlan

show interface status

show mac address-table
```

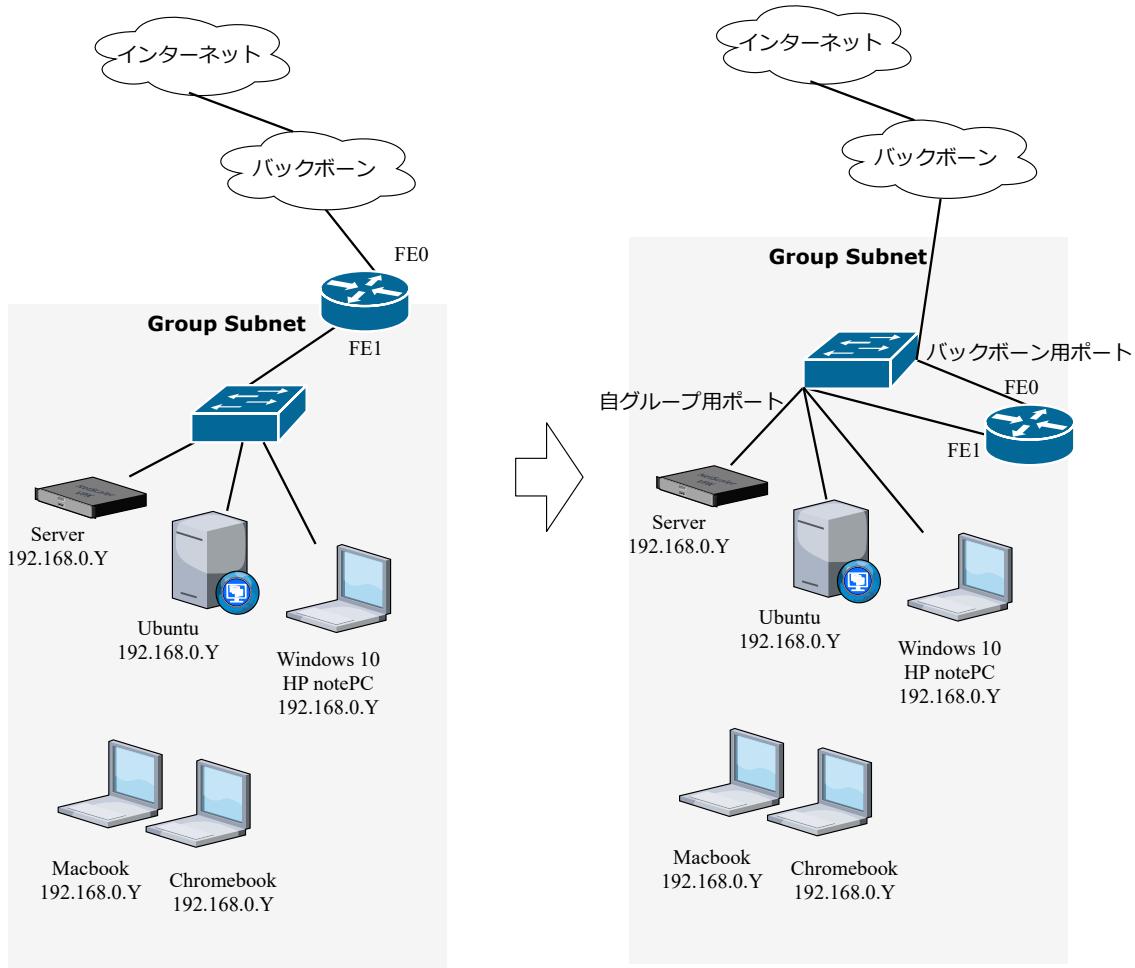


図3.7 接続図

3.12.2 トランク VLAN (4Cのみ)

最後に他グループのスイッチ間を接続するトランクポートを設定

自グループと他グループの VLAN をタグ VLAN として設定する

(行か否かは各グループで判断)

```
switch12(config)#int range fa0/23-24
```

```
switch12(config-if-range)#switchport mode trunk  
(VLAN トランクポートに設定)
```

```
switch12(config-if-range)#switchport trunk allowed vlan XXX,YYY  
(自グループ用 VLAN XXX とバックボーン用 VLAN XXX を設定)
```

```
switch12(config-if-range)#exit
```

23 のインターフェースを安全のためシャットダウンさせる。
(24 は有効にする)

```
switch12(config)#int range fa0/23
switch12(config-if-range)#shutdown
switch12(config)#int range fa0/24
switch12(config-if-range)#no shutdown
```

3.12.3 Spanning Tree 無効化 (4C のみ)

Cisco 社の Web 「Catalyst スイッチのポートで STP を無効にする」を参考に設定する。
ポートの抜き差しを行い、差し直したときに、オレンジ色の STP 学習中の状態が 50 秒続かずに、すぐにグリーンのリンクアップ状態になることを確認する。
また、ループを作ると、サブネット内で通信障害起こることも確認する。

3.13 動作確認

- 自グループ用ポート同士で通信可能
- 自グループの Windows PC をバックボーン用ポートに接続すると通信不可能
- 以上を相互に ping や traceroute で確認する。
- シャットダウンポートには、何を接続してもリンクアップしないことを確認 (4C).

3.14 考慮すべき点

- VLAN を設定することで、どのようなメリットがあるか。
- VLAN を用いても効果が無い場合や、VLAN では機能が不足している点

第 4 章

DNS・メールサービス

4.1 目的

インターネットを使った様々なアプリケーションでは、通信の宛先・送信元を識別するのにホスト名を用いることが多い。ホスト名とは、通信を行う端末に対し何らかの意味を持たせた文字列を用いて名前を付けたものである。一方、インターネットの通信プロトコル IP では、IP アドレスを用いて端末の識別を行う。そこで、アプリケーションが通信を行うためにホスト名を指定した後、IP パケットでデータを伝送するために、そのホスト名に対応する IP アドレスを求める必要がある。これを名前解決と呼ぶ。

DNS (Domain Name System) は、インターネットの標準的な名前解決システムであり、ホスト名を用いた通信を行うために DNS サービスを構築する必要がある。

電子メールは、DNS を用いる代表的なアプリケーションの一つであり、DNS と同じく 1980 年代より使われ、現在でもインターネットでの主要なアプリケーションとして広く利用されている。電子メールで用いられる電子メールアドレスでは、メールドメインと呼ばれる部分に DNS で指定される名前が用いられる。

4.2 DNS とは

ネットワーク層プロトコルである IP では、ホストの識別をするのに IP アドレスを用いる。IP アドレスは 32 ビットで表される整数であり、表記する際は、192.168.0.1 というように、1 オクテット (8 ビット = 1 バイト) ずつ 10 進法で、ピリオドで区切って表記される¹。これを、英語では dotted-quad decimal notation (ドット付きの 4 つの 10 進表記) と呼ぶ。

一方で、ホストの管理は IP アドレスとは独立して組織あるいは運用者単位で行っている。このため、ホストの識別は、会社や企業、部署の構成や、サーバで提供するサービスに対応して行えることが望ましい。このように、サービスの種類や、組織の構成などの情報をもとに決めるホストの識別情報として、ホスト名を用いる。

実際の IP 通信は、IP アドレスを識別情報として通信を行うので、ホスト名と IP アドレスの対応付けを行い、適宜変換する仕組みが、名前解決 (Hostname Resolver) である。名前解決で最も単純なものは、**Hosts ファイル** と呼ばれるもので、IP アドレスとホスト名の対応を、テキストファイルで格納しておき、OS はその情報を参照して、両者の変換を行う。

Hosts では、大規模なネットワークの名前管理は困難であるため、インターネットのような大規模なネットワークにおいても実用的に管理・運用できる名前解決の仕組みとして、DNS (Domain Name System) が使われている。DNS により、電子メールや WWW などのサービスを利用する際に、利用者はホスト名

¹ 現在のバージョン 4 は 32 ビットのアドレス空間を持つが、次のバージョン 6 (IPv6) では、128 ビットに拡大される。

の情報のみを用いて anyone@ugs.kochi-tech.ac.jp や http://www.kochi-tech.ac.jp/ などのアプリケーション層アドレスを用いることが可能となっている。

4.3 実験内容 (1)

DNS サービスを構築する。ここでは、DNS サービスとは下記に挙げる 2 つ、他ドメイン向けに DNS で自サーバ名の名前提供を行うサービスと、自ドメインのクライアント PC 向けの他ドメインのサーバ名解決を行うサービスでを構築する。

- サーバコンピュータでの DNS サービスの構築
 - 自ドメインのネームサーバ（権威ネームサーバ、コンテンツサーバ）の構築
 - キャッシュサーバ（フルサービスリゾルバ）の構築
- クライアントから DNS サーバ（キャッシュサーバ）への問い合わせ設定（クライアントのリゾルバ設定）

である。

4.4 必要となる情報

DNS 設定の方針

本実験では次のような設定の方針に基づいて、DNS を設定する。

- (1) Ubuntu server でネームサーバを構築する。ソフトウェアは BIND9 を用いる。

- 各グループごとにドメインを構成し、それぞれのグループのドメインのネームサービスを server にて行う。
- ドメイン名は、gX.exp.info.kochi-tech.ac.jp とする。X はグループ番号であり、グループ 13 であれば、g13.exp.info.kochi-tech.ac.jp である。
- 各グループのホスト名から IP アドレスを引けるようにする（正引き）
- 逆引き設定は行わない（逆引き=内部の IP アドレスからホスト名を引けること）。
- ドメインの Web サーバは、www.domain 名でアクセスできるようにし、実際の Web サーバは、server となるようにする。

- (2) server に DNS キャッシュサーバ²の機能を構築する。

² フルサービスリゾルバともいうが、ISC BIND での正確な用語を使えば、各クライアント PC からの再帰問い合わせ (recursive query) 要求を受け付け、自らは root server をはじめ、各ドメインのサーバと直接の問い合わせ (非再帰的=non recursive query) を繰り返す (iterative) リゾルバである。

- `server` 上の DNS キャッシュサーバにて、各グループ内のコンピュータは `server` を DNS サーバ（再帰リゾルバ）として名前解決できるようにする。
- 各クライアントのドメイン名、ネームサーバアドレスを、各クライアントが `server` の DNS サーバを検索するよう設定する。

また、各グループ内の端末の IP アドレス、およびホスト名は、表 4.1 となる。

表 4.1 各グループ内の端末の IP アドレスとホスト名（ただし、Y はグループ番号 X に 10 を加えた数）

計算機	ホスト名	IP アドレス
サーバ	<code>server</code>	server の IP アドレス
Ubuntu Desktop	<code>linux</code>	Ubuntu Desktop の IP アドレス
Windows 10	<code>win</code>	Windows 10 の IP アドレス
Macbook	<code>mac</code>	DHCP のためここでは設定しない
ChromeBook	<code>note</code>	DHCP のためここでは設定しない

その他に追加するレコードとして、ネームサーバ（= `server` 自身）の NS レコードも忘れないに設定すること。

また、今後、Web サーバ、メールサーバをこのドメインに追加設置する際は、CNAME レコード、MX レコードをそれぞれ追加する必要がある。

設定ファイルの場所

設定ファイルは以下の場所に置くこと。

`named.conf` /etc/bind

ゾーンファイル等のデータを置くディレクトリ /etc/bind

ゾーンファイル名 /etc/bind/gX.zone (X はグループ番号)

ワーキングディレクトリ /var/cache/bind

4.5 必要となる知識

先に述べたように、普段我々は計算機を識別するためにホスト名を用いている。しかし TCP/IP では計算機の識別に IP アドレスを指定しなければならぬ、実際の運用にあたって、この問題を解決しなければならない。ホスト名から IP アドレスを求めるのことを名前解決（Name Resolution）といい、以下に示す方法がある。

hosts ファイル

最も簡単な名前解決の方法は、各ホストが、ホスト名と IP アドレスとの対応表を持っておき、それを参照するというものである。この対応を記したファイルは hosts ファイルと呼ばれ、以下に示すような書式となっている。hosts ファイルは、通常 /etc の下に置かれる。ただし、Microsoft Windows では、C:\Windows\system32\drivers\etc\hosts である。

```
#■/etc/hosts ファイルの例

127.0.0.1      localhost
192.168.0.1    mainserver.info.kochi-tech.ac.jp  mainserver

### 上記の例では、ホスト名「mainserver.info.kochi-tech.ac.jp」
### または、「mainserver」と指定したら、192.168.0.1
### と解決することを示す。
```

この方法は単純であり、数が限られている組織内であれば、DNS サーバを構築することなく、全てのホスト名とその IP アドレスを上記のテキストファイルに記述するだけで、簡単に実現できるホスト名解決方式である。しかし、大規模なネットワークでの名前解決や、インターネット規模での名前解決には適さない。また、動的なネットワーク構成の変更には対応しづらい、すべてのホストが同じファイルを保持しなければならない、などという欠点がある。比較的ホスト数が少なく、内に閉じた LAN であればこれでも構わないが、世界中のホストと通信するインターネットでは、ホスト名による名前解決は不可能である。

LAN 内のディレクトリサービス

組織や LAN の中であれば、ディレクトリサービスを使ってホスト名情報（hosts ファイルに相当する情報）を、クライアントに提供する方法もある。LDAP (Lightweight Directory Access Protocol), NIS (Network Information Service, yp:yellow page(電話帳) とも)、マイクロソフトの NT ドメインを利用したホスト名解決は、現在でも使われる場面がある。

複数の名前解決手段の併用

hosts ファイル、LDAP や NIS、次に説明する DNS など、複数の名前解決サービス（name service）手段を併用することができるが、この場合は問い合わせに優先順位が付けられ、先に問い合わせたもので答えが見つかったものから適用される。

この優先順位は、ネームサービススイッチファイル /etc/nsswitch.conf に書いてある。

例えば、Ubuntu 20.04 では、

```
...
hosts:          files mdns4_minimal [NOTFOUND=return] dns
...
```

と書いてあり、files (hosts ファイル) が最優先、次に mDNS、それで見つからない場合は、DNS を調べるよう設定されている。

もし、DNS の設定が反映されていないようであれば、hosts ファイルなどが優先されている場合があるので、注意する。

DNS

インターネットにおける名前解決の方法として用いられているのが DNS である。前説の hosts ファイルでは、インターネット全体の名前解決は不可能であるので、分散管理ができる DNS が用いられている。

DNS は、分散型データベースであり、クライアントからの名前解決要求（ポート番号 53 への接続）に対して回答を行なう機能を有する。そして、同一ドメイン内の各ホスト（クライアント）にこのようなデータベース索引サービスを提供するサーバである。DNS では、サーバでホスト名のデータベースが一元管理されるので、hosts ファイルの欠点であった動的なネットワーク構成の変化にも容易に対応できる。

DNS では、名前を管理する名前空間を図 4.1 のような階層的なツリー構造で構成しており、ホスト名は、図 4.1 に示したツリーのラベルを最下層から順に並べ、“.” で区切って表記する。例えば www.kochi-tech.ac.jp や ugs.kochi-tech.ac.jp などである。そして、これらの場合 kochi-tech.ac.jp をドメイン名という。すなわちドメインとは、名前空間の分割の単位であり、階層的なツリー構造の名前空間のあるノードから先を構成するサブツリーである。このドメインを単位として、名前空間での名前の命名・変更の権利を、世界中の組織に分散させる。あるドメインが自分のドメインの一部のサブツリーを別の組織の管理に任せることを、ドメインの委譲と呼び、委譲されたドメインはサブドメインと呼ぶ。世界で唯一の名前であるドメインが、唯一の組織にサブドメインを委譲する形で名前空間を分割する。このような名前の命名方法を用いれば、世界中で計算機のホスト名を重複することなく、命名することができる。なお、ルート階層のラベルは空ラベル（“”）であり、実際のホスト名では見えなくなっている。ルート階層での命名権は IANA にあり、IANA がルートのサブドメインであるトップレベルドメイン (TLD) をサブドメインとして定義し、それぞれのサブドメインを委譲する。

図 4.2 は DNS による名前解決処理の様子を示している。DNS では階層的な名前空間を用いているため、ホスト名 www.kochi-tech.ac.jp のアドレスを取得する際の処理は次のようになる。

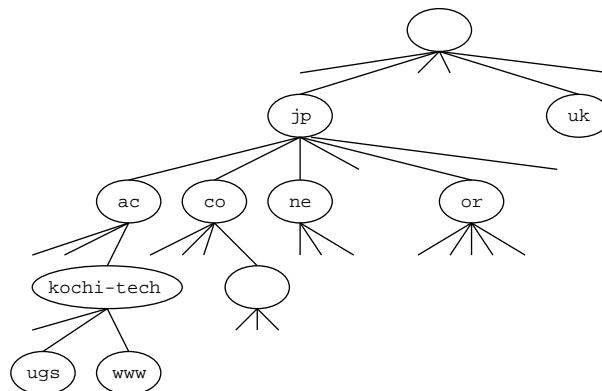


図 4.1 階層型ドメイン空間

- (1) クライアントがローカルネームサーバに www.kochi-tech.ac.jp のアドレスを問い合わせ。
- (2) ルートネームサーバにアドレスの問い合わせ。ルートネームサーバは jp ネームサーバのアドレスを教える。
- (3) jp ネームサーバにアドレスの問い合わせ。jp ネームサーバは ac.jp ネームサーバのアドレスを教える。

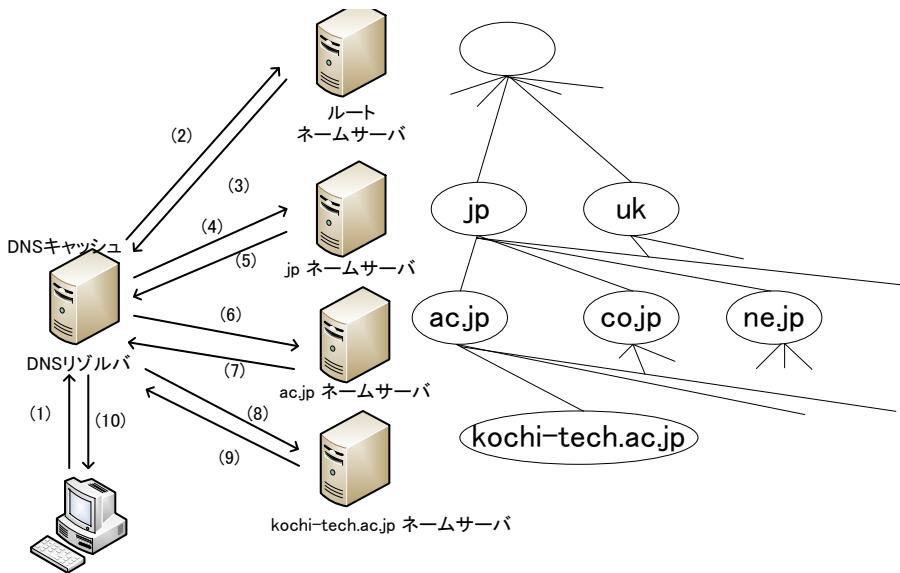


図 4.2 DNS による名前解決

- (4) `ac.jp` ネームサーバにアドレスの問い合わせ. `ac.jp` ネームサーバは `kochi-tech.ac.jp` ネームサーバのアドレスを教える.
- (5) `kochi-tech.ac.jp` ネームサーバにアドレスの問い合わせ. `kochi-tech.ac.jp` ネームサーバは `www.kochi-tech.ac.jp` のアドレスを教える.
- (6) ローカルネームサーバがクライアントにアドレスを応答する.

DNS サーバは世界中のコンピュータのホスト名と IP アドレスを知っているわけではない。名前空間の第一階層にある `jp` ネームサーバは、第二階層である `ac.jp`, `co.jp`, `ne.jp` などドメインのネームサーバのホスト名と IP アドレスを知っている。同様に、`kochi-tech.ac.jp` ネームサーバは、`ugs.kochi-tech.ac.jp`, `www.kochi-tech.ac.jp` のホスト名と IP アドレスを知っているのである。このようにおのののネームサーバは、ゾーンとよばれるドメイン空間の一部についての情報（データベース）を完全に保持している。この場合、「ネームサーバはこのゾーンに対する権威（authority）をもっている」という。

図 4.1 のドメイン名空間の場合、ホスト名から IP アドレスを検索することはできるが、IP アドレスからホスト名を検索することができない。つまり、このドメイン名空間ではドメイン名を索引としてしているため、ドメイン名から IP アドレスを検索することは容易であるが、IP アドレスからドメイン名を検索するためにはすべてのデータを調べなければならないからである。そこで DNS では、図 4.3 に示すようなアドレスをラベルとする `in-addr.arpa` というドメインを作りこの問題に対応している。

図 4.3 に示すのは、`www.kochi-tech.ac.jp` の IP アドレス `210.163.144.42` の `in-addr.arpa` ドメイン名である。ドメイン名の読み方の規則に従うと、このホストの `in-addr.arpa` サブドメイン名は `42.144.163.210.in-addr.arpa` となる。DNS はこのような IP アドレスを索引とするドメイン名空間により、ホスト名から IP アドレスを検索するのと同じ仕組みで IP アドレスからホスト名を検索することができる。

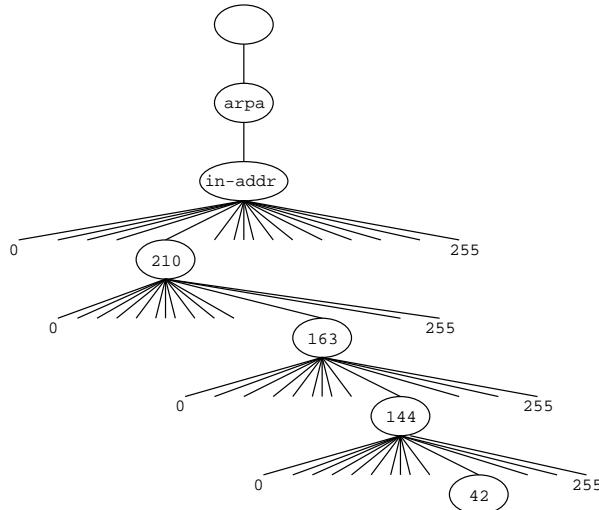


図 4.3 in-addr.arpa ドメイン名空間

以上から DNS は、ゾーン内のホストの、ホスト名から IP アドレスを検索するデータベース、IP アドレスからホスト名を検索するデータベース、ルートネームサーバの情報を保持する必要がある。また安全性的観点から、DNS は通常、プライマリマスターとスレーブと呼ばれる 2 つのサーバで運用する。スレーブはプライマリマスターのデータベースファイルを定期的にコピー、バックアップし、プライマリマスターが停止している時などは、プライマリの DNS サーバとして機能する。

BIND

DNS サービスのための実装プログラムとしては BIND を用いる。BIND は、California 大学 Berkeley 校 (UCB) で開発・実装された DNS サービスのためのプログラムで、4.2 BSD UNIX 上で最初に実装された。現在は ISC(Internet Systems Consortium) によりメンテナンスが行われており、同サイトで最新版が活発にリリースされている。

UNIX はもちろん、その他のプラットフォーム上で利用できる DNS サーバは、ほとんどがこの BIND のプログラムをもとに開発されている。また、BIND が DNS プロトコルのリファレンス実装³となっている。

³ 同プロトコルのサービスプログラムとして中心的な役割を担い、DNS プログラムを開発する人は BIND を参考にすべきである、という位置付け

4.6 DNS サービスの構築の手順

4.6.1 BIND のインストール

Server にパッケージシステムを用いて BIND をインストールする手順を説明する。

パッケージシステムとは、Linux のディストリビューションや FreeBSD, NetBSD など、OS ごとに様々なオープンソースなどの追加ソフトウェアについて、ダウンロード・コンパイル・インストールなどの処理をパッケージ化して、簡単なコマンドのみでインストールできるように準備されたものである。

Debian 系 (Ubuntu) では apt コマンド、Redhat 系 (CentOS, SuSE 等) では yum コマンドを用いる。

```
$ sudo su
(↑管理者権限になる)

# apt update
(↑パッケージサーバに最新情報を照会)
# apt install bind9
(↑BIND 9 をインストール)
```

これで、インストールは終了である。DNS の設定ファイルなどは、Ubuntu のパッケージの場合、/etc/bind に格納される⁴。

4.6.2 BIND の全体設定

まず、BIND の全体設定として、設定ファイル named.conf の設定を行う。

```
# vi /etc/bind/named.conf
```

このファイルは、BIND 全体の動作を設定する。具体的には、どのドメインを管理するのか、そのドメインに関する情報はどのディレクトリに格納するか、マスターかスレーブか等である。

Ubuntu のパッケージの場合、named.conf の内容は以下のようなものとなっているはずである。

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

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

⁴ BIND をソースコードからインストールした場合は、/etc/named である。

これは、具体的な設定はここでは行っておらず、include の右側で指定される別ファイルの内容を取り込み、設定に反映されることを示している。すなわち、named.conf で直接設定を行うのではなく、指定されている各ファイル内での設定を推奨していることを示している。以下ではそれぞれのファイルでの設定内容について示す。

各ファイルでの設定を行う前に、以下のようにしてすでに配置されているファイルを別名で待避させておく。

```
# cd /etc/bind
# cp -p named.conf.options named.conf.options.org
```

cp コマンドはコピーを行う（本テキスト末尾の付録参照）が、-p オプションを付けることで、所有者、パーミッション、タイムスタンプ（作成時刻、変更時刻、アクセス時刻）などを保存（preserve）してコピーする。付けない場合は、所有者は実行者、タイムスタンプは実行時、パーミッションもデフォルトのものに変更されてコピーされるため、ファイルバックアップには適さない。

named.conf.options 内には BIND の動作に関するオプションを記述する。ワーキングディレクトリの指定はここで行う。named.conf.options を新規に作成し、次のように記述する。

```
options {
    directory    "/var/cache/bind";
};

// BIND が動作する際のワーキングディレクトリ
```

named.conf.local 内で自分が管理するドメインを zone で指定する。named.conf.local を新規に作成し、次を参考に記述する。以下の例は、gX.info.kochi-tech.ac.jp ドメインの情報を、ゾーンファイル /etc/bind/gX.zone に記述する場合である。

```
// ↓下記記述中の X はグループ番号にすること
zone "gX.exp.info.kochi-tech.ac.jp" {
    type master;
    file  "/etc/bind/gX.zone";
};

// gX... ドメインのプライマリサーバであり、その情報はゾーンファイル
// に書かれていて、それを用いて解決する。file にはゾーンファイルの
// パスを書く。自分がセカンダリの場合は、type を slave とし、
// ゾーンファイルは自ら作成せずに、代わりに
// プライマリサーバの IP アドレスを指定し、そこからゾーン転送してくる
```

named.conf.default-zones 内で、ルートゾーンなどが既に指定されているので、確認する。".." はルートゾーンを表しており、ルートサーバに問い合わせるよう設定されている（問い合わせタイプ= hint）。ヒントファイル /usr/share/dns/root.hints には、ルートサーバの IP アドレスが記述されている。めったにないが、ルートサーバの IP アドレスが変更され世界中の DNS サーバの設定変更が要請される場合がある。このようなときは、ヒントファイルを最新のものをダウンロードして差し替えるか、新しいルートサーバの IP アドレスに修正する。

```
zone ".." {
    type hint;
    file  "/usr/share/dns/root.hints";
```

```
};

// ルートドメインは、ヒントファイル（通常は A から M までの 13 個のルートサーバの IP アドレスを記載）

// を使って解決。ヒントファイルのパスを file で示す。
```

4.6.3 ドメイン（ゾーン）の設定

BIND の全体設定が終わったら、自ドメインのドメイン情報（ゾーン情報）をゾーンファイルに記述する。ゾーンファイルでは、そのドメインの IP アドレス・ホスト名等の情報を記述する。

```
# vi /etc/bind/gX.zone

ゾーンファイルに記述する内容はこの先の BIND ゾーンファイル例を参照。

(以下は任意だが行うのを推奨)

# named-checkconf /etc/bind/named.conf
# named-checkzone (管理ドメイン) /etc/bind/gX.zone

# systemctl start bind9

(各種動作確認)
```

動作確認でうまくいっていなければ、設定ファイルを確認する。ログ等も必要に応じて確認する。

ログは、/var/log/syslogなどを確認する。

注意：設定ファイルを書き換えた場合は再度 named を再起動する。そうしないと、設定が反映されない。これは、postfix 等、systemctl で動作を制御する多くのソフトウェアで同様である。

```
# systemctl restart bind9
```

(1) BIND ゾーンファイルの書式

BIND でのゾーンファイルの書き方は下記の通りである。

ゾーンファイル内では、named.conf.local の zone で指定したゾーン（ドメイン）名が、ホスト名に補完して追記され、完全なドメイン名 (FQDN) となる。「.」で終わるホスト名は補完されず、それ自体が完成した FQDN ホスト名である。

```
$TTL      1H
(TTL は、キャッシュの有効期限。DNS を変更しても世界中の DNS サーバのキャッシュに情報が残り、これは最大で TTL 時間は古い情報が保持される。)

@       IN      SOA      ドメイン名の FQDN. (空白) メールアドレス. (ただし、@を.に置き換えたもの)
(
    2011072504 ; serial number 何でも良いが日付+番号が良く使われるゾーンファイルを修正したら必ずこの値を上げる。上げないとバージョンが上がってないと認識さ
```

れ、セカンダリサーバ（スレーブサーバ）で内容が更新されない。	
3h	; refresh after an hour セカンダリサーバがマスターに情報更新がないか問い合わせる間隔
1h	; retry after 1 hour リフレッシュに失敗した場合に再度リフレッシュするまでの間隔
1w	; expire after a week マスターにリフレッシュできない場合にスレーブがいつまで情報を保持するかの時間
1d)	; minimum TTL is one day ネガティブキャッシュ=存在しないという情報をいつまでキャッシュさせるか
IN NS	ネームサーバの FQDN
ホスト名 (FQDN) IN A	IP アドレス
別名 IN CNAME	実ホストの FQDN
メールドメイン IN MX	メールサーバのホスト名 (メールドメインとは、メールアドレスの@の右側である)

(2) DNS の設定項目の詳細

BIND が DNS サービスを提供するには、以下のファイルを作成する必要がある。

- データベースファイルが置かれたディレクトリ情報や named の起動オプションなどを記述したコンフィグファイル (named.conf)
- 各ドメインごとのホスト名と IP アドレスの対応を記述したデータベース情報を格納するゾーンファイル
- ルートサーバの IP アドレスを記述したヒントファイル

設定ファイルの作成には、ファイルを作成する場所（ディレクトリ）や、ピリオド “.”、セミコロン “;” に十分注意をして、vi 等のエディタで作成する。ここでの内容を間違って設定すると、DNS は正しい挙動を示さず、今後作成するサーバは正常に動作しなくなる。

ゾーンファイルの各行の意味は下記の通りである。

左端のカラムがホスト名であり、.（ピリオド）で終わるホスト名は FQDN、そうでないものはゾーンのドメイン名が補完される。名前が @ となっているものは、ドメイン名が補完される。空白のホスト名は、直前の行のホスト名が補完される。

SOA SOA レコードには、サーバの FQDN、メールアドレスを表す FQDN(メールアドレスの @ のかわりに . に置き換える)、さらにかっこの中に、シリアル番号（日付と通し番号が良い）、リフレッシュ時間、リトライ時間、expire 時間、TTL などを書く。これらの時間は、実験においては短くしておく。

NS ネームサーバの FQDN。自分自身も定義する。セカンダリがある場合はこれも書く。

MX メールサーバの FQDN。MX の後に優先順位を決める番号を書く。小さい数ほど優先される。

A ホスト名と IP アドレスを書く。

CNAME ホスト名の別名を書く。

named.conf ファイル、ゾーンファイルの作成が終了したら、設定ファイルの文法的なチェックを行う。コマンド実行の結果、エラーが出力されないことを確認する。

root.hints (システムによっては named.root とも) ファイルは下記のような意味を持っている。

- ルートサーバの情報を記述するヒントファイル。
- 世界の 13 のルートネームサーバをこのファイルにすべて記述する。
- ヒントファイルは最新のルートネームサーバの情報を記述

以上の設定を行った上で、サーバのリゾルバ設定を変更する。新しく立ち上げた DNS サーバアドレス (自分自身の IP アドレス) に変更し、同時にドメイン名も変更する。

4.6.4 本実験での BIND ゾーンファイル例

本実験でのゾーンファイルの例は下記の通りである（下記はグループ X の例であり、グループ番号は自分のものを使う）。

```
$TTL    100
@       IN  SOA  server.gX.exp.info.kochi-tech.ac.jp.  postmaster.gX.exp.info.kochi-t
ech.ac.jp.  (
              2020041701
              100
              100
              100
              100 )
              IN  NS   server.gX.info.kochi-tech.ac.jp.
server  IN  A    サーバの IP
www     IN  CNAME  server
linux   IN  A    デスクトップ linux の IP
win     IN  A    Windows 10 の IP
```

4.6.5 DNS サーバの起動と動作確認

bind の起動を行う。

```
# systemctl enable bind9
↑ 再起動時に BIND の自動起動

# systemctl start bind9
↑ BIND の（手動での）起動
(直接コマンドで起動する場合は /usr/sbin/named だがパッケージでは通常用いない)

# ps auxww | grep named
↑ named プロセスがあることを確認
```

もし起動していなければ、ログなどでエラーを確認し修正する。

```
# cd /var/log
# less syslog (など)
...
```

4.6.6 サーバOSでの参照先DNSサーバ(リゾルバ)設定

構築したDNSキャッシュサーバを参照するよう、DNSクライアントの設定を変更する。

現在は、高知工科大学DNSキャッシュサーバ172.30.0.2が設定されているはずなので、これを今回構築したDNSサーバのIPアドレスに変更する。

- サーバ

/etc/netplan/以下のYAMLファイルを修正する。

```
nameservers:
  addresses:
    - DNSキャッシュサーバのIPアドレス(自分自身なら127.0.0.1)
  search: [自グループのドメイン名]
```

netplan設定後、有効化する。

```
# netplan apply
```

サーバでdigコマンドを用いて、以下のように様々なホスト名、IPアドレスを入力し、DNSサーバの動作状態を確認する。

```
# dig server.gX.exp.info.kochi-tech.ac.jp
(ホスト名serverのIPアドレスを問い合わせ)
:
```

正しいIPアドレスが引ければ、DNSは正常に動作している。答えが返って来ない、または、間違った答えが返ってくる場合は、各設定ファイルを再確認する。また、/var/log以下のファイルにエラーログが記録されるので参考にするとよい。

```
# cd /var/log
# less syslog (など)
...
```

また、nslookupコマンドや、pingコマンドでも確認する。

```
# nslookup server.g12.exp.info.kochi-tech.ac.jp
# ping server.g12.exp.info.kochi-tech.ac.jp
```

nslookupやpingでは、自ドメイン名を省略できる。

```
# nslookup server
# ping server
```

4.6.7 クライアントOSのリゾルバ設定

サーバでの名前解決が正常に行えるようになったら、各クライアントのリゾルバ設定を立ち上げたDNSサーバに変更する。

- Windows 10
「コントロールパネル」→「ネットワークとインターネット」→「ネットワークと共有センター」→「アダプタの設定の変更」から、用いているネットワークインターフェースのプロパティから、IPv4の設定の項のDNSサーバを変更する。
- Ubuntu Desktop
`/etc/netplan/`以下の YAML ファイルをサーバと同様に変更する。ただし DNS キャッシュサーバの IP アドレスに気を付ける。

4.7 発展：Macbook, ChromeBook の固定IP設定

現在、Macbook, ChromeBook は DHCP による IP アドレス動的割り当て（自動割り当て）になっており、DNS キャッシュサーバも DHCP により大学のアドレスが設定されている。

これを、固定 IP アドレスにし、DNS サーバも今回設定したサーバに変更する。

用いる IP アドレスは、DNS の設定方針にある表の通りとする。

4.8 動作確認

自グループで構築した DNS サーバを利用し、

- 自グループ内のホスト名の正引き
- 他グループのホスト名の正引き
- 外部ネットワークの名前解決

が行えるか確認を行う。`dig`, `nslookup` で確認する。

また、実際のアプリケーション動作確認として下記も行う。

- 自グループマシンへの ping をホスト名で実施
- 自グループ Web サーバを `http(s)://www.gX.exp.info.kochi-tech.ac.jp/` でアクセス
- 外部 Web サービスへアクセス

4.9 考慮すべき点

名前の解決方法 TCP/IP ネットワーク上で名前解決を行うにはどのような技術が存在し、それぞれどのようなメリット・デメリットがあるか。

DNS による名前解決の仕組み DNS とはどのような仕組みで名前の解決を行うのか。また、リゾルバ・キャッシュサーバはどのように違うものか。

管理外ホストの名前解決 自分の管理外のホストに対しどのように名前の解決を行うのか。

hosts ファイルなどとの併用 hosts ファイルなど、他の名前解決サービスと DNS サーバを併用するときの注意。この挙動を変更したい場合はどうするか。

4.10 電子メールの送受信

電子メールは、ネットワーク上でテキスト情報をやりとりするアプリケーションである。電子メールは、差出人と宛先が明示され、差出人から宛先へと情報が転送される。宛先のアドレスが分かれれば、誰でもその宛先人に対して情報を送ることができる。このように、電子メールサービスは、「メール（手紙）」の名の通り、ネットワーク上の郵便の手紙のようなものといえる。

電子メールの配送は、差出人のコンピュータから、メールサーバをいくつか経由し、宛先のメールサーバまで配送される。宛先メールサーバでは、宛先人のメールスプールにメールを保存し、宛先人は自分のコンピュータからメールスプールへアクセスしメールを読み出す。

電子メールサーバの役割は、電子メールの受信、電子メールの送信、電子メールのメールスプールへの保存の3つである。

図4.4はあるユーザがメールを送信して相手がそのメールを読むまでのメールの動きを示している。

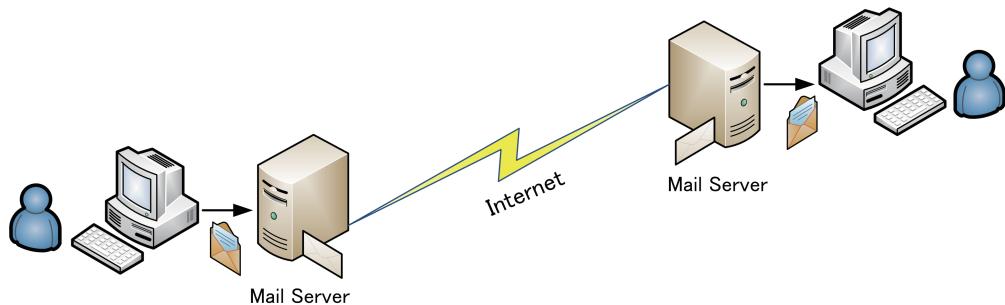


図4.4 メールの送受信

ユーザが作成したメールはメールソフト（MUA:Mail User Agent の略。UNIX の mail コマンドのようにユーザがメールの送受信を行うために使用しているソフトウェア）によりメールサーバ（MTA:Mail Transfer Agent の略。電子メールシステムにおいて、メールの配送を担当する部分。）に送られる。MTA は送られてきたメールの宛先を見て、送信先のメールサーバを決定し配送するという処理を行う。送信先のメールサーバは受け取ったメールをメールスプールと呼ばれる保管場所に保管する。最後に、相手はメールソフトを用いてメールサーバにアクセスし、メールスプールに保管された自分宛のメールを取得する。このように、メールサーバの役割はメールサーバ間でのメールの配送と MUA とのメールの送受信である。

4.11 実験内容（2）

本章において必要となる作業は

- SMTP サーバの構築
- POP サーバの構築
- MUA を用いたメールの送受信

である。

4.12 必要となる情報

構成

グループ内のメールサービスを server にて構築する。

具体的には、下記を提供する。

- グループ内ユーザ全員のメールアカウント提供
- グループ内端末からの送信用 SMTP サービス
- グループ内端末への受信用に POP サービス
- グループユーザ全員を登録した簡易メーリングリスト（エイリアスで構築）

メールサーバである server は server に登録されているユーザ<username>宛てのメール username@gX.exp.info.kochi-tech.ac.jp を受け取り、MUA がインストールされたクライアントとメールの送受信を行う。また、メールサーバである server は他のグループ宛のメールをそれぞれのグループのメールサーバに配達する。

送信用 SMTP サーバ名は、smtp.gX.exp.info.kochi-tech.ac.jp、受信用 POP サーバ名は、pop.gX.exp.info.kochi-tech.ac.jp とすること。

注意!：今回、実験内でパスワードの平文（ひらぶん）が表示される場所があるので、サーバのログインパスワードは各自、普段の生活で用いていない実験だけでの捨てパスワードに passwd コマンドで設定しなおしておくこと。

4.13 必要となる知識

メール配達と DNS

メールサーバはメールの宛先を見て送信先のメールサーバを決定する。ここでは、例えば、メールサーバが foo@gX.exp.info.kochi-tech.ac.jp という宛先のメールを受け取ったとき、どのようにして相手のメールサーバを知るか説明する。

メールサーバは、DNS を用いて相手のメールサーバを調べる。DNS には MX レコードと呼ばれるドメインにおけるメールサーバのホスト名を示す値が登録されている。つまり、メールサーバは宛先メールアドレスから相手のドメインネームを取り出し（メールアドレスの@マーク以降の文字列）、そのドメインの MX レコードを調べ、相手のメールサーバを知ることができる。例えば、メールサーバは foo@gX.exp.info.kochi-tech.ac.jp のメールを配達する場合、gX.exp.info.kochi-tech.ac.jp ドメインの MX レコードを DNS で調べて、メールサーバのホスト名を取得し、そのホスト名から（A レコードを調べることで）、IP アドレスを取得し、その IP アドレス宛にメールを送るという動作をする。

メールサーバ

メールサーバは2つのソフトウェアを使用してメールの配達とMUAとの送受信を行っている。一つはMTAと呼ばれるソフトウェアである。MTAはSMTP⁵というプロトコルを使用してメールの配達を行う。例えば、MTAはMUAからメールを受け取った時、まず、メールの宛先を見て自分が管理しているメールアドレスのメールであればマシン内のメールスプールに移し、管理していないメールアドレスのメールであれば、そのメールアドレスを管理しているメールサーバに配達するというような動作をする（図4.5）。

MTAの代表的なソフトウェアには、sendmailやpostfix、qmailなどがあり、世界中でもっとも使われているのはsendmailであるが、現在、採用が増加しているMTAにpostfixがある。本実験では、このpostfixを用いてMTAの構築を行う。

MTAにより送受信され宛先メールサーバへ到着したメールは、メールサーバ上のメールスプールに保存される。ユーザは、メールサーバにアクセスし、メールスプール上に保存されたメールを読み出しMUA上へ受信する。MUAへの受信プロトコルには、POP⁶が広く用いられており、メールサーバには、MTAの他に、POPサーバの機能が必要である。POPサーバの機能は、MUAから受信の接続を受け付け、ユーザ名とパスワードにより正当なメール受信者であるか認証を行った後、メールスプールからメールを読み出しMUAへ転送する。（図4.5）。

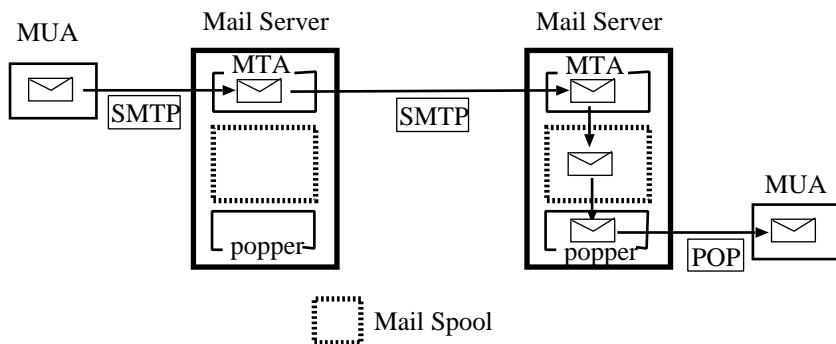


図4.5 MTAとpopper

エイリアス (aliases) ファイル

通常の電子メールサーバでは、特に設定をしない限り、メールアドレスはユーザ名@ドメイン名のようになる。

ドメイン名の部分は、DNSサーバのMXレコードで指定されるメールドメイン名や、Aレコードで指定されるメールサーバのホスト名(FQDN)である。一方、ユーザ名の部分は、メールサーバ上のログインユーザ名である。しかし、メールサーバではメールアドレス1つにつき、必ずしもログインアカウントを作成する必要はない、必要に応じてメールアドレスを追加することができる（注意：メールスプールが個別に必要である場合は、ログインアカウントが必要である）。このような場合に、メールアドレスの別名を作成する仕組みがメールにおけるエイリアスである。このエイリアスを用いることで、下記のようなものを実現することができる。

- メールアドレスの別名

⁵ Simple Mail Transfer Protocol の略。SMTPはRFC5321で規定されているプロトコルである。

⁶ Post Office Protocol の略。RFC1939に規定されている。

- 別のメールアドレスへの転送
- 複数のメールアドレスへの配送（同報アドレスなどと呼ばれる簡易的なメーリングリスト）
- 外部ファイルに記述されたアドレス全てへの配送
- 外部プログラムを呼び出し、そのプログラムへメール内容を出力（フィルタ動作等の処理）

エイリアスは、通常 /etc/aliases ファイルで記述され、書式の例を以下に示す。

**aliases ファイルの例
(メールドメイン: mail.jp とする)**

```
other: real
Family.Given: real
forward: my-home@my.provider.address.com
simple-ml: boy-a, boy-b, girl-a
process: "|program"
list: :include:/path/to/list-file
```

1 行目の例は other@mail.jp という宛先のメールが real@mail.jp に配送されることを示す。メールスプールは、ユーザ real のもの 1 つであるが、2 つのメールアドレス other@mail.jp と real@mail.jp を使い分ける場合などに用いる例である。

2 行目の例は Family.Given@mail.jp という宛先のメールが同様に real@mail.jp に配送されることを示す。Family.Given というログインユーザ名は、ピリオドが含まれる他、長すぎる点⁷が問題となり、作成することができない OS もあるが、エイリアスに登録しておけば、メールアドレスとして用いることができるため、ユーザ名とメールアドレス名を変えたい場合などに用いる。

3 行目の例は forward@mail.jp という宛先のメールが、別のメールサーバ my-home@my.provider.address.com に転送される例である。大学から家に転送する場合などに用いることができる。この場合、このサーバにはメールスプールは存在しないので、このサーバにメールは保持されない。

4 行目の例は simple-ml@mail.jp という宛先のメールが boy-a@mail.jp, boy-b@mail.jp, girl-a@mail.jp の 3 つのメールアドレスに配送される例である。簡易的なメーリングリストや同報アドレスとして用いる。

5 行目の例は process@mail.jp という宛先のメールをメールサーバが受け取ったら、program というプログラムを起動し、メールの内容を標準入力に渡し処理を行う。迷惑メールフィルタやウイルスチェックプログラムを利用したり、メーリングリストシステムのコマンド処理などに用いられる。

6 行目の例は、list@mail.jp のメールが、/path/to/list-file に記述された全てのメールアドレスに配送される例であり、高機能メーリングリストプログラムなどで用いられる。

注意点として、postfix を始めメールサーバプログラムは、/etc/aliases ファイルを直接参照するわけではなく、ハッシュ化されたデータを持つ。このため、編集後は newaliases コマンドを実行し、このハッシュデータの更新する必要がある。

⁷ 伝統的な UNIX ではユーザ名は 8 文字

4.14 postfix のインストール

MTA として最も歴史が古く、広く用いられているソフトウェアに sendmail があるが、古いためにソフトウェアの構造、設定が複雑で、設計に古い部分もある。近年、MTA としての機能や性能の向上をはかった qmail や postfix などのソフトウェアが普及してきている。本実験では、sendmail との運用互換性が高い postfix を用いて MTA の構築を行う。

postfix を apt からインストールする。

```
$ sudo su

# apt update

# apt install postfix
(メニューにて)
Internet Site [Enter]
を選択

mail name: には、英語で指示された通り（メールのドメイン名）を入力する。

# vi /etc/postfix/main.cf
■ mynetworks に自ネットワークのセグメントを追加
(xxx.xxx.xxx.xzz のようにネットワークアドレスと
プレフィックスによる表記で記述。192.168 の場合は、/24 全てで良い)
■ myhostname がサーバの FQDN を書く

# systemctl restart postfix (もう動作しているため)
```

以上の設定が問題なく終われば、下記の 3 つのプロセスが動作していることを確認する。

```
# ps auxww | grep post
root      8736  0.0  0.2  65408 ...19:06  0:00 /usr/lib/postfix/sbin/master
postfix   8737  0.0  0.2  67476 ...19:06  0:00 pickup -l -t unix -u -c
postfix   8738  0.0  0.2  67524 ...19:06  0:00 qmgr -l -t unix -u
```

これで、MTA の設定は終了である。SMTP プロトコルを用いて、実際に送受信の確認を行う。

```
# telnet 127.0.0.1 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 server.gx.exp.info.kochi-tech.ac.jp ESMTP Postfix
HELO server.gx.exp.info.kochi-tech.ac.jp
```

```

250 server.gX.exp.info.kochi-tech.ac.jp
MAIL FROM: exp@gX.exp.info.kochi-tech.ac.jp
250 2.1.0 Ok
↑ SMTP の発信元アドレスを通知

RCPT TO: postmaster@gX.exp.info.kochi-tech.ac.jp
250 2.1.5 Ok
↑ SMTP の宛先アドレスを通知 (MTA はここへ配達)

↓ ここからメールの内容 (ヘッダ+本文)
DATA
354 End data with <CR><LF>.<CR><LF>
To: postmaster@gX.exp.info.kochi-tech.ac.jp
From: exp@gX.exp.info.kochi-tech.ac.jp
Subject: TITLE
(空行) ↑ ここまでがヘッダ、空行を入れて ↓ 以下が本文
Honbun
-
250 2.0.0 Ok: queued as 29D7D1CC1F
QUIT
221 2.0.0 Bye
Connection closed by foreign host.
serverX#

```

メールスプールは、/var/mail に設定されている。Postmaster宛のメールは、root へ届くよう設定したので、root のメールスプール/var/mail/root の内容を less などのページャで確認する。下記のようにファイルの末尾にメッセージが書き込まれていれば、メールは届いている。

```

From yourname@your.mail.domain Thu Jun 16 09:38:39 2011
Return-Path: <yourname@your.mail.domain>
X-Original-To: postmaster@gX.exp.info.kochi-tech.ac.jp
Delivered-To: postmaster@gX.exp.info.kochi-tech.ac.jp
Received: from myserver?name (localhost [127.0.0.1])
        by server.gX.exp.info.kochi-tech.ac.jp (Postfix) with SMTP id
29D7D1CC1F
        for <postmaster@gX.exp.info.kochi-tech.ac.jp>; Thu, 16 Jun 2011
09:37:39 +0900 (JST)
To: postmaster@gX.exp.info.kochi-tech.ac.jp
From: yourname@your.mail.domain
Subject: TITLE
Message-Id:
<20110616003800.29D7D1CC1F@server.gX.exp.info.kochi-tech.ac.jp>
Date: Thu, 16 Jun 2011 09:37:39 +0900 (JST)

Honbun

```

冒頭の From は、SMTP の MAIL FROM: コマンドで入力したもの、2行目以降 DATA コマンドで入力したものに、配送途中の MTA が書き込んだ情報が追加されている。最初の空行までがヘッダ、それ以後、ファイルの末尾か、次の From で始まる行までがメッセージの本文である。このようなメッセージのフォーマットは、RFC 5322 で規定されている。

次に POP をインストールする。POP には dovecot を用いる。

```
# apt update
(上記は、postfix の際に行っていれば不要)

# apt install dovecot-pop3d
```

デフォルトでは、生パスワードでの POP3 認証が無効になっているため、下記ファイルを修正して有効にして dovecot を再起動する。

```
# vi /etc/dovecot/conf.d/10-auth.conf
disable_plaintext_auth = no

# systemctl restart dovecot (もう動作しているため)
```

POP についても同様に、telnet で動作を調べることができる。

```
server#telnet localhost 110
USER ユーザ名
PASS パスワード
+OK
LIST
QUIT
```

注意 パスワードは生で画面表示されるので、差障りのある場合は passwd コマンドなどであらかじめ差障りのないものに変更しておくこと。

PASS コマンドでパスワードを入力した後、ERR が出力されずに OK と出力されれば良い。ただし、root ユーザは認証に失敗する OS がある。

4.14.1 DNS の設定の追加

(1) MX レコードの追加

前述のようドメイン名によるメール配信を行うには、DNS に MX レコードが必要である。そこで、以下の設定を追加する。ただし、DNS にて既に設定済みの場合は必要ない。また、SMTP サーバ、POP サーバも CNAME にて別名設定しておく。

■ /etc/bind/gX.zone ファイル

メールドメイン	IN	MX 10	メールサーバホスト名
---------	----	-------	------------

なお、メールドメインとメールサーバホスト名を FQDN 指定する場合は、最後のピリオドの有無に気をつける。

(2) CNAME の設定

この設定の意味は、「server のマシンの別名として, pop という名前をつける」ことである。つまり, pop サーバ名として「pop.gX.exp.info.kochi-tech.ac.jp」を設定している（実体は server である）。同様に SMTP も CNAME として与える。DNS にて設定済みの場合は必要ない。

■ /etc/bind/gX.zone ファイル（正引きデータベース）

pop	IN	CNAME	server
smtp	IN	CNAME	server

設定を変更した後に、BIND の再起動を行う必要がある。また, nslookup, dig, ping 等のコマンドを利用して、正常に動作しているかの確認も忘れないようにすること。

4.15 MUA の設定, および, postfix, dovecot のテスト

クライアントの MUA の設定を行い、メール送受信のテストを行う。

Cent OS, Windows には、メールクライアントが付属していないため、GNU Public License でフリーの使用ができる Thunderbird を Mozilla⁸ のサイト（下記 URI）からインストールする（言語や環境別になっている）。

<https://www.mozilla.org/en-US/thunderbird/all/>

CentOS は、Applications → System Tools → Software から、Mozilla Thunderbird mail/newsgroup client をインストールすれば良い。メニューの Internet の中に Thunderbird が表示される。

MacOS X の切手のアイコンの MUA は、OS インストール時にデフォルトでインストールされる MUA であるので、これを用いても良いし、Thunderbird を別途インストールしても良い。

なお、Thunderbird は、拡張子が .exe のものが Windows 版インストーラ、.dmg は Mac 版、.tar.bz2 は Linux 用である。

また、Server 用 MUA として通常の UNIX では mail コマンドがあるが、Ubuntu では、apt install mailutils としてインストールする必要がある（今回の実験では、mail コマンドが無くても実験できるが、必要と判断すればインストールするのは構わない）。

その他の MUA の設定

MUA 設定は下記のようにユーザーアカウントの設定を行う。

名前	:	適当で良いがローマ字が無難
電子メールアドレス	:	ユーザ名@gX.exp.info.kochi-tech.ac.jp
送信メール	:	smtp.gX.exp.info.kochi-tech.ac.jp
受信メール	:	pop.gX.exp.info.kochi-tech.ac.jp
POP3 アカウント	:	ユーザ名
パスワード	:	(server で設定したユーザのパスワード)

⁸ Mozilla はオープンソースのブラウザ、メールクライアント等を開発している団体

以上の設定を行った後、クライアント端末の MUA を使って、ユーザ間でメールを送信し、お互いにメールが受信できるか確認する。

送信/受信できなかった場合は、設定内容や DNS, postfix が起動しているかを確認する。/var/log/ ディレクトリの syslog, maillog ファイルにログが記録されるので、これを利用して原因を調べる。

Thunderbird の設定（参考）

新しいアカウントを設定する → メール → メールアカウントを設定する
(New Message → Existing Mail Account...)

（ダイアログ：メールアカウント設定）

（Mail Account Setup）

あなたのお名前：ローマ字名等（適当に）

メールアドレス：exp@gX.exp.info.kochi-tech.ac.jp

パスワード：（記録させる場合は入力。毎回入力してもよい）

（入力ができたら）「続ける」

自動で進められる場合は警告の画面の設定に進む

自動的に設定が完了したら危険性についての警告を理解したうえで進める、とすると設定が完了する

- 主な設定内容の確認ファイル一覧

postfix : /etc/postfix/main.conf
 /etc/aliases
DNS : /etc/bind/gX.zone

4.16 考慮すべき点

今回の実験を行うにあたり、以下のようなことについて考慮する必要がある。

宛先 電子メールはどのようにして届け先を判別するのか。

メールサーバ メールサーバは電子メールを送受信する上でどのような働きをするのか。

第 5 章

データベース・Web システム

今日においては、基幹システムなどが組織内ネットワークやインターネットといったネットワークを介する Web システムとして提供されることも多い。大規模なシステムにおいては基本的にデータの管理にデータベースを使うことになるため、Web システムとの連携技術が重要となる。また、そのようなシステムではアクセスする状況やデータそのものの変化に応じた処理を行う必要があり、Web ブラウザに返す Web 文書を Web サーバが動的に生成する技術が必要である。

5.1 データベース

コンピュータシステムにおいては、データの記憶は単に保存領域に保存することだけでなく、必要なデータを容易に取り出せる検索容易性も重要な要素である。通常、データをそのまま工夫せずに記憶した場合、必要なデータを探して取り出すためにはデータの先頭から比較照合を行いながらデータを探す必要がある。このため、データに構造を持たせて記憶する（構造化）。例えば、配列に添字（index）を付与する、表形式（table）にてデータを記憶するなどは、データの構造化の例である。

データベースには、検索容易性やデータの一貫性を確保するために、いくつかの技術が用いられているので、下記で説明する。

5.1.1 データ形式

一般に、データベースでは検索を容易にするために、データに対していくつかの属性を付与して記憶する。最も単純なものは配列であり、添字である整数に関連付けてデータを記憶する。配列では、添字がデータに対するキーとなり、キーを指定すると、キーに関係するデータを読み出すことができる。このように、データベースにおいては、データを検索する際にシステム入力する項目をキーと呼ぶ。

配列をさらに進めたものが、連想配列である。連想配列は、キーとして一般の文字列を用いたものであり、perl や ruby, python などのスクリプト言語には最初から組み込まれている。

配列や連想配列では、どちらもキーとデータのペアを記憶しており、キーを手がかりとして関係するデータを取得している。このペアをさらに拡張して、データに対して複数の属性を持たせたものが、テーブル形式によるデータの記憶である。表計算ソフトウェアで用いられる表は、その例であり、例えば、住所録では、データに対して、「学籍番号」、「氏名」、「住所」、「電話番号」、「生年月日」などの属性を付与して記憶する。テーブルでは、データの属性は対等であり、どの属性をキーにして、どの属性を出力するかは検索に応じて決まる。例えば、特定の氏名に対して、その人の住所を出力することもあれば、特定の住所に対して対応する氏名を求めることがある。現在のデータベースシステムは、このテーブル形式でのデータ記憶を基本となっている。

ここで、各属性を持つ一つのデータをレコード、そのデータの属性のことをフィールドと呼ぶ。

5.1.2 ハッシュ (Hash)

記憶したデータから、キーに適合するものを照合しながら探す作業は、データ数（レコード数）を n とすると、一般に $\mathcal{O}(n)$ の計算コストがかかる。辞書において必要な単語を求める際に、インデックスがなければ最初の項目から一つずつ探ししていくことになることと同じである。

このような計算コストがかかるのは、必要なデータが格納されている位置が分からぬいためである。このため、必要なレコードがどこに格納されているかを知ることができれば、その記憶場所を読み出すことができるため、データの取り出しを早めることができる。理想的には $\mathcal{O}(1)$ にすることができる。このため、データにインデックスを付与する。配列はキーそのものがインデックスであったが、一般的なデータベースでは、キーは文字列などのデータである。このキーとインデックスの関連付けの方法により、検索の容易さが決まる。例えば辞書において、頭文字ごとにデータを集める方法は、データの検索速度を頭文字に使われる文字数の分だけ減らすことができる。アルファベット 26 文字であれば平均的に 1/26 に、日本語であれば 1/50 になる。

このように、何らかの形でデータのキーから、そのデータを代表する値を求め、その値ごとに記憶場所に決めることを、インデクシング (indexing, 索引付け) と呼び、その代表値をインデックス (index, 索引) と呼ぶ。インデクシングはデータ検索の基本的な技術である。

キーからインデックスを求める際、なるべくデータの分布に偏りがないようにすることが望ましい。辞書の例で言うと、「z」で始まる単語は少なく、「e」で始める単語は多いため、検索キーにより検索性能にばらつきが起きてしまう。そこで、ハッシュ関数 (hash function) を用いて、ばらつきを極力減らしたテーブルがハッシュテーブル (hash table) である。

ハッシュ関数は、ハッシュテーブル以外にも多くの分野で用いられ、多くのハッシュ関数が定義されている。また、それぞれの分野において、ハッシュ関数に求められる性質は異なるが、データ検索の分野では、なるべくばらついていることが重要である。

5.1.3 リレーションナルデータベース (RDB)

リレーションナルデータベース (関係データベース) は、複数のテーブルの間でキーを介して新たなデータの関係を見出すことを可能にするデータベースである。例えば、顧客がどのようなものを購入したかの売上一覧のテーブルと、顧客情報テーブルから、男性が購入した商品の一覧や、関東地方に出荷された商品の一覧を求めることができる。

このようなデータベースのモデルの理論的な背景は、1960 年代から 70 年代にかけて IBM の Edgar F. Codd により体系付けられた関係モデルである。その後、1970 年の Codd の論文¹に触発され、Larry Ellison により、データベースに応用された。Larry Ellison は 1970 年代にデータベースソフトウェアである Oracle を開発し、そこで SQL と呼ばれるデータ検索言語を開発した。現在、Oracle は世界でも有数のシェアを持つデータベースソフトウェアであり、SQL も多くのデータベースソフトウェアに採用され、標準化されている²。

¹ Codd, E. F. : "A relational model of data for large shared data banks," Communications of the ACM 13 (6), pp.377-387 (1970)

² ISO/IEC 9075 : 2008

5.1.4 SQL

SQL は、現在一般に用いられているデータベース管理システム (DBMS) での検索クエリを表す言語である。

その文法は、機能に応じて下記の 3 つに大別される。

- データ定義
- データ操作
- データ制御

データ定義には、CREATE, DROP, ALTER などがあり、それぞれ、テーブルの作成、削除、(テーブルそのものの定義) 変更などである。

データ操作は、テーブルに対してのデータ操作であり、INSERT, UPDATE, DELETE, SELECT などであり、それぞれデータの挿入、更新、削除、取り出しである。

データ制御は、データ操作の権限付与、削除を行う GRANT, REVOKE、データの一貫性を確保するためのトランザクションの BEGIN, COMMIT, ROLLBACK などである。

代表的な書式を下記に示す (5.4.5 節でも様々な例を示す)。

データ挿入 INSERT INTO テーブル名 (フィールド名 1, フィールド名 2) VALUES (値 1, 値 2);

レコード挿入 INSERT INTO テーブル名 VALUES (値 1, 値 2, …, 値 n)

更新 UPDATE テーブル名 SET フィールド名 2=値 2, フィールド名 3=値 3 WHERE 列名 1=値 1;

削除 DELETE FROM テーブル名 WHERE フィールド名 1=値 1;

取り出し SELECT * FROM テーブル名 WHERE 列名 1=値 1;

文末のセミコロン「;」は、C/C++ 言語と同様必ず必要なので、気を付ける。

5.1.5 代表的な RDBMS

代表的な RDBMS を挙げる。

MySQL

GPL で配布されるオープンソースソフトウェアであり、多くの Web ベースアプリケーションなどのバックエンドとして用いられる。1995 年にスウェーデンの企業 MySQL AB により開発され、Sun、さらに Oracle に買収されている。コマーシャルライセンスのものもある。

PostgreSQL

BSD ライセンスに類似するライセンス形態で配布されるオープンソースソフトウェア。カリフォリニア大学バークレイ校で 1980 年代から開発されており、歴史の古いソフトウェアである。

Oracle

Oracle 社が開発・販売する商用ソフトウェアであり、世界の商用ソフトウェアで有数のシェアを持つ。

Microsoft Access

Microsoft Office システムの 1 つとして販売されており、小～中規模のデータベース市場で用いられることがある。

Microsoft SQL Server

Microsoft社が販売する、中～大規模データベースソフトウェアである。Windowsベースのサーバシステムで用いられることが多い。

これらは、すべてSQLによる問い合わせを行うことができるが、それらの互換性は微妙に異なる。このため、必要とするソフトウェアがこれらのデータベースと一緒に運用することができるかどうかは、よく調査しておく必要がある。

5.2 Webシステム

動的コンテンツは、Webアクセスの際にプログラムを動作させることで、同じURLにアクセスした場合でも状況に応じて出力を変化させる仕組みであり、これまで解説したCGI、PHPなどもその一例である。この動的コンテンツは、サーバおよびブラウザ上でプログラムを動作させて、様々な処理を行うため、Web上のアプリケーションとして応用することができる。このような仕組みを一般に、Webアプリケーション、Webアプリケーションシステム、あるいは単にWebシステムと呼ぶ。

また、一般に、頻繁にデータの更新をWebシステムで行う場合は、データの書き込み、読み込み部分を、データベースを通して行うことが多い。これは、データベースへのアクセスが、データベースアクセス言語であるSQLを通して行うことができるため、データの書き込み、読み込み、更新などがプログラムから処理をしやすいこと、ファイルのロックなど、データの一貫性や同時アクセスの際の正常なデータ処理をデータベースシステムに委ねることができることなどが理由である。

このように、Webシステムでは、一般にバックエンドにデータベースを用いる。バックエンドに用いられるデータベースは、MySQLをはじめ、PostgreSQLやOracleなどがあるが、オープンソースのものではMySQLが多い。このため、OSであるLinux、HTTPサーバApache、MySQL、PHPのWebシステムのコンポーネントをまとめて、LAMPなどと呼ぶこともある。

5.2.1 動的コンテンツ

WWWサービス構築の際に述べたように、HTTPサーバによるWebサービスには、そのコンテンツの生成の方法により、2通りの実現方法がある。一つ目は静的なコンテンツであり、もう一方は動的なコンテンツである。静的なコンテンツは、情報の発信者が、あらかじめ公開するファイルをHTML形式や画像などのファイル形式で作成しておき、これをHTTPサーバの公開するディレクトリの配置することで、ファイルを公開する。これに対して、クライアントからのリクエストに応じて、その場で処理を行い結果となるコンテンツデータを動的に生成して、レスポンスを行うものを動的コンテンツと呼ぶ。

動的コンテンツの実現技術として下記のようなものがある。

SSI

Server Side Includeと呼ばれ、HTML形式で記述されたコンテンツのなかに、いくつかのコマンドを記述することができ、そのコマンドをサーバはリクエストを受けるたびに実行し、そのコマンドが書かれた部分を、そのコマンド結果の出力で置き換える。

CGI

Common Gateway Interfaceと呼ばれ、サーバはリクエストを受けると、そのURIで指定されたプ

ログラムを実行する。このプログラムの標準出力に出力されたデータを、HTTP サーバはそのままクライアントへ転送する。プログラムは、HTML でも画像ファイルでも任意のデータを出力できるが、正しい形式で出力する必要がある。また、プログラムは実行可能権限がなければ実行できない。また、スクリプト言語で書かれたプログラムは、一行目でそのスクリプトの正しいパスが書かれていないと実行できない。Perl で書かれているものが最も多く、一部 Ruby や Python もあり、性能を重視した CGI の場合は、C 言語なども用いられる。

PHP

HTML 形式のファイルの中に、`<?php ~ ?>` と書いた部分にプログラムを記述することができる。CGI に比較して、HTML 形式の出力をプログラムで記述する必要がなく、Web アプリケーションの開発言語として広く使われている。

その他

Java 技術を用いた Java Servlet, JSP (Java Server Pages), Microsoft 社の Visual Basic 技術, ASP 技術を用いた、VBScript, ASP などがある。

5.2.2 Wiki

Wiki はコンテンツ作成を、Web ブラウザから行うシステムの一種であるが、コンテンツの作成・編集がなるべく簡単な操作・記述で行うことができるよう、独自のタグを用いて、見出しや新しいページの作成、他のページやファイルへのリンクを記述できるようになっている。

複雑な認証機構などを持たないものが多いので、小規模な Web ページ公開で用いることが多いが、大規模な運用も運用次第では可能である。

5.2.3 ブログ

ブログ (Blog) システムは、Weblog を略したものであり、もともとは個人が Web 上に様々な出来事や考えを記していた日記のようなコンテンツを発展させたものである。現在では、ブログは世に広く普及し、個人の情報発信だけでなく、有名人や企業からのプロモーション、政府、機関などからの公式な情報発信手段としても、しばしば用いられる。

日記 (diary) の形のコンテンツは、ブログ以前から既に広く普及していたが、これを Web システム上でコンテンツ管理を行えるように、更に、コメントトラックバックという機能で、記事の読者の側からも情報に対するコメントを返信・発信できるようにしたものがブログである。ブログシステムの構築のために多くのソフトウェアがあるが、MovableType と WordPress がよく用いられる。MovableType はブログシステムの初期から使われてきたが、現在では、GPL で配布されるオープンソースの WordPress が広く使われている。

5.2.4 CMS (Content Management System)

Content Management System は、その名の通り Web コンテンツを管理するための Web システムであり、Web コンテンツそのものを Web の仕組みを用いて、作成、編集、管理するものである。これまで、エディタや HTML 編集ソフトウェアを用いて、別途 HTML ファイルの形でコンテンツを準備し、Web サーバの公開ディレクトリに置くことで公開していたが、作成から公開、編集、管理まですべて Web ブラウザから行えるようになる。

CMS の構築のためには、どのユーザがどのような処理をしているかを Web サーバ側で把握するために、Cookie などの仕組みが用いられる。

5.2.5 SNS (Social Networking System)

特定のコミュニティあるいは不特定多数の人との人間関係のネットワークを築いたり、維持したりするためのシステムである。コミュニティを新たに作り出すなどの目的にも用いられる。コミュニケーションツールの一つとも捉えることができ、電子メールや従来のメッセージツール、掲示版 (BBS) に取って代わるツールともなり得る。Facebook や LINE, twitter のサービスが有名であり、無料電話サービスからはじまった Microsoft の Skype, Google+, Instagram, 中国の微博 (ウェイボー), テンセント QQ など多くのサービスが世界で展開されている。

5.3 実験内容 (1)

データベースのインストール

MySQL をインストールする。

データベースの操作

指定されたデータのインポートを行い、データベースの操作を行う。

(1) 端末の設定

- 日本語をデータとして用いるため、Windows の Putty や Desktop Linux や Mac のターミナル（端末）を使う。サーバのコンソールでは標準では日本語は文字化けする。
- Putty 等の端末の文字コードを UTF-8 に設定
- Mac や Linux はデフォルトが UTF-8
- ssh で server にログイン
- ログイン後、シェルの文字コードを UTF-8 に設定
- シェルの文字コードは、環境変数 LANG で設定（サーバ日本語設定については、5.4.2 節を参照）

(2) データのダウンロード、文字コード変換

インポートを行うデータはメインサーバ (192.168.0.1) の /pub/data ディレクトリにある CSV 形式の以下の 3 ファイルである。

- bunrui.csv
- tanka.csv
- uriage.csv

これは、ある中華料理店の商品分類表、メニュー単価表、売上表を Excel にて作成し、CSV ファイルで保存したものである。

(3) nkf で日本語文字コード変換

データファイルは、Microsoft Excel で作成され保存された Shift-JIS コードによる CSV ファイル³で、一行目にフィールドの項目名が日本語である。改行コードは Windows 形式の改行である。このままでは MySQL で用いることができないため、日本語コード、改行コードをそれぞれ UTF-8、UNIX 改行への変更を行う。

Unicode 対応の日本語コード変換および、日本語テキスト閲覧プログラムを導入する。前者として nkf、後者として lv がよく用いられる。

- nkf コマンドを apt を使いインストール

なお、Ubuntu では通常のページャ less も UTF-8 に対応しているので、lv を用いずにそれをそのまま用いても良い。Ubuntu のパッケージ apt にて nkf（必要があれば lv も）を導入し変換する。使い方は、それぞれ--help や -h オプションによるヘルプを参照のこと。nkf 使用前・使用後は、nkf --guess にてコードの確認をすること。

- nkf コマンドを使い、
 - 文字コード: UTF-8
 - 改行コード: UNIX

に変換

(4) データベース構築

- データベース名は sales
- データベースはの文字コードは UTF-8 とする

(5) テーブル構築

テーブル sales, price, foodgroup を作成する。それぞれの属性は下記のようとする。

以下、テーブル名: フィールド名(属性), ... を示す。

- sales : date(text), food(text), amount(integer)
- price : food(text), fee(integer)
- foodgroup : food(text), foodgroup(text)
- コード変換後の urage.csv, tanka.csv, bunrui.csv をそれぞれテーブル sales, price, foodgroup へ読み込む
- ただし、一行目は読み込まないように vi などでファイルから先頭行を削除しておく。

データのインポート後、「describe テーブル名」や「select * from テーブル名」で内容を確認しておく。

³ CSV: Comma Separated Value カンマ','で各フィールドが区切られた表データ。一般的な表計算ソフトウェアで読み書きできる形式である。

(6) データ検索・操作

ここまで の作業を行った後、以下のデータベース操作を行う。

- データの追加
 - テーブル sales の 2006/12/31 の売上にあるサイダー 2 本を削除し、同日のオレンジジュースを 2 本追加する。
- データ抽出
 - テーブル sales から 2006/12/26 に売り上げた食品の品目と個数を表示する（日付の表示はしない）。
 - テーブル sales の食品名の横に、テーブル foodgroup の分類名を参照して、日付、食品名、分類、個数を表示する。
 - テーブル sales から、12/30 に売り上げたもののうち、麺でかつ単価が 750 円以上のものについて、売り上げ日、食品名、個数を表示する。

5.4 必要な知識

5.4.1 文字コード・改行コードの種類

データベースの作成に先立ち、様々なテキストファイルを扱う必要があるが、日本語には様々な文字コードがあり、また OS によって改行コードも異なるため、これらに対応する環境を構築する必要がある。

文字コードには、シフト JIS, EUC-JAPAN, JIS, UTF-8 (Unicode) があるが、MySQL 等サーバ系では、今後は可能な限り UTF-8 を統一していく方が良いだろう。

しかし、Excel などシフト JIS を扱うアプリケーションも多いため、変換が必要である。

文字コード	主な用途	LANG, LC_ALL 等 環境変数の値
UTF-8	ユニコード一つ、汎用的、古いシステムでは動作しない可能性あり。ASCII 互換のため、世界的に広く用いられるユニコード。1 文字 1~3 バイト。Linux 等で用いられる。なお、Unicode には他にも多くの規格がある。	ja_JP.UTF-8
シフト JIS	Microsoft Windows, Apple Mac OS X 等の標準。Excel の出力する CSV ファイル等で用いられる。日本語 1 文字 2 バイト。	ja_JP.SJIS
EUC	Extended Unix Code の略（古い UNIX 用、FreeBSD の標準日本語コード）。日本語 1 文字 2 バイト	ja_JP.eucJP
JIS	JIS コード、日本工業規格標準、7 ビットのみで簡潔するため、電子メールで用いられている。電子メールでの JUNET, ISO-2022-JP も同じコード。日本語 1 文字 2 バイトだが、日本語コード列の始まりと終わりに各 1 バイトの制御コード（エスケープシーケンス Kanji-in, Kanji-out）が入る	ja_JP.JIS0208

また、改行コードには 3 種類ある。

システム	コード名称	ASCII コード (16 進数)
UNIX	LF (ラインフィード: 新しい行を追加するという意)	0A
Microsoft Windows (DOS)	CR + LF	0D 0A
Apple Mac OS	CR (キャリッジリターン: 行頭に（カーソルを）復帰するという意)	0D

異なるコードに対応し、相互変換するために日本では nkf が広く用いられる。

日本語コード変換フィルタ NKF の使い方

主な使い方

- ・ファイルの文字コードを調べる
`nkf --guess ファイル`
 - ・file1 を UTF-8, UNIX 改行にして file2 とする
`nkf -w -Lu < file1 > file2`
- 他のオプションは `nkf --help` で調べる

5.4.2 日本語 UTF-8 を扱うための手順

日本語 UTF-8 を扱うためには、下記の 3つ全てを UTF-8 に対応させる。

(1) 端末 (CentOS の「端末」, Mac OS X の「ターミナル」, Putty の設定.)

(2) シェル (ここでは root の sh)

(3) データベース

(a) mysql コマンドの設定

(b) MySQL サーバ (mysqld) の設定

(c) 作成するデータベースでの設定

1つ目は、Putty の設定の「ウインドウ (Window)」→「変換 (Translation)」→「文字セット変換 (Character set)」の設定を UTF-8 にする⁴.

2つ目は、Putty 等からサーバにログイン後、下記の設定をする (5.4.1 節).

(LANG 変数のデフォルトは「C」).

シェルの環境変数 LANG に以下の値を代入

```
# locale-gen ja_JP.UTF-8  
(↑最初の 1 回だけ。これは下記コマンドで ja_JP.utf8 がないとき行う、locale の追加生成コマンドである)  
  
# locale -a  
  
# export LANG=ja_JP.UTF-8  
  
# locale  
LANG=ja_JP.UTF-8  
となることを確認
```

環境変数を消去する場合は、

```
# unset LANG
```

とする。

以上の環境変数は、シェルを exit するたびに、消える。

再度ログインするたび設定する。

もし、自動的に設定したい場合は、シェルのスタート時に読み込まれる

下記ファイルの最終行に書き込んで置くと、ログインのたびに読み込まれる。

⁴ CJK あるのは、Chinese, Japanese, Korean の特殊な文字にも対応するということ。

```
# vi ~/.bashrc
```

3つ目は5.4.3節の2つを設定する。

5.4.3 MySQL インストール

serverにパッケージシステムからmysqlをインストールする。

```
# apt install mysql-server
```

MySQL用の管理者アカウント(`root`)パスワードを入力する。

設定ファイルは、`/etc/mysql/my.cnf`であるが、個別のセクションについては、`/etc/mysql/conf.d/`および`/etc/mysql/mysql.conf.d/`以下の別ファイルに分けられ、`my.cnf`から`include`されている。

```
# vi /etc/mysql/conf.d/mysql.cnf
```

下記の行を末尾に追加

```
default-character-set=utf8
```

```
# vi /etc/mysql/mysql.conf.d/mysqld.cnf
```

下記の行を末尾に追加

```
character-set-server=utf8
```

MySQLの再起動

```
# systemctl restart mysql
```

5.4.4 データベース構築の手順

データベース構築の手順を下記に示す。

- (1) 日本語の入出力が必要になるため、日本語の入力・表示ができる端末を用いる必要がある。具体的には、WindowsのPuttyを使い、「Window(ウィンドウ)」→「変換(Translation)」の項目で、Remote Character SetをUTF-8(CJK)⁵に設定する。
- (2) Puttyから、ServerにSSHでログインし、rootになる。
- (3) 漢字コード・改行コード変換フィルタをServerにパッケージからインストールする。手順は以下の通り。

⁵ Unicode規格のUTF-8文字コードで、Chinese, Japanese, Korean文字を使うモード

```
# apt install nkf  (< nkf のパッケージ名の設定)
  (日本漢字フィルタ)
```

- (4) CSV (Comma Separated Value) 形式 (カンマ “,” で各フィールドの値が区切られた表 (テーブル) を表したテキストファイル) のファイルを入力に用いる。Microsoft Excel などで作成された CSV ファイルは、日本語文字コードが Shift JIS、改行コードは、Windows/MS-DOS 用改行コードになっていることが多い。
- (5) ftp にて、3つの CSV ファイル (bunrui.csv, tanka.csv, uriage.csv) をダウンロードする。

```
anonymous (ftp) でログインする
# ftp 192.168.0.1
Connected to 192.168.0.1.
220 mainserver FTP server (Version 6.00LS) ready.

FTP ログインユーザ名は ftp か anonymous
Name (192.168.0.1:exp): ftp
331 Guest login ok, send your email address as password.
パスワードは空
Password:

ftp> cd /pub/data
ftp> ls
ftp> get ファイル名.csv

3つの CSV ファイルをダウンロード
```

- (6) nkf コマンドで文字コードを調べる。

```
# nkf --guess ファイル名
```

- (7) nkf コマンドで文字を変換する。

```
# man nkf または nkf --help | less
オプションの意味を調べる
# nkf 必要なオプション ファイル名 > 別のファイル名
```

必要なオプションは、「出力を UTF-8 にすること」、「出力を UNIX 改行にすること」の2点である。このようにして UTF-8, UNIX 改行コードに変換する。

- (8) 各データファイルの先頭行には各列のタイトルがあり、これは MySQL にインポートする際には必要無いため、1行目を削除したものを MySQL の入力に用いる。

データベースの文字コード

設定の確認はデータベース起動後に、mysql コマンドで MySQL に接続し、下記のコマンドで調べることができる。

```
mysql> show variables like 'char%';
+-----+-----+
| Variable_name      | Value   |
+-----+-----+
| character_set_client | utf8mb3 |
| character_set_connection | utf8mb3 |
| character_set_database | utf8mb3 |
| character_set_filesystem | binary |
| character_set_results | utf8mb3 |
| character_set_server | utf8mb3 |
| character_set_system | utf8mb3 |
| character_sets_dir | /usr/share/mysql/charsets/ |
+-----+
8 rows in set (0.01 sec)
```

データベースへの接続

mysql コマンドを用い、-u で接続するユーザ名を -p でパスワードによる認証を行う。

```
$mysql -u root -p
```

データベース一覧の出力

存在するデータベースを一覧で表示したい場合、show databases; (mysql のコマンドは ; までを 1 行と解釈する) を用いる。

```
mysql> show databases;
```

MySQL が自身の動作に用いるデータベースがいくつか作成されている。

データベースの作成

新規にデータベースを作成する場合は create database を用いる。例として testdatabase を作る場合

```
mysql> create database testdatabase;
```

となる。

データベースの選択

データベースを利用する場合は `use` コマンドでデータベース名を指定する。

```
mysql> use データベース名;
```

テーブルの作成

データベースに対し、新規にテーブルを作成する場合 `create table` コマンドを用いる。例として、データベース `testdatabase` に対し、テーブル `address` を作成する場合、

```
mysql> use testdatabase;
mysql> create table address (
    -> name text,
    -> address text,
    -> phone integer );
```

となる。ただし、テーブル `address` は要素として `name(text)`, `address(text)`, `phone(integer)` の 3つを持っている。

ファイルからのインポート

`mysql` では、`load` コマンドで外部ファイルからの一括入力が行える。例えば、各行にフィールドの値「名前、住所、電話番号」を列挙したファイル `addresslist` から先ほどの `testdatabase` 上のテーブル `address` にインポートを行う場合

```
# mysql -u root --local-infile=1 -p
mysql> use testdatabase;
mysql> load data local infile "addresslist" into table address fields
terminated by ',',';
(addresslist ファイルから、',,' で区切られた各項目を、テーブル address の各フィールドへ代入
する.)
```

となる。

ただし、MySQL 8.0 から安全上の理由で、サーバ側でのローカルファイルからのデータ一括入力の可否を制御する変数 `local_infile` のデフォルト値が 1 (可) から、0 (不可) の変わっているので、その場合は 1 に設定する。

```
mysql> select @@local_infile;
+-----+
| @@local_infile |
+-----+
|          0   |
+-----+
```

↑ 0 と表示されたら、下記コマンドで 1 にする

```
mysql> set persist local_infile=1;
↑ 通常一度設定したら、再度設定するまでは有効
```

表の確認

テーブルの内容を確認したい場合、`select * from テーブル名`となる。例として、テーブル `address` の中身を確認する場合、

```
mysql> select * from address;
```

となる。

データベースの閲覧

例として、データベースとして `mysql` にある `user` テーブルに設定されているユーザ情報を閲覧する場合は、下記のように行う。

```
$mysql -u root -p
mysql> show databases;
          (mysql データベースがあることを確認)
mysql> use mysql;
mysql> show tables;
          (user テーブルがあることを確認)
mysql> describe user;
          (user テーブルの属性を確認)
mysql> select host,user,password from user;
          (user テーブルからホスト名・ユーザ名・パスワードの欄を出力)
```

MySQL では、ユーザ名と接続ホストの組でパスワードを記憶する。ホスト名とユーザ名、ハッシュ化されたパスワードが表示されていることを確認する。

データベース `sales` の作成と、3つのテーブルの作成を行う。

```
# mysql -u root -p
mysql> create database sales;
mysql> use sales;
mysql> create table sales (
    -> date text,
    -> food text,
    -> amount integer );
mysql> create table price (
    -> food text,
    -> fee integer );
mysql> create table foodgroup (
    -> food text,
    -> foodgroup text );
mysql> exit
```

日本語コード変換、改行コード変換、1行目を削除した CSV ファイルをインポートする。

```
# mysql -u root --local-infile=1 -p
mysql> use sales;
mysql> load data local infile "変換後のuriage.csv" into table
sales fields terminated by ',';
mysql> load data local infile "変換後のtanka.csv" into table price
fields terminated by ',';
mysql> load data local infile "変換後のbunrui.csv" into table
foodgroup fields terminated by ',';
```

5.4.5 SQL 言語によるデータベース操作

テーブル内のデータ操作の例を下記に示す。フィールドとは、列の名前のこととを表し、例えば住所録なら「住所」や「姓」、「名」などにあたる。

データ挿入 フィールド 1 が値 1、フィールド 2 が値 2 となるデータを挿入する。

```
INSERT INTO テーブル名 (フィールド名 1, フィールド名 2) VALUES (値 1, 値 2);
```

レコード挿入 レコード（1行のデータ）を追加する。

```
INSERT INTO テーブル名 VALUES (値 1, 値 2, …, 値 n)
```

更新 フィールド 1 が値 1 のデータの、フィールド 2 を値 2 に、フィールド 3 を値 3 に変更する。

```
UPDATE テーブル名 SET フィールド名 2=値 2, フィールド名 3=値 3 WHERE フィールド名 1=値 1;
```

削除 フィールド 1 が値 1 のデータを削除する。

```
DELETE FROM テーブル名 WHERE フィールド名 1=値 1;
```

取り出し (A) フィールド 1 が値 1 のデータの全てのフィールドを取り出す。

```
SELECT * FROM テーブル名 WHERE フィールド名 1=値 1;
```

(B) フィールド 1 が値 1 でかつフィールド 2 が値 2 のデータのフィールド 1 とフィールド 3 の値を取り出す。

```
SELECT フィールド名 1, フィールド名 3 FROM テーブル名 WHERE フィールド名 1=値 1 and フィールド名 2=値 2
```

(C) テーブル 1 とテーブル 2 からテーブル 1 のフィールド 1 とテーブル 2 のフィールド 1 が対応しており、テーブル 1 のフィールド 2 が値 2 で、かつテーブル 2 のフィールド 3 が値 3 のデータについて、テーブル 1 のフィールド 1 とテーブル 2 のフィールド 3 の値を取り出す。

```
SELECT テーブル名 1. フィールド名 1, テーブル名 2. フィールド名 3 FROM テーブル名 1, テーブル名 2 WHERE テーブル名 1. フィールド名 1=テーブル名 2. フィールド名 1 and テーブル名 1. フィールド名 2=値 2 and テーブル名 2. フィールド名 3=値 3
```

文字列を含むキーは、下記のように、シングルクオーテーションで囲む。

```
mysql> delete from sales.sales where sales.date='2016/12/31' and  
sales.food='コカコーラ';
```

数値はクオーテーションで囲まない。

```
mysql> delete from sales.sales where sales.price=100;  
# 100 のもの  
mysql> delete from sales.sales where sales.price<>100;  
# 100 円ではないもの
```

5.5 実験内容 (2)

動的コンテンツ

動的コンテンツを生成するための設定と動的コンテンツの配置を行う。

- Apache にて SSI, CGI が実行できる環境の設定
- 簡単な動的コンテンツの作成
- PHP の動作環境をサーバに作成

簡単な動的なコンテンツの作成として、時刻を表示するプログラムをそれぞれ下記の技術を用いて作成し、それぞれの技術的違いを考えてみよ。

- SSI
- CGI (用いる言語は何でも良いが、本テキストではシェルスクリプトを提供)
- PHP

次に、CGI でカウンタプログラムを作成せよ。カウンタプログラムはユーザからリクエストを受けるたびに、カウンタの値を 1 つずつ増加させ、そのウェブページに何回のリクエストがあったかを表示するものである。プログラムやカウンタを記録するデータファイルの書き込み権限に注意すること。

Web システムのインストール 1

CGI, PHP を用いた掲示板 (Minibbs), Wiki (PukiWiki) のインストールを行う。

Web システムのインストール 2

MediaWiki のインストールを行う。

Web システムのインストール 3

WordPress のインストールを行う。

5.6 必要となる知識

5.6.1 SHELL の環境変数による言語設定

シェルの環境変数の設定は、C シェル系であれば setenv, B シェル系であれば export を用いる (Linux の標準シェル Bash は B シェル系)。

言語系の設定は、下記のように locale コマンドで用いられる変数を閲覧できる。

```
#locale
LANG=ja_JP.eucJP
...
LC_ALL=
```

LANG 変数が全ての言語系の挙動を決定するので、この変数を変更する。この変数に設定できる値は、locale -a コマンドで調べられる。

一般には、日本においては下記のようなものが用いられる。

C	英語
ja_JP.SJIS	日本語 (Shift JIS)
ja_JP.UTF-8	日本語 (UTF 8)
ja_JP.eucJP	日本語 (EUC JAPAN)

本実験では、日本語が表示できない端末も用いることがあるため、「C」にする。

5.6.2 PHP の MySQL 拡張のインストール・有効化

Wordpress や MediaWiki 等の LAMP 系 Web システムは PHP から MySQL を操作し、DB 上にデータを保存するため、PHP の MySQL 連携機能をインストールする

```
# apt install php-mysql

PHP 機能追加・変更後には、Apache 再起動が必要

# systemctl restart apache2
```

更に、MediaWiki をインストールするには、XML およびマルチバイト文字列（漢字等）が必要であり、下記もインストールする。

```
apt install php-xml
apt install php-mbstring
```

5.6.3 MediaWiki のインストール

MediaWiki のアーカイブを取得し展開する。

MySQL 上で MediaWiki 用データベースを作成する。

```
mysql> create database mediawiki;
mysql> create user 'mediawiki'@'localhost' identified by 'root00';
mysql> grant all on mediawiki.* to 'mediawiki'@'localhost';
```

この例では、MySQL 上にて、MediaWiki 用に、

データベース名 mediawiki

データベース上の接続用 MySQL ユーザ名 mediawiki

接続元 localhost (同一サーバの Apache/PHP からのデータベース操作のみ許可)

接続用パスワード いつもの

許可するデータベース操作内容 localhost からの mediawiki ユーザに mediawiki データベースへの全ての操作を許可

の内容でデータベースを作成している。

展開したディレクトリの名前を mediawiki に変更し、Web サーバの公開ディレクトリにディレクトリ内容を全て移動して、owner を全て WWW サーバのものに変更し、その後の設定は、ブラウザから接続を行う。

最後に生成されたコンフィグ用ファイル LocalSettings.php を mediawiki のディレクトリに移動する（権限も他ファイルと同じになるよう確認）。

MediaWiki の設定において、下記のように設定すること。

```
\end{cli}
wikiname: HP の名称どのような名前でも良い。グループ名が分かるもの。
mail: グループのマーリングリスト用メールアドレス
Language: 希望の言語
Admin user: root (mediawiki 管理用ユーザ)
    MediaWiki 管理用ユーザの
    パスワード : root00
Database: mediawiki
    MySQL 上でのデータベース名
DB username: mediawiki
    + パスワード
    MySQL でのユーザ名/パスワード
```

MediaWiki での補足

ブラウザから設定後、LocalSettings.php をサーバにアップロードするよう言われるので、SSH のファイル転送機能である SFTP, SCP を用いてアップロードする。root 権限ではアップロードできないので、一般ユーザで転送した後、サーバ側の root ユーザで、MediaWiki のディレクトリに移動すること。

Mac, Linux の場合

```
$ scp LocalSettings.php user@xxx.yyy.zzz.www:  
(user はサーバのユーザ名, xxx.yyy.zzz.www はサーバの IP)
```

これで、user のホームディレクトリに転送する。

Windows で Putty の pscp コマンドを使う場合

```
putty のあるディレクトリに LocalSettings.php を移動  
コマンドプロンプトを開き、putty のディレクトリに移動  
> pscp LocalSettings.php user@xxx.yyy.zzz.www:  
(user, xxx.yyy.zzz.www は scp と同様)
```

その他、Windows では、WinSCP というソフトウェアを用いることで、GUI でアップロードできる。

この後、サーバ上で、mediawiki のディレクトリに移動させておく。

5.6.4 WordPress のインストール

WordPress のアーカイブを取得して展開した後、展開されたディレクトリを、公開ディレクトリの下に、wordpress という名前で移動する。

ja.wordpress.org から Wordpress の PHP ファイル全体をダウンロードする。

```
$ wget https://ja.wordpress.org/latest-ja.tar.gz
```

```
# sudo su
```

ファイルを www 公開ディレクトリに移動する

```
# mv latest-ja.tar.gz /var/www/html
```

tar.gz 形式は、tar コマンドで解凍・展開する

まず展開する前に中身を確認する

```
# tar tvzf latest-ja.tar.gz
```

wordpress ディレクトリ以下に様々なファイルがあることを確認

問題無ければ展開

```
# tar xvzf latest-ja.tar.gz
```

wordpress ディレクトリがあることを確認

```
root 権限では正常に動作しないため、権限を Apache プロセスのものにする
```

Apache プロセスを確認

```
# ps auxww | grep apache
```

プロセスが動作するユーザを確認し、ディレクトリ以下の全てのファイルのオーナを変更
(-R は recursive 再帰的に全てのファイルを変更)

```
# chown -R オーナ(ユーザ名) wordpress
```

以上でブラウザから、wordpress ディレクトリに接続することができる。

```
https://www.gX.exp.info.kochi-tech.ac.jp/wordpress
```

ただし、下記データベース設定を行わないと設定は進められない。

このディレクトリおよびディレクトリ中のすべてのファイル・サブディレクトリの owner を Apache の実行権限に変更しておく。

MySQL 上にて、データベース wordpress を作成し、wordpress を操作する MySQL のユーザ wordpress も作成し、データベース wordpress への操作権限を grant 文で付与する。

```
# mysql -u root -p

mysql> create database wordpress;

mysql> create user 'wordpress'@'localhost' identified by 'root00';

mysql> grant all on wordpress.* to 'wordpress'@'localhost';
```

Web ブラウザで http://<Web サーバのホスト名>/wordpress/ に接続し、設定を行う。

WordPress の設定において、各種設定内容は下記のようとする。

データベース名：上での設定通り

データベース接続ユーザ名・パスワード

→ 上での設定の通り

テーブルの接頭辞 → そのままで OK

Wordpress のユーザ名：Wordpress 管理用ユーザ root

パスワード：(いつもの)

5.7 デバッグ

下記の点に注意する。

書き込み権限 CGI, PHP 等のプログラムは、Apache の動作権限で実行されますので、ファイルの書き込み等は、その権限で書き込みできる必要がある (/etc/apache2/envvars にある環境変数 APACHE_RUN_USER, APACHE_RUN_GROUP に権限が示されている)。

エラー対処 Internal Server Error が出力された場合は、動的コンテンツでの実行エラーです。ログファイル //var/log/apache2/ 以下のファイルを参照したり、コマンドラインからその CGI を実行するなどして、デバッグする。

5.8 動作確認

データベース

MySQL が正しくインストール、設定されデータベースの操作が行えるか確認する。

動的コンテンツ

アクセスを行うごとに、CGI で構築されたカウントが増加するか、SSI, CGI, PHP 全てで時刻表示が正常に行われるか確認する。

掲示板・wiki の閲覧や編集が行えるか確認する。

Web システム

Web システム用のデータベースが正しく構築できているか確認する。構築した Web システムについて、それぞれ閲覧、更新、修正などの作業が正しく行われるか確認する。

5.9 考慮すべき点

データベース

- MySQL などのリレーショナルデータベースを用いる利点。例えば、CSV ファイル等の表計算形式のファイルにデータを格納し、適宜プログラムから操作することに対する比較。
- リレーショナルデータベースのリレーショナル（関係）の部分について、他のデータベースと比較してどのような点が特徴的であるかを考える。

Web システム

- バックエンドとしてデータベースを用いる場合の利点が何かを考える。
- Web システムは、HTTP サーバ、動的コンテンツ (PHP)、データベース (SQL) を組み合わせて構築するが、単なる動的コンテンツとデータベースを用いる Web システムとの比較・考察を行う。
- CMS として、Wiki, Blog という、Web システムで代表的なものの、それぞれの特徴を考察する。合わせて、SNS 等、その他の Web システムの例をいろいろ調査、考察するのも良い。
- Web アプリケーション、SaaS、クラウドなどのキーワードについて例を挙げながら、それらを実現する要素技術の関連について考える。また、将来どのような Web アプリケーションが考えられるかを考察するとよい。

5.10 実験内容(2')

5.10.1 HTTPS サーバの設定～認証局による証明書の署名(これ以降は実験4Cのみ)

Ubuntu 標準手順での TLS 対応設定では、サーバ証明書およびその秘密鍵として、サーバ名などが正しく設定されていない self-signed (自己署名) 証明書が使われている。

詳しくは、Ubuntu の SSL/TLS 設定が書かれている下記の設定ファイルを参照する。

- /etc/apache2/sites-enabled/default-ssl.conf

このファイルの内容は、大まかに下記の意味を持っている。

```
<IfModule mod_ssl.c>
```

→ Apache2 用 SSL 機能モジュールがある場合に以下を設定

```
VirtualHost _default_:443
```

→ 仮想ホスト(バーチャルホスト)機能で、任意の IP アドレスのポート 443 でアクセスされた場合の設定(それ以外、例えば、80 番や、特に設定した別の IP アドレスなどでは以下の設定は有効にならない)

```
ServerAdmin サーバ管理者の連絡先メールアドレス
```

```
DocumentRoot /var/www/html
```

→ コンテンツ(トップページ)の場所

```
ErrorLog, CustomLog
```

→ エラーログの場所

```
SSLEngine on
```

→ SSL(TLS)機能の有効化

```
SSLCertificateFile
```

→ SSL 証明書(公開鍵と鍵の持ち主(身元)の情報)ファイルのパス。Ubuntu デフォルトでは、/etc/ssl/certs/ssl-cert-snakeoil.pem となっている。

```
SSLCertificateKeyFile
```

→ SSL 証明書の秘密鍵ファイルのパス。Ubuntu デフォルトでは、/etc/ssl/private/ssl-cert-snakeoil.key となっている。

SSLCertificateFile のコメントに書かれているように、デフォルトは乱数で生成された /etc/ssl/certs/ssl-cert-snakeoil.pem が証明書として、/etc/ssl/private/ssl-cert-snakeoil.key がその秘密鍵として指定されている。snakeoil は、いんちき薬の意であるが、サーバの身元を保証しない適当な証明書である。これを自己署名(self-signed)証明書と言うが、俗にオレオレ証明書と呼ぶ。

このままで、経路上の第三者が通信内容を傍受することが不可能な、暗号化された Web ページの構築はできており、研究室内などのごく小規模でインターネットへの公開サービスを必要としない場合は、この形での運用も可能である。しかし、公開サービスとする場合、第三者による偽のサイトと、本物のサイトとの見分けが付かず、フィッシングや偽情報発信等の詐称の問題が発生し得る問題がある。そのため、OpenSSL を用いて正式な証明書・秘密鍵を作成して運用する。その手順は以下の通りである。

- (1) OpenSSL を使い、秘密鍵と公開鍵（証明書署名要求（CSR⁶）=署名前の証明書）を作成
- (2) 証明書署名要求ファイルに署名を行い、証明書ファイル（CRT⁷）を作成
- (3) 秘密鍵・証明書を配置し、Apache 設定ファイル `default-ssl.conf` の `SSLCertificateFile`, `SSLCertificateKeyFile` で両ファイルを指定

作成した証明書は、然るべき認証機関⁸にて、所定の費用を支払うなどして署名される必要がある。この方法が通常の機関で行われる運用である。サーバを複数台運用する場合は、中間認証局を設置しその認証局証明書が上位認証局やルート認証局にて署名された後、各サーバの証明書を中間認証局で署名する。次に 2 つ目の方法として、大学などの小規模機関では、独自のルート認証局を設置し機関内のクライアントに安全な手段でルート証明書を配布・インストールして運用する場合もある。ここでは、この方法を用いることとし、サーバで認証局を設置し、認証局の証明書をルート証明書としてクライアントにインストールし、サーバ証明書を認証局で署名して用いる。

5.10.2 Apache のサーバ証明書の作成

OpenSSL のコマンドを使う。情報としては、下記が分かりやすい。

- Ubuntu のサーバ運用マニュアル

<https://ubuntu.com/server/docs/security-certificates>

コマンド `openssl genrsa` による秘密鍵生成の際、`-des3` などによる秘密鍵を暗号化して保存するオプションは通常外しておく運用が多いため、ここでもそのようにする⁹。

```
# openssl genrsa -out server.key 2048
-> RSA アルゴリズムでの秘密鍵の作成。ファイル名は server.key 鍵長は 2048 ビット

# openssl req -new -key server.key -sha256 -out server.csr
-> 秘密鍵 server.key に対応する証明書署名要求（CSR）を作成
証明書の内容の入力は次の通りとすること。
署名アルゴリズムには SHA256 を使うこと。
これは、工科大の認証局の証明書ファイルの要求仕様による
```

Country Name:

⁶ Certificate Signing Request

⁷ Certificate

⁸ ルート CA あるいはルート認証局と呼ぶ。VeriSign 社が有名。日本法人は、「合同会社シマンテック・ウェブサイトセキュリティ」。他にも GMO などが証明書発行業務を行っている

⁹ サーバの管理者ユーザがだれでも、秘密鍵を簡単に読み込み、コピーなどができると、セキュリティ上良くないため、鍵ファイルにも暗号化を行い、パスワードを入力しないと読み込めないようにする設定だが、サーバ証明書の場合、Apache を再起動するたび、パスワードを入れる必要があり、サーバの自動起動などにおいて運用上の問題が起こる。そのため、サーバにログインアクセスできる人を最小限に絞って管理することで、サーバ秘密鍵へのパスワードは省略する運用も行われる。

→ JP

State or Province Name:

→ Kochi

Locality Name:

→ Kochi-shi

Organization Name:

→ Kochi Prefectural Public University Corporation

Organization Unit Name:

→ 無

(入れる場合は大学に要事前申請)

Common Name:

→ サーバの FQDN

www.gX.exp.info.kochi-tech.ac.jp

(X: グループ名)

(ブラウザからアクセスされる URL ホスト名)

Email Address:

→ ピリオド「.」のみ入力し Enter

(サポート外なので入力キャンセル)

extra 項目は、何も入れなくて良い。

工科大の認証局は、ルート認証局 (CA) ではなく中間認証局 (CA) である。ルート認証局は文部科学省所管の国立情報学研究所 (NII¹⁰) である。

NII の CA による証明書作成については、下記の情報も参考にする。

<https://meatwiki.nii.ac.jp/confluence/pages/viewpage.action?pageId=26188809>

5.10.3 証明書の作成

正しく署名された証明書を作成するため、証明書署名要求 server.csr ファイルを、TA に渡す。

後日、上位 CA (情報システム課) にて署名した証明書を各班に返すので、それを配置すること。

それまでの間、次の自己署名証明書を作成して使う。

```
# openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

signkey にて自分の秘密鍵を指定するため、自己署名となる。

有効期間は、作成時から 365 日である。

server.crt が証明書ファイルである。

¹⁰ National Institute of Informatics

5.10.4 証明書ファイルの配置

Apache にて秘密鍵・証明書を設定して再起動すれば設定は終了である。

```
# mv server.key /etc/ssl/private/
# mv server.crt /etc/ssl/certs/
(証明書ファイル名は各自確認し、適宜読み替えること)

# vi /etc/apache2/sites-available/default-ssl.conf
SSLCertificateFile      /etc/ssl/certs/server.crt
SSLCertificateKeyFile   /etc/ssl/private/server.key
```

秘密鍵・証明書の owner , パーミッションとも、元からあった `snakeoil` に合わせておくこと。

Apache を再起動する。

工科大から、下記 2 ファイルをもらったら、下記の証明書登録、中間 CA の証明書登録をする。

- 証明書ファイル (crt)
- kut.ca (中間証明書ファイル)

```
# mv server.crt /etc/ssl/certs/
新しい証明書を置く

# mv kut.crt /etc/ssl/certs/
工科大の中間証明書を置く

# vi /etc/apache2/sites-available/default-ssl.conf
SSLCertificateFile      /etc/ssl/certs/server.crt

SSLCertificateChainFile /etc/ssl/certs/kut.crt
→ こちらは無ければ追加（証明書チェーン）
```

Apache 再起動

5.10.5 注意点

証明書は、鍵長やアルゴリズム、ホスト名やパラメータの値など、様々な制約があり、少しでも合致しない証明書は、不正な証明書として扱われ、ブラウザでエラーが表示される。

その他、有効期間も重要であるので、下記に注意する。

- 有効期限を過ぎた証明書は不正。
- 長期のサーバ運用では、証明書の定期的なアップデートに注意する。

- 有効期限前の証明書は不正。
サーバの時計の設定には注意する。

第 6 章

ネットワークセキュリティ

ネットワークの境界において、必要のない通信を遮断するファイアウォールやパケットフィルタと呼ばれる機能を有効にすることは重要である。また、それらによって外部の機器への通信を制限される内部の機器のために NAT という機能による通信の仲介を行わせることがある。NAT は内部から外部に対する通信を制御するだけでなく、内部ネットワークのセキュリティ確保にも役立つ。

6.1 ネットワークセキュリティ

これまで、TCP/IP によるネットワークの構築を行い、多くのネットワークサービスを立ち上げ、利用者の利便性の向上につながるネットワークシステムの構築を行ってきた。

ルーティングを行い TCP/IP の接続性を向上させ、DNS、Web、電子メールシステムを構築することで、インターネットを通して世界中から接続可能なネットワークサービスの構築を行った。同時に、ファイル転送・ファイル共有などの仕組みを構築し、情報のやりとりの利便性も向上した。

これまでのネットワークサービスの構築においても、端末ごとの OS のアップデートやウイルス対策、サービスごとのアクセス制限など、個々のコンピュータやサービス単位でのセキュリティの設定を行ってきた。

大規模ネットワークの管理を行う場合でも、全ての端末に対して常に万全のセキュリティを求めるることは重要であるが、セキュリティホール 1 つからネットワークの端末全体の危険が誘発されるような事態も起こり得ることや、多重保護の観点からもネットワーク単位でのセキュリティも重要なとなる。

このような目的でネットワーク全体の保護に用いられるセキュリティ技術として、ファイアウォール（防火壁）がある。ファイアウォールは、ネットワークの入口で、セキュリティ上好ましくない通信や、不必要的通信を遮断し、アクセス制限を行う。ファイアウォールには、ネットワーク外部からの攻撃などを遮断する意図の他、ネットワーク内部からの外部へ向けての攻撃などを防ぐ意図などもある。具体的には、外部からの攻撃を誘発したり助けたりするような内部から外部へ向けてのアクセスや、ウイルスなどに感染した内部ホストからの外部への攻撃をネットワークの入口で防ぐ意図がある。

6.1.1 ファイアウォール

ファイアウォールは、ネットワークのアクセスを選別し必要のあるアクセスのみを通過させ、そうでないアクセスを遮断する技術である。

ファイアウォールを用いることによりネットワーク・セキュリティを向上させることができる。しかし、ファイアウォールを構築するだけですべての攻撃を防ぐことはできない。例えば、ファイアウォールではメールに添付されているコンピュータウイルスを除去することはできない。また、Web サーバに対して行なわれる正常な http アクセスを装った DDoS 攻撃¹には対処できない。そして、新しい脅威に対しては、

¹ distributed denial of service の略。複数の拠点から標的のサーバへ一斉に DoS 攻撃を仕掛ける。

防御できない場合も十分に考えられる。そのため、ファイアウォールの運用においては必ず他のセキュリティ技術と組み合わせることが必要である。

ファイアウォールには、大きく分けて次の二種類がある。これは、パケット処理を行うネットワークレイヤーによる違いである。

パケットフィルタリングファイアウォール

図6.1に示すように、内部ネットワークと外部ネットワークの境界に位置するルータにおいて、パケットフィルタリングルールに従って、二つのネットワーク間でのパケット転送を制御する。宛先IPアドレス・送信元IPアドレス、プロトコルの種類やポート番号等、IP・TCP・UDPヘッダの情報に基づく制御を行う。パケット毎に処理を行い、データの塊や接続状態（セッションの状態）などは基本的に関知しない。ただし、近年の高機能ファイアウォール専用製品ではステートフルインスペクションと呼ばれるステート（＝TCPの状態、Establishedか否かやSyn, Finなども条件に加える）を加味したパケットフィルタや、HTTPやSMTPのコンテンツ、ウイルス、攻撃検出もある程度も行うディープパケットインスペクションなども一般的になってきている。このような機器では、処理が複雑になり状態を記憶する必要もあり、性能を出すためにはメモリや処理性能が求められる（遅くとも良ければ、サーバでも可能）。

アプリケーションゲートウェイファイアウォール

図6.2に示すように、アプリケーション層において、定められたルールに従ってパケットを中継する、あるいは拒否するファイアウォールである。このため、アプリケーションごとの制御や、データ単位での制御、ユーザ単位での制御が行え、パケットフィルタリングよりもきめ細かい制御が可能となる。実際の接続を、このゲートウェイが代行することになるので、プロキシ（Proxy：代理）とも言う。

DNSのキャッシュサービス、電子メールのSMTPサービスも、クライアントの要求をサーバが代行して行うという点で、一種のアプリケーションゲートウェイである。

近年では、パケットフィルタとアプリケーションゲートウェイを組み合わせるパターンが多くなっている²。ここでは、パケットフィルタで内側（trust側）と外側（untrust側）の通信を完全に遮断し、サーバだけは外部との必要な通信を許可し、内側のPCからは、アプリケーションゲートウェイやメールサービス・DNSサービスを経由して外側のサービスを受信できるようにする。

このように、LANの一部を完全に外部から遮断する形は、安全性が高いが、利便性の面で利用者が不便になる点もある。

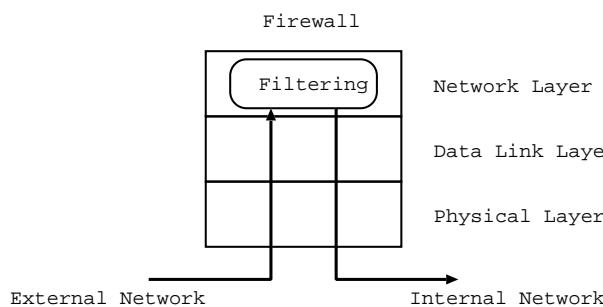


図6.1 パケットフィルタリング

² 実際には、内側（trust側）、外側（untrust側）をパケットフィルタで分離し、さらに、内側と外側の両方からアクセス可能なDMZ（非武装地帯）を作成して、アプリケーションゲートウェイとなるサーバをここに設置する。

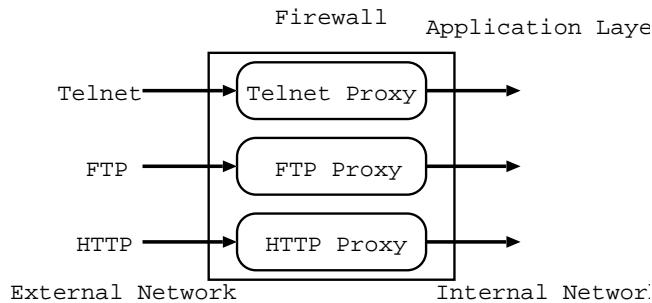


図 6.2 アプリケーションゲートウェイ

6.1.2 NAT

Network Address Translation (NAT: ネットワークアドレス変換) は、IP パケットの送信元アドレス、宛先アドレスの一方または双方を書き換えて転送するものである。狭い意味で NAT と言う場合、IP アドレスのみの書き換えを指すことがある。

UDP データグラム、TCP セグメントの送信元ポート番号、宛先ポート番号も同時に変換する場合もあり、これを、NAPT(Network Address Port Translation)、PAT (Port Address Translation: Cisco 社が主に呼称)、IP Masquerade (IP マスカレード: Linux システムが主に呼称) と呼ぶ。あるいは、NAPT を単に NAT と呼称する場合も多く、むしろ現在は、アドレス加えてポート番号も変換する場合がほとんどであるので、NAT = NAPT と考えてもほとんど差し支えない。

内側から外側ネットワークへの接続に NAT を用いると送信元 IP アドレスなどのネットワーク情報が置き換わるため、下記のような利点があり、よく利用される。

- 1 つの IP アドレスで複数のコンピュータをインターネットへ接続する。
- 外側から内側への接続を制限し、かつ、内側のネットワーク情報を隠蔽することで、簡易的なファイアウォールとしてセキュリティを向上させる。
- NAT の外側からは、1 台のコンピュータとして見せることができるために、ある LAN をルーティングプロトコルやルーティング設定を行うことなく、別の LAN に接続することができる。家庭内 LAN のインターネットへの接続は、この例である。

なお、家庭用のインターネット接続用ブロードバンドルータや無線 LAN ルータと呼ばれる製品は、NAPT を行うものである（静的ルーティングや RIP による L3 ルーティングは一部サポートされているが、ほとんど用いることはない）。このため、ネットワーク技術とは異なる場面で、単にルータといった場合に、NAPT を指すこともしばしばあるので注意する（ネットワーク技術に関する話を行う場では、NAPT の意味でルータの語を用いることはまれである）。

NATの種類

スタティック NAT

狭い意味での NAT であり、IP アドレスを単順に置き換える。置き換える前の IP アドレスと、置き換える後の IP アドレスが一対一に対応し、常にその対応通りに置き換わる場合をスタティック(静的)NAT と呼ぶ。内側からも外側からも、また、すべてのポート番号の UDP/TCP、あるいはそれ以外のすべての IP パケットについて変換を行い、常にパケットを転送することができるので、サーバを開する際などに用いることができる。欠点として、セキュリティは、NAT を行わずに直接外側ネットワークに接続する場合と変わらないことから、十分なセキュリティ対策を行わないと NAT 内側ネットワークが危険になることや、外側の IP アドレスを常に 1 台につき 1 つずつ必要であることが挙げられる。すなわち対応づけられたコンピュータ以外は、その IP アドレスを用いることはできない。

スタティック NAPT (PAT)

置き換える前の IP アドレスおよびポート番号と、置き換える後の IP アドレスおよびポート番号の対を、あらかじめルータに静的に設定しておくものである。すなわち、NAT テーブルの変換エントリをあらかじめ作成しておく。事項で説明するオーバーロードあり動的 NAT は、内側から外側へのアクセスがあると、その時に動的に NAT エントリが NAT テーブルに作成されるが、静的 NAT ではあらかじめ作成しておくため、外側から内側へのアクセス要求にも、変換および転送ができる。このため、この後に説明するポートフォワーディング(ポート転送)に用いられ、サーバ公開(NAT 内部に置いたサーバを外部からアクセス可能にする)に用いる。設定したポート以外は転送されないため、サーバに公開ポート以外のパケットフィルタを設定したのにセキュリティ効果もある。

NAPT(ダイナミック、オーバーロード)

PAT、マスカレーディング、オーバーロードあり動的 NAT などとも呼ばれ、一つの置き換える後の IP アドレス(通常、外側の IP アドレス)に対し、複数の置き換え前の IP アドレス(内側の端末の IP アドレス)を対応づけることができる。これは、ポート番号(ソース=内側)を変えることで実現するもので、多くの TCP アプリケーションは、問題なく動作する。利点として、多くのコンピュータに対して必要なインターネット通信用のパブリック IP アドレスが 1 つで済むため、IP アドレスを節約できること、原理的に外側からの接続は、そのままでは内側に転送できないため、簡易的にファイアウォールとして用いることができるがあげられる。内側に公開サーバを置く場合は、別途、ポートフォワーディングを行う必要がある。

NAT テーブル

IP アドレス変換の対応表は、NAT を行うルータのメモリに記録され、外側から内側宛のパケットを受信した際に参照され、テーブルに従って内側に転送される。NAT テーブルに対応するものが無い場合は、パケットは破棄される。

Cisco IOS であれば、下記のコマンドで NAT テーブルを確認できる。

```
Router> show ip nat translations
```

NAT テーブルは、通常内側から外側へのアクセスがあった場合にテーブルに追加され、その後外側から返って来る返信パケットは、テーブルのエントリに従って転送される。

一度作成したエントリは、static なものは永遠に、動的なものも数分は保持されるため、消去したい場合は、下記のコマンドでエントリを全て削除する。これを行うと、通信中の TCP セッションは強制切断され

る。すなわち SSH やリモートログインなどは、サーバに端末やシェルなどのプロセスを残したまま強制切断され、POP や SMTP 通信中であれば、メールの送受信に異常が起こる。

```
Router> en
Router# clear ip nat translation *
```

6.1.3 ポートフォワーディング

NAPT を行うと、内側から発信されたパケットの返信ではなく、外側ネットワークから直接外側 IP アドレス宛に行われる通信は、NAT テーブルにエントリがないため、転送されない。すなわち、公開サーバを内側に置くことができない。そこで、例えば、HTTP サーバを公開する際の TCP 80 番ポート宛など、あらかじめ決めておいたポート宛の通信は、NAT テーブルにエントリとして手動で追加しておくことができる。これを、ポートフォワーディング（ポート転送）と呼び、HTTP サーバやメールサーバなどを、内側ネットワークに置くことができる。

ポート単位で転送の可否を決めるため、それ以外のポート宛の通信はルータで破棄される。このため、簡易的なファイアウォールとして用いることができる。

6.2 実験内容(1)

パケットフィルタの設定

LAN の入口のルータにてパケットフィルタを行う。

ルータによるアクセスリストを用いセキュリティーレベルを向上させる場合、先にネットワークのセキュリティに関するポリシーを策定する必要がある。ここでは以下のようなセキュリティーポリシーとする..

- サーバ以外のコンピュータ：内部→外部の通信は NAT(NAPT, PAT) で許可
- サーバ以外のコンピュータ：外部→内部の通信は不許可
- ただし、Ubuntu Linux Desktop への外部からの SSH 接続は、ポートフォワーディング（静的 NAT）で許可。このとき、ルータの外部 IP アドレスへの TCP ポート 8022 への接続を、Linux Desktop の TCP ポート 22 (SSH) へ転送するようにする。
- サーバ：外部ネットワークとサーバ間で、DNS、メール (MTA 同士) 送受信、Web 通信の送受信を行えるようにする。
- サーバから外部へは、任意の TCP 通信、DNS 通信を可能とする。

本実験でのネットワーク構成を図 6.3 に示す。図 6.3 の 172.21.Y.S, 172.21.Y.L, 172.21.Y.W, はそれぞれ、各グループのサーバの IP アドレス、Linux の IP アドレス、Windows の IP アドレスである。

6.3 必要となる知識

6.3.1 IOS によるパケットフィルタリング

Cisco IOS アクセスコントロールリスト

Cisco 社製ルータに搭載されている IOS には、アクセスコントロールリスト (Access Control List = ACL, 単にアクセスリストとも呼ぶ) と呼ばれるアクセス制限を行う機能があり、この ACL を用いてパケットフィルタリングを行う。

アクセスリスト

アクセスリストには、標準アクセスリストと拡張アクセスリストの 2 種類が存在する。標準アクセスリストでは、送信元 IP アドレスのみを見てパケット通過の可否を判断する。これに対して、拡張アクセスリストでは下記に示すようにヘッダの様々な情報を用いて可否を判断する³。

³ なお、宛先 IP アドレスのみでパケットフィルタを行う場合は、ルーティングを用いれば良い。パケット破棄を行いたい宛先ネットワークに対して、Next hop インターフェースとして、Null0 インターフェースを指定した経路を作成することで、パケットが破棄される

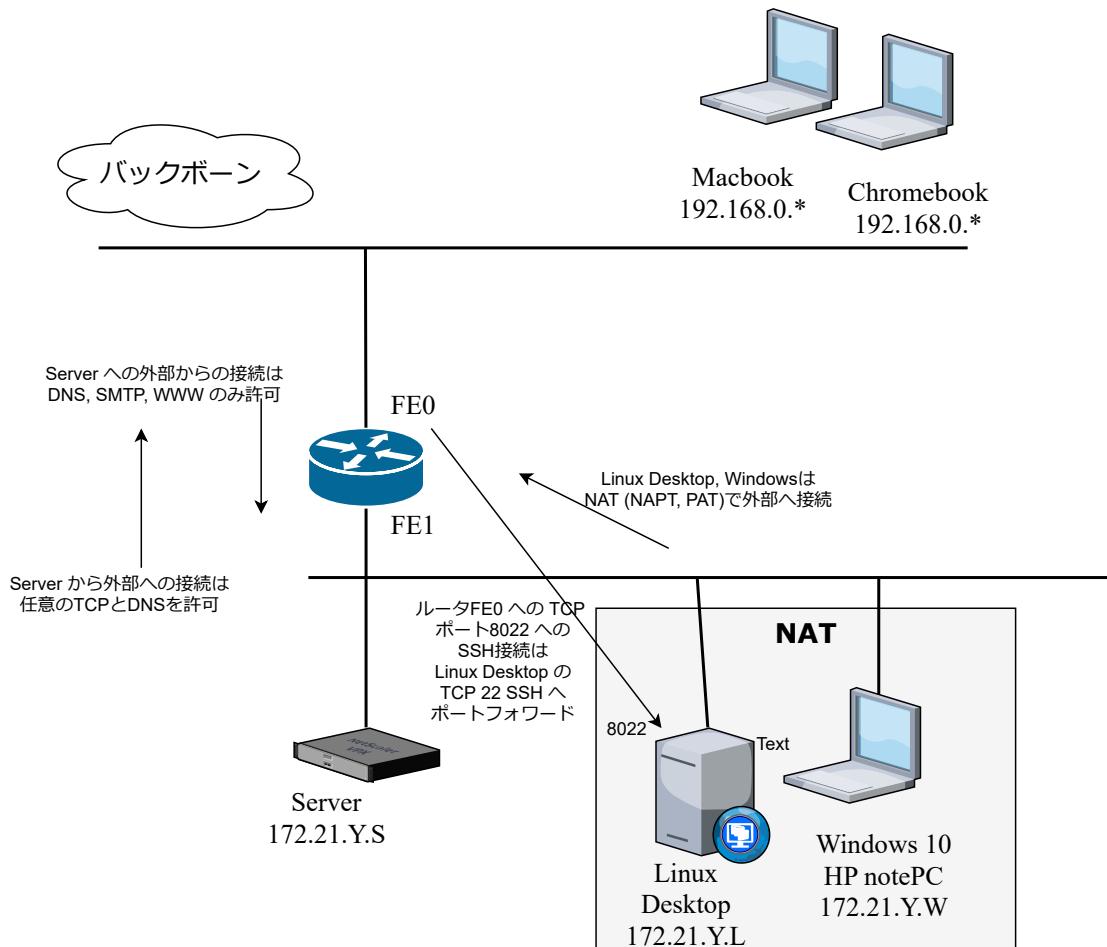


図 6.3 ネットワークの構成図

- 送信元 IP アドレス
- 宛先 IP アドレス
- TCP, UDP, ICMP, GRE, その他の IP プロトコルの種類
- TCP や UDP の送信元・宛先ポート番号
- 送信元・宛先 MAC アドレス
- ICMP タイプ・コード
- TCP のオプション

標準アクセリストでは、単純に送信元 IP アドレスのみを見れば良いような単純なルールに用いられる。具体的には、攻撃ホストとして認定されているようなインターネット上の IP アドレスからのアクセスを拒否する場合や、ウイルス感染が確認された内部ホストからのパケットを拒否する場合などに用いる。拡張アクセリストは、サービス毎に細かい処理をする必要がある場合に使用する。

アクセリスト番号が 1-99 までのものは標準アクセリスト、100-199 までは拡張アクセリストとして認識される。ルータは通常、外向きのインターフェースと内向きのインターフェースがあり、外部から内部へのパケットの遮断を行う場合は外向き側のインターフェースにアクセリストを、内部から外部へのパ

ケットの遮断を行う場合は内向き側のインターフェースにアクセリストを用いる。どちらのアクセリストの場合でも、アクセリストを作成した場合に最後に全て拒否するという暗黙の設定が入るので、基本的には必要なものを許可していくという方針でアクセリストを作成する。

拡張アクセリストの書式と設定方法

[拡張 ACL 書式]

```
access-list リスト番号 アクション プロトコル
(続き) 送信元 IP アドレス 送信元ワイルドカードマスク [eq 送信元ポート番号]
(続き) 宛先 IP アドレス 宛先ワイルドカードマスク [eq 宛先ポート番号]
(続き) [established] [icmp type] [icmp code]
```

- ブラケット [] 内は省略することができる。
- リスト番号は、ひとまとめのアクセリストに対して同じ番号（100 199）を付与する（199: 標準 ACL ⇒ 送信元 IP のみで判断, 100 199: 拡張 ACL ⇒ IP・TCP・UDP ヘッダ）
- アクション → permit (許可) or deny(拒否)
- プロトコル → ip(すべての IP パケット), tcp, udp, icmp
- ポート番号 → UDP・TCP の時のみ有効。ポート番号を指定。省略した場合ポート番号の検査は行わない（すべてのポート番号が該当する）。
- established → TCP の時のみ有効。ACK フラグのあるパケットを照合。
- icmp type → ICMP の時のみ有効。ICMP タイプ番号を指定。
- icmp code → ICMP の時のみ有効。ICMP コード番号を指定。

このようなパケット照合のパターンを複数行続けて書いたものがアクセリストである。パケットの照合はリストの上から順に行われ、最初にマッチしたパターンのアクションが実行され、この後のリストは評価されない。最後まで照合を行ってもマッチしなかったパケットは破棄される。これを、最後の行に deny の行があるような振舞いであることから、暗黙の deny と呼ぶ。

送信元 IP アドレス・宛先 IP アドレスは、ワイルドカードを設定することにより、複数のアドレスをまとめて指定することが可能である。ワイルドカードマスクは、IP アドレスを 32 ビットの 2 進数表記にした際、チェックを行うビットは 0、チェックを行わないビットは 1 とし、8 ビット毎にドットで区切った 10 進数 4 つで表す。

例えば、172.21.39.0 ? 172.21.39.255 を指定する場合は、172.21.39.0 0.0.0.255 とする。0.0.0.0 255.255.255.255 とした場合は、任意の IP アドレスを意味し、これは any と書いても良い。また、単一の IP アドレスのみを照合する場合、"IP アドレス 0.0.0.0" とする。これは、"host IP アドレス" と書いても良い。

次に、定義したリストを ip access-group コマンドでインターフェースに設定する。

```
interface fastethernet0
 ip access-group リスト番号 in(out)
```

この例は、リスト番号のアクセリストを、fastethernet0 からパケットが入って来る (in) 際（あるいは、出て行く際 (out)）評価する設定である。

アクセリストの設定

アクセリストの設定はルータにコンソール接続をして行う..

下記はグループ 12 の例

```
[[101 番のリストは、外側インターフェース向け]]
router12(config)# no access-list 101

router12(config)# access-list 101 permit tcp 0.0.0.0 255.255.255.255
サーバの IP 0.0.0.0 established

(↑ access-list 101 permit tcp any host サーバの IP established
と書いても良い)

router12(config)# access-list 101 permit tcp any host サーバ IP eq 53
router12(config)# access-list 101 permit udp any host サーバ IP eq 53
router12(config)# access-list 101 permit tcp any host サーバ IP eq 25
router12(config)# access-list 101 permit tcp any host サーバ IP eq 80
router12(config)# access-list 101 permit tcp any host サーバ IP eq 443

router12(config)# access-list 101 permit udp any eq 53 host サーバの IP

router12(config)# exit

router12# show access-lists
```

アクセリストは上から順にルールがチェックされ、最初に適合したルールが実行されて終了する（ソフトウェアの if-else if-else if ... 構造と同様）。そのため、順序も意味を持ってくる。

次に、ルータでインターフェースへの適用を行う。

```
router12(config)# interface FastEthernet0
router12(config-if)# ip access-group 101 in
    (外向けのインターフェース FastEthernet0 に外部から入ってくるパケットに対しアクセリスト 101 を適用する)
router12(config-if)# end
```

ここで、サーバや PC から、外部接続ができるかの状況を確認する。

アクセリストの消去は、下記のコマンドで行う。誤りがあれば再設定する。

```
router12(config)# no access-list XXX(消したいアクセリスト番号)
```

同様に、インターフェースに対するアクセスグループを消したい場合には以下のようにする。

```
router12(config-if)# no ip access-group XXX in
```

6.3.2 NAPT の設定

NAPT の設定は以下の作業をルータで行うことで実現できる。

- (1) プールの設定（インターネット側の通信に用いる IP アドレスの定義）
- (2) アクセスリストの設定（変換の対象となる内側の IP アドレスの定義）
- (3) NAT の定義（プール名とアクセスリスト番号と NAT の種類）
- (4) インタフェースの設定（どの IF が内側で、どの IF が外側か）

詳しくは、Cisco 社のマニュアルが参考になる。

<http://www.cisco.com/c/en/us/support/docs/ip/network-address-translation-nat/13772-12.html>

上記 URL の「Configuring NAT to Allow Internal Users to Access the Internet Using Overloading」を参考にする。

設定の内容の方針は下記のように考えれば良い。

- (1) プール名は自分達で決める。用いる IP アドレスは、ルータの外側 IP アドレス（1つのみ）。prefix はルータ外側 I/F のネットマスクの prefix を用いる。
- (2) アクセスリストの番号は、1 から 99 で自分達で決める（100 以降は拡張アクセスリストと呼ばれ、NAT では用いない）。Action は permit を用いる。
- (3) NAT の定義は、内側 LAN からのソース IP アドレスのみを変換するので、inside source を用いる。
- (4) NAPT (PAT) をしたので、overload（複数の内部 IP が同時に 1 つの Pool アドレスを使う）指定を行う。

確認は、下記 URI にアクセスすることで行う。

<http://192.168.0.1/index.cgi>

IP アドレスが NAT で設定したものになっていれば良い。

各グループの公開サーバへのポートフォワーディングは、上記 URI の「Example: Redirecting TCP Traffic to Another TCP Port or Address」を参考にする。

注意点として、サーバの IP アドレスが外部からどのように見えるかをよく考え、DNS 等の設定変更が必要であれば、正しく変更しておくこと。また、DNS の上位サーバ側などで変更が必要であれば、TA に申し出ること。

6.3.3 NAPT の設定例

```
### NAPT 設定。Windows と Linux Desktop を Router FEO
### のアドレスで外へ出られるようにする

### 変換対象の内側 IP アドレスをアクセスリストを使って定義

router12(config)# access-list 1 permit LinuxDesktop の IP  0.0.0.0
router12(config)# access-list 1 permit Windows の IP  0.0.0.0
```

```
### 変換後の外側 IP アドレスを Pool ("g12"=グループ 12 の
### IP アドレスのプール) で定義

router12(config)#ip nat pool g12 ルータ FE0 の IP ルータ FE0 の IP prefix-length 24

### NAT 変換ルールを定義

router12(config)#ip nat inside source list 1 pool g12 overload

### インターフェースに NAT の内側・外側を定義

router12(config)# int fa0
router12(config-if)# ip nat outside

router12(config)# int fa1
router12(config-if)# ip nat inside

### パケットフィルタ用アクセリスト（例では 101 番）に
### NAT 用の宛先を持ったパケットの許可

router12(config)# access-list 101 permit ip any host ルータ FE0 の IP
```

Windows, Linux から <http://192.168.0.1/index.cgi> に接続して IP を確認する。

6.3.4 ポートフォワーディングの設定例

```
### ルータの外部 (FE0) IP への TCP port 8022 接続を
### 内部 Linux Desktop IP の TCP port 22 へ転送する

router12(config)# ip nat inside source static tcp LinuxIP 22 RouteFE0_IP 8022
```

6.3.5 設定確認

下記で設定や状態を確認する

(設定)
show running-config

(インターフェース状態)
show ip interface brief

(アクセリスト)

```
show access-lists
```

(NAT テーブル)

```
show ip nat translations
```

(NAT テーブルを消去したい場合は、下記で消える。)

現在ユーザが接続中のセッションも全部切られる。)

```
router12# clear ip nat translation *
```

(ARP テーブル)

```
show ip arp
```

6.4 動作確認

パケットフィルタ

他グループ (TA) の端末からルータ、ハブ、サーバ等に ping を行い、通信の可否が期待通りの動作か確認する。また、自ネットワークから他のネットワークに対しても、通信の可否を確認する。

telnet コマンドで、他グループから、ポート接続の可否を確認する。

`telnet 相手 IP 相手ポート`

`connected` と表示されれば接続されている

エスケープ文字`^]` で接続終了できる

`(Control +])`

NAT (NAPT)

- 内側ネットワークの各端末から外側ネットワークのサーバへ WWW などのアクセスが行えること。
- 外側ネットワークの端末からは、ルータの IP アドレスからアクセスがあるよう見えること。
クライアントのブラウザから `http://192.168.0.1/` にアクセスし、”...check your IP, please go here” のリンクにアクセスし、表示される IP アドレスが期待通りか確認する。
- 外側ネットワークから、内部の公開サーバへはアクセスできるが、それ以外の端末・サービスへはアクセスできないこと。
- 外部から内部サーバへは、SSH はできないこと。

- 外部からルータ外部の TCP 8022 へ SSH をすると、Desktop Linux にログインできること。
- ルータ上で NAT テーブルに適切なエントリがあること。

6.5 考慮すべき点

ファイアウォール

パケットフィルタ・アプリケーションゲートウェイには、それぞれ処理速度・細かな制御という利点があるが、これらの他のタイプのファイアウォールを考えることはできるか。

NAT, NAPT

- ネットワーク分割として NAT を用いることが有用である場合とそうでない場合。
- 公開すべきサービスは何か、公開すべきでないサービスは何を考える。
- 様々な NAT 方式とその違いを、仕組みと目的・用途を対応づけて考える。

第 7 章

最終課題

最終課題では、AWS 上で Wordpress を運用できるネットワークサービスを構築する。

7.1 目的

クラウドサービス大手のアマゾン社が提供するクラウドサービス「Amazon Web Service」(AWS) を使った Wordpress による Web サービスの運用を行う。AWS では、数多くのネットワーク・Web サービスが提供されているが、そのうち、Elastic Computing Cloud (EC2) と呼ばれる仮想マシンサービスによりサーバの構築を行う。

クラウドサービスに対して、オンプレミス型のサービスがあり、これはハードウェアを設置してオペレーティングをインストールしネットワーク構築を全て行った上で必要なサービスを導入するものであるが、これらのものではハードウェアの性能は最初の導入時に決まり、後から性能を大きくしたい（スケールアップ、スケールアウト）場合でも、容易には行えない。また逆にコストの面から利用にあった性能にスケールダウンすることも難しい。クラウドサービスを使うことにより、ハードウェアを利用時間単位で、必要な性能を必要なだけ使用し、その分だけコストを支払うことができる。Elastic には柔軟で伸び縮みできるという意味があり、こうした利用形態が可能である。また、障害やアクセス集中などに対する耐性も、大手のクラウドを用いる方が対処しやすことも多く、こうしたことでもクラウドを利用する利点になる。

7.2 内容

Amazon Web Service 社の Elastic Computing Cloud にて、Linux, Apache2, MySQL, PHP を使った、SSL/TLS による Wordpress サービスを構築するために、下記を行う。

- (1) AWS EC2 に Linux OS による仮想マシンインスタンスを用意
- (2) EC2 のパケットフィルタファイアウォールによるセキュリティポリシーの設定で、SSH, HTTP, HTTPS, DNS によるアクセスを許可
- (3) インスタンスに Web サーバ、データベース、Web アプリケーション用実行環境 LAMP (Linux, Apache2, MySQL, PHP) を導入
- (4) Apache2 にて SSL/TLS を行うモジュールの導入
- (5) AWS 上に DNS サービスの構築
- (6) Let's Encrypt サービスによる SSL/TLS 接続用証明書の取得

- (7) データベース上に Wordpress 用のデータベースとアクセス用ユーザを作成し、ユーザに適切にアクセス許可を設定
- (8) Wordpress の初期設定を行い、Web コンテンツ公開用ディレクトリにファイル群を配置し、ファイルのオーナ、アクセス許可を設定

SSL/TLS による HTTPS 接続のためには、DNS による正式なドメイン名のあるホスト名が必要であり、DNS と Apache を用いたドメイン認証サービス (Domain Validation) である Let's Encrypt サービスによる証明書の署名・交付を受ける。

DNS の設定には、上位 DNS サーバへの設定が必要となるため、実験スタッフに上位 DNS 設定に必要な情報（ドメイン名、DNS サーバ名、DNS サーバ IP アドレス）を伝える。

7.3 実験内容（3回分）

構築は、以下の使用、手順で行う。

7.3.1 仕様

- AWS EC2 は、教育機関用サービス AWSEducate (awseducate.com) から情報学群実験 4C を選択して AWS にログイン。
- 用いるインスタンスは、t2.micro (その他のものでは追加課金となる場合がある)。
- 用いる OS は、Amazon Linux 2 (AML2)
- AML2 が用意する LAMP 導入手順を用いる。
 - この場合、MySQL の代わりに派生版の MariaDB がインストールされるので、これを使う。使い方は MySQL と同一である。
- DNS サービスを BIND にて構築する。
 - ドメイン名 (グループ X の場合): aX.exp.info.kochi-tech.ac.jp
 - DNS サーバ名 (グループ X の場合): server.aX.exp.info.kochi-tech.ac.jp
 - DNS サーバ IP アドレス: AWS EC2 から割り当てられて IPv4 パブリックアドレス
 - Web サーバ名 (グループ X の場合): www.aX.exp.info.kochi-tech.ac.jp
 - BIND インストールは、1台目のサーバだけで良い (Web サーバを 2台以上作成する場合)
- SSL/TLS 用証明書は、Let's Encrypt サービスを使う。
- Wordpress URL(グループ X の場合): <https://www.aX.exp.info.kochi-tech.ac.jp/>
 - 2台目以降を構築する場合は、<https://www2.aX.exp.info.kochi-tech.ac.jp/> と、www の後に番号を付与する。

7.3.2 AWS EC2 インスタンスの準備

既に、各グループで用意済みの t2.micro, AML2 のインスタンスを使う（第1章第3回実験参照）。

7.3.3 AWS EC2 インスタンスにログイン

秘密鍵 (*.pem ファイル) を使ってインスタンスにログインする。

UNIX 系 (Linux, Mac) の場合

```
$ ssh -i 秘密鍵ファイル ec2-user@IP アドレス  
(IP アドレスはインスタンスのもの)
```

Windows の場合は、下記を参考に、PuTTY から接続。

「PuTTY を使用した Windows から Linux インスタンスへの接続」

https://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/putty.html

7.3.4 パスワード等の注意

パブリックなインターネットスペースでのサーバ構築は、常にセキュリティ侵害のリスクに晒されるため、下記のような注意をし、セキュリティリスクを可能な限り避けること。

- パスワード等は、安易なもの、辞書にあるもの、短いものとせず、パスワード生成器で生成されるような強固なものを用い、メンバーが限られた SNS スペース (slack) などで適宜共有する。
- phpinfo() 等のテストプログラムも弱点などを提供し脅威となるため、テスト後は削除する。
- セキュリティグループでのファイアウォール設定もなるべく最小限とすること。

特に、東京オリンピック等のイベント開催時には、セキュリティ侵害攻撃が急拡大する傾向にあるため、日ごろから常に注意をする。

7.3.5 LAMP のインストール

基本的には、AWS 公式チュートリアル「Amazon Linux 2 に LAMP ウェブサーバーをインストールする」に沿って構築する。

https://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/ec2-lamp-amazon-linux-2.html
別紙に用意する通り、

7.3.6 セキュリティ（ファイアウォール）の設定

DNS と Web サービスを行うにあたり、DNS, HTTP, HTTPS アクセスを許可するファイアウォール（パケットフィルタ）設定を AWS に行う。

- AWS Educate にサインイン
- My Classrooms で情報学群実験第 4C を選択 (Go to classroom)
- Workarea → AWS Console (ここで残金が 10 ドルを切っている場合は教員まで報告を)
- 最近アクセスしたサービス等で ec2 選択
- インスタンス下の「セキュリティ」タブからセキュリティグループ (sg-.....) を確認、クリック
- Edit Inbound rules で SSH が許可となっているので、これに下記を追加、ルールを保存する
 - HTTP
 - HTTPS
 - DNS(TCP)
 - DNS(UDP)
 - 全て、ソースは、Anywhere-IPv4 (0.0.0.0/0) を選択

7.3.7 DNS サーバの設定

BIND をインストールし、ドメイン「aX.exp.info.kochi-tech.ac.jp」のゾーンファイルを作成しホストする

```
$ sudo su
# yum update
# yum install bind

/etc/named.conf が DNS 設定ファイル

# vi /etc/named.conf
```

下記 2 行をコメントアウト

(デフォルトはセキュリティ対策で自アドレスのみアクセス可になっているため)

```
listen-on port 53 { 127.0.0.1; };
listen-on-v6 port 53 { ::1; };
↓
// listen-on port 53 { 127.0.0.1; };
// listen-on-v6 port 53 { ::1; };
```

下記を書き換え

(デフォルトはセキュリティ対策で自分のみクエリ可になっているため)

```
allow-query { localhost; };
```

```
↓
allow-query { any; };
```

下記を書き換え

(セキュリティ上再帰検索を禁止するため)

```
recursion no;
```

↓

```
recursion yes;
```

(DNS セキュリティ詳細については、Web 等の「オープンリゾルバ対策」を参考にすること)

下記ゾーンを追加する

(X はグループ名)

```
zone "aX.exp.info.kochi-tech.ac.jp" {
    type master;
    file "zone.aX";
};
```

次にゾーンファイルを作成する

ゾーンファイルは、/var/named に作成

```
# vi /var/named/zone.aX (X: グループ名)
$TTL    100
@       IN      SOA     aX.exp.info.kochi-tech.ac.jp. postmaster.aX.exp.info.kochi-tec
h.ac.jp. (
            001
            100
            100
            100
            100 )
        IN      NS      server.aX.exp.info.kochi-tech.ac.jp.
server  IN      A       AWS EC2 のパブリック IPv4 アドレス
www     IN      CNAME   server
```

DNS サーバを再起動する。

```
# systemctl restart named
```

(AM2 (RedHat 系 Linux) では、named がサービス名)

以上で DNS サーバの作成は完了する。

7.3.8 LAMP 設定の続きと WordPress

以降、LAMP 設定の続き、SSL/TLS 設定と、WordPress のインストールを別紙（チュートリアル）を元に行っていく。

第II部

付 錄 編

付録 A 章

vi エディタの使い方

UNIX には Emacs や mule などの他に vi と呼ばれるエディタがある。 vi は UNIX で標準提供されているエディタで、すべての UNIX システム上で利用することができる（Emacs は豊富な機能を持ったエディタであるが、通常は標準で提供されてはいないため、使用するためにはインストールする必要がある）。

A.1 vi の起動

vi はファイル名を引数に与えて起動する（ファイル名は省略できる）。

% vi [filename]

A.2 vi のモード

vi には、コマンドモードと入力モードがある。 vi を起動した直後のモードはコマンドモードであり、文字を入力できるモードにはなっていない。コマンドモードでは、カーソルの移動、ファイルの保存、テキストの削除、複写などの操作を行う。文字を入力するには入力モードに入る必要がある。コマンドモードから入力モードへの切替えは、`i` などのキーで、また入力モードからコマンドモードへの切替えは `Esc` で行う。

A.3 コマンドモード

コマンドモードでは、コマンドに先行して数引数を与えることができ、ほとんどのコマンドは数引数をコマンドの繰り返し数と解釈する。例えば、`5 j` と入力すれば `j j j j j` と入力することと同じになる。

A.3.1 カーソルの移動

上下左右のカーソルの移動には、コマンドモードでそれぞれ **k** **j** **h** **l** のキーを使う。その他、カーソル移動のコマンド（キー）を以下にまとめる。

コマンド	移動先	コマンド	移動先
j	1行下	G	最終行
k	1行上	nG	n行目
h	1文字左	H	画面の最上行
l	1文字右	M	画面の中央行
0	現在行の先頭	L	画面の最下行
\$	現在行の行末	b, B	前の単語の先頭
^	現在行の先頭の単語の先頭	w, W	次の単語の先頭
Enter	1行下の先頭	e, E	現在もしくは次の単語の末尾

A.3.2 削除、複写、移動

テキストの削除には次のコマンドを使う。

コマンド	意味
x	カーソル位置の文字を削除
X	カーソル位置の直前の文字を削除
dd	現在行を削除
dw	カーソル位置からその単語の最後までを削除
d\$	カーソル位置から行の最後までを削除
d^	カーソル位置から行の先頭までを削除
J	現在行と次行を1行に連結

vi は直前の削除あるいは置換コマンドによって消されたテキストを退避バッファに保存している。また、指定した領域を退避バッファに複写することもできる。テキストを退避バッファに保存した後でカーソルを移動し、退避バッファの内容を現在変集中のテキストに挿入することで、テキストの移動や複写ができる。以下にコマンドを示す。

コマンド	意味
P	退避バッファの内容をカーソル位置に挿入
p	退避バッファの内容をカーソル位置の直後に挿入
yy	現在行を退避バッファに保存
yw	カーソル位置から単語の終りまでを退避バッファに保存

A.3.3 検索、置換

文字列の検索は、**/** の後に検索したい文字列を入力して **Enter** キーを押す。検索を繰り返す場合は、**n** を入力する。また、前にさかのぼって検索する場合には **/** のかわりに **?** を用いる。

文字列を置換する場合は、まず **/** で文字列を検索し、ここで **c** **w** の後に変更する文字列を入力して **Esc** キーを押す。

A.3.4 コマンドの取り消しと再実行

vi では、直前の編集コマンドのみ取り消すことができる。**u** を入力すると直前に実行されたコマンドが取り消される。また、**.** は直前のコマンドを再実行する。

A.3.5 ファイルの読み書き

ファイルの読み書きはコマンドモードで行う。作成（修正）したテキストをファイルとして書き込むためには **:** の後に **w** を入力する¹。

ファイルの読み書きや **vi** の終了などを行う場合には、通常 **:** を入力する。**:** を入力することにより最下行に“**:**”が表示されるから、ここにコマンドを入力する。ファイル操作のコマンド²を以下に示す。

コマンド	機能
:w	現在のファイル名で保存
:w filename	<i>filename</i> に保存
:w!	強制書き込み
:e filename	<i>filename</i> を新たに読み込み
:e!	ファイルの再読み込み
:r filename	<i>filename</i> の内容をカーソル行の次行に挿入

A.3.6 vi の終了

vi の終了の方法には、ファイルに保存後終了、内容が更新されている場合には警告メッセージを表示する終了、強制終了の 3 通りがある。それぞれの終了コマンドは次の通りである。

コマンド	機能
ZZ	編集中の内容を保存して vi を終了
:wq [filename] Enter	編集中の内容を <i>filename</i> に保存し vi を終了
:q Enter	vi を終了
:q! Enter	vi を強制終了

¹ ただし、UNIX ではファイルにオーナー、グループ、パーミッション（使用許可）という属性があるため、“Read-only file, not written; use ! to override.”などのメッセージが表示され、書き込めないことがある（書き込み不許可のパーミッションのとき）。パーミッションを変更する場合には **chmod** コマンド（付録 B 参照）を用いる。

² これらのコマンドは入力後に **Enter** キーを押す必要がある。

A.4 モードの切替え

コマンドモードから入力モードには **i** キーを押すと切り替わる。入力モードに切り替わると自由に文字列を入力することができる。**i** キーでは、現在のカーソル位置で入力モードに切り替わる。入力モードへの切替えコマンドを以下に示す。

コマンド	機能
a	カーソルの直後で入力モード
A	現在行の行末で入力モード
i	カーソル位置で入力モード
I	現在行の最初の単語の先頭で入力モード
o	カーソル行の 1 行下に空行を挿入し入力モード
O	カーソル行の 1 行上に空行を挿入し入力モード
R	カーソル位置で上書き入力モード

入力モードからコマンドモードへの切替えは **Esc** キーにより行う。

付録 B 章

UNIX コマンド集

この章では、サーバの管理上よく使われるコマンドを取り上げてその機能を紹介する。

この章で紹介するコマンドは、ここで紹介する機能以外も持ち合わせているものもある。各コマンドについて詳しく知りたい場合はオンラインマニュアル等を利用して自分で調べて欲しい。

この章で紹介するコマンド以外にも有用なコマンドは沢山あるので、色々と調べてみて欲しい。

ファイル管理

B.1 chgrp (CHange GRouP)

指定したファイルのグループ所有権を変更する

◆書式

```
% chgrp グループ名 ファイル名
```

◆機能説明

chgrp は、引数にグループ名とファイルを指定してファイルのグループを変更する。このコマンドを実行するユーザーは、変更後のグループに属していないといけない。

◆使用例

```
% ls -l file    (← file の詳細な情報を表示する)
-rw-r--r--    1 user1    group1  2058 Sep 18 15:33 file

% chgrp group2 file    (← file の所属するグループを group2 に変更する)
% ls -l file
-rw-r--r--    1 user1    group2  2058 Sep 18 15:33 file
%
```

ファイル管理

B.2 chmod (CHange MODE)

ファイルのパーミッションを変更する

◆書式

```
% chmod [-R] パーミッション ファイル名
```

R: ディレクトリに対して変更内容を再帰的に処理する

```
% ls -l file  (<- file の詳細な情報を表示する)
-rwxr-xr-x  1 user1  group2  2058 Sep 18 15:33 file
```

`rwxr-xr-x` で、文字があるところを 1、ないところを 0 とすると

`111101101` → これを 3 ビットずつ 8 進数にすると `755`

(詳細は下記)

これを使って `chmod 755 ファイル名`などとする

◆機能説明

ファイルやディレクトリのパーミッションを変更する。引数として、パーミッションと変更したいファイルを指定する。パーミッションの指定には、数字で指定する方法と文字列で指定する方法の 2 通りある。文字表記において、`r` は読み込み、`w` は書き込み、`x` は実行の権利を示す。

数字で指定する場合、引数のパーミッション部にパーミッションに対応した数列を指定して変更を行う。

パーミッションの指定は、3 桁の数列で指定をする。その 3 桁の数列は、左から作成ユーザ、グループ、その他のユーザに対してのパーミッションを示しており、読み込みを許可する場合は 4、書き込みを許可する場合は 2、実行を許可する場合は 1 の値をプラスする。

以下の例のモード `754` は、

- 左の桁が 7 なので、作成ユーザに対して読み込み(4)と書き込み(2)と実行(1)を許可する($4+2+1=7$)。
- 中の桁が 5 なので、所属グループのメンバに対して読み込み(4)と実行(1)を許可する($4+1=5$)。
- 右の桁が 4 なので、その他のユーザに対して読み込み(4)を許可する($4=4$)。

◆使用例

```
% ls -l file  (<- file の詳細な情報を表示する)
-rwxr-xr-x  1 user1  group2  2058 Sep 18 15:33 file

% chmod 754 file  (<- file のモードを 754 に変更)
```

```
% ls -l file   (← file の詳細な情報を表示する)
-rwxr-xr--  1 user1  group2  2058 Sep 18 15:33 file

% ls -l file   (← file の詳細な情報を表示する)
-rwxr-xr-x  1 user1  group2  2058 Sep 18 15:33 file

% chmod o-r file   (← その他のユーザの読み込みを不許可にするには o-r とする)

% ls -l file   (← file の詳細な情報を表示する)
-rwxr-x--x  1 user1  group2  2058 Sep 18 15:33 file
%
```

ファイル管理

B.3 chown (CHange OWNer)

指定したファイルの所有者およびグループを変更する

◆書式

```
# chown [-R] ユーザー名 [: グループ名] ファイル名
```

R: ディレクトリに対して変更内容を再帰的に処理する

◆機能説明

chown は引数にユーザとファイル名を指定してファイルの所有者を変更する。このコマンドはスーパーユーザしか実行できない。

◆使用例

```
% ls -l file  (- file の詳細な情報を表示する)
-rw-r--r--    1 user1  group1  2058 Sep 18 15:33 file

# chown user2 file  (- file の持ち主を user2 に変更する)
% ls -l file
-rw-r--r--    1 user2  group1  2058 Sep 18 15:33 file
%
```

ファイル管理

B.4 df (Disk Free) / du (Disk Usage)

ディスクの状態を表示する

◆書式

```
% df [-k] ファイルシステム
% du [-k] ディレクトリ名
```

k:kbytes 単位で表示

◆機能説明

df は、指定したファイルシステムの使用可能な容量を表示する。また、全体の容量と使用済み容量も表示してくれる。

du は、あるディレクトリ中の全ファイルが使用している領域を計算し、ファイルのなかにディレクトリがあれば、再帰的にその中の領域を計算する。

◆使用例

```
% df -k  (←各パーティションの容量の情報を 1k バイト単位で表示する)
Filesystem      kbytes   used  avail capacity  Mounted on
/dev/dsk/c0t0d0s0    96975  17785  69493   21%    /
/dev/dsk/c0t0d0s6   770943  619640  97337   87%   /usr
/proc              0       0     0    0%   /proc
fd                 0       0     0    0%   /dev/fd
/dev/dsk/c0t0d0s5    96975  21743  65535   25%   /var
/dev/dsk/c0t0d0s7  14332795 3420514 10768954  25%   /export/home
/dev/dsk/c0t0d0s4   1018191  143243  813857  15%   /opt
swap               1091416      64  1091352   1%   /tmp
%
% du   (←カレンントディレクトリにあるファイルの容量をみる)
488    ./netscape
1      ./nsmail
1      ./Mail/inbox
1      ./Mail/draft
1      ./Mail/trash
4      ./Mail
2      ./ssh
1493   .
%
```

テキスト処理

B.5 grep (Global Regular Expression Print)

パターンにマッチする行を表示する

◆書式

```
% grep 文字列 [ファイル名]
```

◆機能説明

ファイルの中身から引数に指定された文字列を含む行を表示する。文字列の指定には正規表現を用いる事ができる。

また、ファイルを指定しない場合は標準入力から読み込むので、下の例のように出力が多い処理の出力を grep に向ける事により、必要な情報のみを抜き出して表示する事ができる。

◆使用例

```
% ls /dev | grep console    (←/devの中から console という文字列があれば表示する)
console
%
```

テキスト処理

B.6 more, less, lv

ファイルの中身を表示する

◆書式

```
% more ファイル名
(less, lv も同じ)
```

◆機能説明

指定したファイルのデータを 1 画面ずつ表示する。1 画面に入らない時はスペースで次の部分を表示する。最後まで表示したら自動的に終了する。more より高機能なツールに less がある。

◆ less のオプション、使い方

```
less ファイル名
Space: 次画面
b : 前画面へバック
C-n: 1 行下
C-p: 1 行上
q: 終了
/ : 検索
  n : 次の検索ワード位置
M-< または 1 G : 最初の画面
M-> または G : 最後の画面
F : 追記データを待つ (C-c で待機終了)
```

オプション

- i : 検索時に大文字小文字を区別しない
- X : 終了時に画面をクリアしない

◆使用例

```
# more asppp.log
11:49:19 parse_config_file: Errors in configuration file /etc/asppp.cf
11:49:19 Link manager (135) exited 03/24/99
12:02:44 Link manager (103) started 03/24/99
12:02:44 parse_config_file: no paths defined in /etc/asppp.cf
12:02:44 parse_config_file: Errors in configuration file /etc/asppp.cf
12:02:44 Link manager (103) exited 03/24/99
12:07:05 Link manager (102) started 03/24/99
```

```
12:07:05 parse_config_file: no paths defined in /etc/asppp.cf
12:07:05 parse_config_file: Errors in configuration file /etc/asppp.cf
12:07:05 Link manager (102) exited 03/24/99
10:38:04 Link manager (90) started 04/12/99
10:38:04 parse_config_file: no paths defined in /etc/asppp.cf
10:38:04 parse_config_file: Errors in configuration file /etc/asppp.cf
10:38:04 Link manager (90) exited 04/12/99
11:51:20 Link manager (92) started 04/12/99
11:51:20 parse_config_file: no paths defined in /etc/asppp.cf
11:51:20 parse_config_file: Errors in configuration file /etc/asppp.cf
11:51:20 Link manager (92) exited 04/12/99
16:53:04 Link manager (92) started 04/12/99
16:53:04 parse_config_file: no paths defined in /etc/asppp.cf
-- 繙続 --(22%) (← 繙続と出ている場合、space キーを押せば続きを見る事ができる)
#
```

テキスト処理

B.7 tail

ファイルの最後の部分を表示する

◆書式

```
% tail ファイル名
```

◆機能説明

ファイルの最後の数行を表示する。ログあるいはファイルなどで最後の方にあるデータだけを調べたいときに便利。

◆使用例

```
# tail setuid.today (← setsid.today の最後の数行を表示する)
-r-sr-xr-- 1 root  network  222240 Sep 17 08:46:54 1999 /usr/sbin/ppp
-r-sr-xr-x  1 root  wheel    85504 Sep 17 08:47:04 1999 /usr/sbin/pppd
-r-xr-sr-x  2 root  kmem    13184 Sep 17 07:48:20 1999 /usr/sbin/pstat
-r-sr-xr-x  5 root  wheel    290288 Sep 17 07:48:37 1999 /usr/sbin/purgestat
-r-sr-xr-x  5 root  wheel    290288 Sep 17 07:48:37 1999 /usr/sbin/sendmail
-r-sr-x---  1 root  network  9768 Sep 17 07:48:24 1999 /usr/sbin/sliplogin
-r-xr-sr-x  2 root  kmem    13184 Sep 17 07:48:20 1999 /usr/sbin/swapinfo
-r-sr-xr-x  1 root  wheel    13440 Sep 17 07:48:28 1999 /usr/sbin/timedc
-r-sr-xr-x  1 root  wheel    11232 Sep 17 07:48:29 1999 /usr/sbin/traceroute
-r-xr-sr-x  1 root  kmem    7036 Sep 17 07:48:29 1999 /usr/sbin/trpt
#
```

ネットワーク関連

B.8 ip, ifconfig (InterFace Configuration) , ipconfig

ネットワークインターフェースのパラメータ設定及び確認を行う

◆書式

```

    インタフェース (IF) にパラメータを振る
% ifconfig IF名 [inet IP アドレス] [netmask ネットマスク] [broadcast ブロードキャスト
アドレス]

    インタフェースのパラメータを表示する
% ifconfig -a

    % ip address (ip a と省略可)

    インタフェース IF の有効化
% ifconfig IF名 up

    インタフェース IF の無効化
% ifconfig IF名 down

    インタフェース IF に2つ目の IP アドレスの付与
% ifconfig IF名 alias IP アドレス netmask ネットマスク

    インタフェース IF の2つ目の IP アドレスの削除
% ifconfig IF名 -alias IP アドレス netmask ネットマスク

```

broadcast は省略することができる。省略した場合は、ホスト部のビットが全て1のアドレスがブロードキャストアドレスとして使われる (All 1 broadcast)。

◆機能説明

ifconfig はネットワークインターフェースにパラメータ (IP アドレスやネットマスク, ブロードキャストアドレス) を割り振ったり, ネットワークインターフェースに割り振られたパラメータを見たりする場合に利用する。ネットワークインターフェースの設定は、スーパーユーザのみしか行えないもので注意が必要である。

◆使用例 (1) ネットワークインターフェースにパラメータを割り振る

```

% ifconfig bge0 inet 192.168.0.151 netmask 255.255.255.0 up [Enter]
% ここでは、IP アドレス割り当てと同時にインターフェースの有効化を行って
% いる。

```

これにより、ネットワークインターフェース fxp0 に IP アドレス 172.21.20.15, 24 ビットのネットマスクが割り振られ、172.21.20.191 にブロードキャストするよう設定された。

◆使用例 (2) ネットワークインターフェースのパラメータを参照する

```

% ifconfig -a [Enter]
lo0: flags=849<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
      inet 127.0.0.1 netmask ff000000

```

```
hme0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500  
        inet 172.21.30.10 netmask ffffff00 broadcast 172.21.30.255
```

ネットワークインターフェースに関する情報が表示された。

windows では ipconfig となっている。

ネットワーク関連

B.9 netstat (NETSTATUs)

ネットワークの状態を表示する

◆書式

```
% netstat [-rn]
```

r : ルーティングテーブルを表示する
n : ネットワークアドレスの表示を数字で行う.

◆機能説明

netstat は、ネットワークに関連したさまざまな情報を表示するプログラムである。ここで説明するのはネットワークインターフェースのパケットトラフィックに関するルーティングテーブルをひょうじする r オプションに関してのみであるが、他にもアクティブソケットの一覧や他のネットワークの状態等オプションによって用途に適した情報が得られるので、他の機能を利用したい場合はマニュアルを読んで欲しい。

また、通常 netstat はできる限り IP アドレスをホスト名で表示しようとするが、n オプションをつけることによりアドレスを数字で表示する。

◆使用例

```
% netstat -r [Enter]
```

Routing Table:

Destination	Gateway	Flags	Ref	Use	Interface
172.21.54.0	test	U	3	47873	hme0
BASE-ADDRESS.NET	test	U	3	0	hme0
default	dss.kochi-tech.ac.jp	UG	0	208343	
localhost	localhost	UH		02753237	lo0

%

ネットワーク関連

B.10 nslookup

ネームサーバに対話的に問い合わせを行う

◆書式

```
% nslookup
```

◆機能説明

nslookup は IP アドレスとホスト名の対応を調べる。nslookup を実行して、その後に調べたいホスト名あるいは IP アドレスを入力する。終わるときは exit と入力する。設定した DNS サーバで正しくホスト名と IP アドレスの変換が行われているか調べるときに使う。

◆使用例

```
% nslookup
Default Server: dns.xxx.yyy.zzz
Address: 192.168.1.3

> 192.168.1.1  (← IP アドレス 192.167.1.1 のホスト名を調べる)
Server: dns.xxx.yyy.zzz
Address: 192.168.1.3

Name: machine1.xxx.yyy.zzz (← ホスト名の情報)
Address: 192.168.1.1

> machine1.xxx.yyy.zzz  (← ホスト名 machine1 の IP アドレスを調べる)
Server: dns.xxx.yyy.zzz
Address: 192.168.1.3

Name: machine1.xxx.yyy.zzz
Address: 192.168.1.1

> exit  (← nslookup を終了する)
%
```

ネットワーク関連

B.11 dig

DNS サーバの挙動を表示する

◆書式

```
% dig @[DNS サーバ] [確認したいドメイン名] [レコード]
```

◆機能説明

DNS サーバに対して問い合わせを行い応答結果をセクション毎に表示する。3 番目引数にレコードを追加することでそのタイプの検索結果を表示し、ANY とすると全てのレコードの検索結果を表示する。レコードを指定しない場合は A(ネットワークアドレス) についての情報となる。単に dig とするとルートサーバの表示をする。nslookup と異なり DNS サーバの応答パケットをほぼそのまま表示する。

◆使用例 (1)～DNS サーバに指定したドメイン名の応答を全て表示する

```
% dig @server.gX.info.kochi-tech.ac.jp gX.info.kochi-tech.ac.jp ANY  (← DNS サーバ  
server.gX～における gX.info～の応答を全て表示する)

; <>> DiG 9.3.4-P1 <>> @server.gX.info.kochi-tech.ac.jp gX.info.  
kochi-tech.ac.jp ANY
; (1 server found)
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24369
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1

;; QUESTION SECTION:
;gX.info.kochi-tech.ac.jp. IN      ANY

;; ANSWER SECTION:
gX.info.kochi-tech.ac.jp. 3600 IN SOA  server.gX.info.kochi-tech.ac.jp.  
postmaster.gX.info.kochi-tech.ac.jp. 2002081601 10800 3600 604800 604800
gX.info.kochi-tech.ac.jp. 3600 IN NS   server.gX.info.kochi-tech.ac.jp.
gX.info.kochi-tech.ac.jp. 3600 IN MX   10 server.gX.info.kochi-tech.ac.jp.

;; ADDITIONAL SECTION:
server.gX.info.kochi-tech.ac.jp. 3600 IN A    172.21.1X.2

;; Query time: 1 msec
;; SERVER: 172.21.1X.2#53(172.21.1X.2)
```

```
; WHEN: Mon Mar 2 21:18:02 2009
;; MSG SIZE rcvd: 145
```

◆ dig コマンド実行時の表示内容

- ヘッダ

ヘッダ部分は1行目にはdigコマンドのバージョンとコマンドの内容、2行目には該当したサーバ数、3行目にはグローバルオプションの表示、4行目には応答を受けたことを表示する。5行目には応答パケットのヘッダ情報を表示し、正しく情報を得られた場合はNOERROR、ドメイン名が存在しない場合はNXDOMAINとstatusの後に表示する。6行目flagsでこの応答がどのようなものか知ることができる。aaは権威のある回答であることを示しており、指定したドメイン名の情報を持ったDNSサーバからの応答だということを示している。QUERYには各セクションにいくつ該当するものがあったか表示する。

- QUESTION SECTION

表示させる内容。[ドメイン名][クラス][レコード]の順に並んでいる。例の場合は指定したDNSサーバの持つgX.a360.info.kochi-tech.ac.jpの全ての情報を表示する。

- ANSWER SECTION

DNSサーバの応答内容。[ドメイン名][リフレッシュ間隔][クラス][レコード][データ]の順に並んでいる。例の場合は要求をANYで行っているので、SOA, NS, MXの3つが返って来ている。

- ADDITIONAL SECTION

追加としてDNSサーバの情報を表示する。

- フッタ

フッタには、検索にかかった時間、使用したDNSサーバ、検索した日時、受け取ったデータサイズなどを表示する。

◆使用例(2)～正引き(ドメイン名からIPアドレスを調べる)

```
% dig machine.gX.info.kochi-tech.ac.jp (← machine.gX～のドメイン名のIPアドレスを調べる)

...
;; ANSWER SECTION:
machine.gX.info.kochi-tech.ac.jp. 3600 IN A 172.21.1X.4
...
```

◆使用例(3)～逆引き(IPアドレスからドメイン名を調べる)

```
% dig @server.gX.info.kochi-tech.ac.jp 4.1X.21.172.in-addr.arpa. PTR (← IPアドレス
172.21.1X.4のドメイン名を調べる)

...
```

```
; ; ANSWER SECTION:  
4.1X.21.172.in-addr.arpa. 3600 IN PTR server.gX.info.kochi-tech.ac.jp.  
...
```

上記のようにして、逆引きのレコードである PTR を呼び出すことで IP アドレスからドメイン名を調べることができるが、`-x` というオプションを用いて

```
% dig -x 172.21.1X.4
```

とすることでも同様の結果を得ることができる。

ネットワーク関連

B.12 ping

パケットをネットワーク上のホストへ送る

◆書式

```
% ping [ホスト名 or IP アドレス]
```

◆機能説明

特別なパケットを指定したホストとやり取りする事によって、そのホストがネットワークにつながっているかどうか調べる。

ping によってつながっている事が確認できた事を「ping が通った」と言う場合が多い。

ネットワークがつながっているかどうか調べる時に ping はとても便利なコマンドであるが、昨今セキュリティの問題上 ping を返さないネットワークも出て来ているので ping が通らないからと言ってネットワーク上つながっていないと一概に言えない場合もある。しかし、それは特殊な例であり、ping が通らない場合はつながっていないと判断して構わない。

ping での導通確認は、ネットワークの OSI レイヤーでの第 3 層が正しく動作していることを示す。

◆使用例～ Linux で実行した場合

```
% ping 192.168.1.2  (-> IP アドレスが 192.168.1.2 のマシンに ping をおくる)
PING 192.168.1.2 (192.168.1.2): 56 data bytes
64 bytes from 192.168.1.2: icmp_seq=0 ttl=254 time=1.047 ms
64 bytes from 192.168.1.2: icmp_seq=1 ttl=254 time=0.978 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=254 time=18.238 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=254 time=0.973 ms
64 bytes from 192.168.1.2: icmp_seq=4 ttl=254 time=0.983 ms
~C (-> [Ctrl]+[C]で中断)
--- 192.168.1.2 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.973/4.444/18.238/6.897 ms  (-> ping が通った)

% ping 192.168.1.1  (-> IP アドレスが 192.168.1.1 のマシンに ping をおくる)
PING 192.168.1.1 (192.168.1.1): 56 data bytes
~C (-> [Ctrl]+[C]で中断)
--- 192.168.1.1 ping statistics ---
16 packets transmitted, 0 packets received, 100% packet loss  (-> ping が通らなかった)
%
```

ネットワーク関連

B.13 traceroute, tracert

パケットをネットワーク上での経路を調査する

◆書式

```
% traceroute [宛先ホスト名 or IP アドレス]
```

◆機能説明

ping は特定ホストとの間でパケットの送受信ができるか調べるだけであるが、traceroute を用いると、さらに、どのような途中経路（ルータ）を通って宛先までパケットが到達するか調べることができる。

状況によっては、このコマンドを使うことができない場合、途中一部分が表示されない場合、途中から先が表示されない場合など、完全に動作しない場合もある。（ファイアウォールやセキュリティ関連の設定で、許可されていない場合が多い）

また、状況によっては、ping -r コマンドで、行き・帰りのルートを調査することが可能な場合もある。

-n オプションで、IP アドレスのみを表示させる（ホスト名を解決しない）、-P オプションで送出するパケットの種類を指定（icmp, udp, tcp：デフォルトは udp）することができる。

この他にも多くのオプションがあるが、うまく動作しない場合、オプションを変えると調査可能な場合がある。

Windows では古いバージョンで 8 文字までのコマンド名しか許されなかったため、tracert となっている。

◆使用例

```
% traceroute -n -P icmp 172.30.0.1
traceroute to 172.30.0.1 (172.30.0.1), 64 hops max, 60 byte packets
1 * * *
2 172.21.30.10  0.228 ms  0.235 ms  0.245 ms
3 222.229.72.1  3.111 ms  2.490 ms  2.498 ms
4 172.17.1.1    2.611 ms  2.497 ms  2.368 ms
5 222.229.65.41 2.366 ms  0.743 ms  0.744 ms
6 222.229.65.25 1.117 ms  1.118 ms  0.994 ms
7 172.30.0.1    1.117 ms  0.994 ms  0.994 ms
%
```

（宛先ホストまで、6 つのルータを通過しており、それぞれの IP アドレス
が表示されている。1 つ目のルータは調査できなかったことを示している。）

ネットワーク関連

B.14 route(ROUTE)

手作業でルーティングテーブルを操作する

◆書式

テーブルを追加／削除

```
% route [add | delete] [-net | -host ] 宛先 中継点 [netmask] ネットマスク  
宛先へのルートを検索して表示する  
% route get 宛先
```

◆機能説明

route はルーティングテーブルを手動で追加や削除する際に用いるユーティリティである。route で指定したテーブルはすぐに有効になるので新たな経路を作成している時にいちいち「rc.conf を書き換えて再起動をする」といった手段を踏まなくても経路を模索することができるので便利である。

テーブルの追加や削除、検索等は第 1 引数を切り替えで行う（add の場合テーブルの追加、delete の場合テーブルの削除、get の場合ルートの検索）。また、第 2 引数の切り替えでネットワークに対してのテーブルか（-net）、ホストに対してのテーブルか（-host）を選べる。

ルーティングテーブルを追加した際には、netstat 等で確認を行うことができる。

◆使用例（1）ルーティングテーブルの追加

```
% route add -net 172.21.44 172.21.43.21 -netmask 255.255.255.0 [Enter]  
add net 172.21.44:gateway 172.21.43.21  
%
```

新たなルーティングテーブルが追加された。（172.21.44/24 のネットワークへの中継点は 172.21.43.21 である）

◆使用例（2）ルーティングテーブルの削除

```
% route delete -net 172.21.44 172.21.43.21 -netmask 255.255.255.0 [Enter]  
delete net 172.21.44:gateway 172.21.43.21  
%
```

使用例（1）で作成したルーティングテーブルを削除した。

プロセス管理

B.15 shutdown(SHUTDOWN)

指定時刻にシステムを停止する

◆書式

```
(Linux, FreeBSD)
% shutdown [-h | -r | -k] 停止時間 警告メッセージ
    h : システムを停止する
    r : システムを再起動する
    k : 全ユーザを追い出す
```

◆機能説明

shutdown コマンドはスーパーユーザーでないと行うことはできない。

shutdown は、オプションの指定によって終了方法が異なる。

停止時間は hh:tt 時間表記 (12:10 → 12 時 10 分) や +m 表記 (+5 → 5 分後) といった指定ができる、now とすると +0 の意味をなし、その場で停止しようとする。停止時間は必ず指定しなければならないが、警告メッセージは書かなくても良い。

k オプションを指定した時は、実際にはシステムは停止せずにログイン中のユーザにユーザをログアウトさせるメッセージを送り、それ以後システム停止までログインをできなくさせて、その間にバックアップを取るなどの手段を取ることができる。

システムの停止をするとユーザに不便な思いをさせるので乱発は控え、本当に停止すべき状況かどうかを考えてから停止をしたほうが良い。「シャットダウン癖をつくらない」ほうが良い。

また、p オプションで電源を切るにはハードウェア側で対応していないと意味がないので注意が必要である。

◆使用例 システムを停止する (Linux, FreeBSD)

```
% shutdown -h now [Enter]
*** FINAL System shutdown message from root@test@info.kochi-tech.ac.jp ***
Shutting down demon process
.
.
.
(メッセージが流れる)
.
The operating system has halted.
Please press any key to reboot.   (←電源を切れる状態になった)
```

電源を切れる状態になったことをメッセージで確認してから電源を切る。また、メッセージにもあるように何かキーを押すと再起動が始まる。

◆使用例（2）全ユーザを追い出す（Linux）

```
% shutdown -k now [Enter]
*** FINAL System shutdown message from root@test@info.kochi-tech.ac.jp ***
System going down IMMEDIATELY

Feb 28 17:46:33 test@info shutdown:shutdown by root

System shutdown time has arrived
but you'll have to do it yourself [enter]
%
```

以後、一般ユーザは一端ログアウトしてしまうとログインしようとしても以下のようなメッセージが
出るだけでログインができなくなる。

```
login: user [Enter]

Password: [パスワードを入力して enter]

NO LOGINS: System going down at 17:46
```

プロセス管理

B.16 kill

プロセスを終了させたりシグナルを送ったりする

◆書式

```
% kill プロセス ID
% kill [signal] プロセス ID
```

◆機能説明

kill コマンドは、指定されたプロセス ID のプロセスに対しシグナルを送るコマンドである。シグナル名がついていない時は、指定されたプロセスの実行を停止させる。

自分のプロセスにはシグナルを送信する事ができるが、他人のプロセスに対してはシグナルを送信する事ができない。しかし、スーパーユーザだけは、他人のプロセスに対してもシグナルを送る事ができる。

使えるシグナル名は -l オプションでわかる。よく使われるシグナル名として HUP (Hang UP) がある。HUP シグナルは、指定されたプロセスを一度停止させ再開する信号である。例えば、inetd.conf を変更した場合、inetd プロセスに設定を改めて読み込ませるために以下のように HUP シグナルを送る。

◆使用例

```
% kill -l  (←使えるシグナル名を探す)
HUP INT QUIT ILL TRAP ABRT EMT FPE KILL BUS SEGV SYS PIPE ALRM TERM USR1 USR2
CHLD PWR WINCH URG IO STOP TSTP CONT TTIN TTOU VTIME PROF XCPU XFSZ

# ps aux
freebsd# ps aux
freebsd# ps aux
USER     PID %CPU %MEM    VSZ   RSS   TT  STAT STARTED          TIME COMMAND
root      132  0.0  0.0  1536   848   ??  Is   12:37PM  0:00.00 adjkerntz -i
root      449  0.0  0.0  1888   540   ??  Is   12:37PM  0:00.00 /sbin/devd
root      556  0.0  0.1  3344  1308   ??  Is   12:37PM  0:00.01 /usr/sbin/syslogd -s
root      782  0.0  0.2  6676  3596   ??  Is   12:37PM  0:00.00 /usr/sbin/sshd
root      797  0.0  0.2  6072  3340   ??  Ss   12:37PM  0:00.02 sendmail: accepting conn
smmsp    801  0.0  0.2  6072  3388   ??  Is   12:37PM  0:00.00 sendmail: Queue runner@0
root      807  0.0  0.1  3372  1352   ??  Is   12:37PM  0:00.01 /usr/sbin/cron -s
root      865  0.0  0.2  9400  4372   ??  Ss   12:37PM  0:00.06 sshd: root@pts/0 (sshd)
root      874  0.0  0.1  3376  1412   ??  Ss   12:38PM  0:00.01 rpcbind
root      877  0.0  0.1  3344  1484   ??  Is   12:38PM  0:00.00 mountd
root      879  0.0  0.1  3284  1380   ??  Ss   12:38PM  0:00.02 nfsd: master (nfsd)
root      857  0.0  0.1  3344  1160   v0  Is+  12:37PM  0:00.00 /usr/libexec/getty Pc tt
root      868  0.0  0.1  4664  2460     0  Rs   12:37PM  0:00.02 -csh (csh)
```

```
root      936  0.0  0.1  3424  1144   0  R+    1:07PM  0:00.00 ps aux
# kill -KILL 807(←プロセス ID 807 番を再起動させた)
(kill -9 807 でも良い。9はKILL シグナルのシグナル番号
KILL は強制終了, TERM は終了(中断), HUP は(再起動)である)
#
```

プロセス管理

B.17 ps (Process Status)

プロセスの状態の表示

◆書式

(Solaris) /usr/bin/ps [-uAef]

u : 指定するユーザ ID を持つ全てのプロセスを表示する

A : すべてのプロセスを表示する

e : 現在実行中のすべてのプロセスを表示する

f : 完全形式で表示する

(FreeBSD, Linux) ps [uaxef]

u : ユーザ形式で表示する

a : プロセスグループリーダを除く全プロセスを表示する

x : 制御端末を持たないプロセスを含めて表示する

e : 全てのプロセスを表示する

f : 完全形式で表示する

w : 1 行に入らないものも続けて表示.

ww: さらに長いものも続けて全て表示.

◆機能説明

ps はシステムで動作しているプロセスの情報をプロセス ID 順に表示する。表示される情報はデフォルト（何もオプションをつけない）で、プロセス ID, 制御端末名, プロセス状態, cpu 時間などを表示する。ps はシステムより速く実行できず、他のプロセスと同様にスケジュールされて実行されるので、表示される情報は正確ではない。オプションをつける事により、より多くの情報を表示する。

ps コマンドはサーバでサービスがきちんと動いているか確認したり、プロセスのゾンビが残っていたりしないか確認するときに使い、プロセスを指定する kill のようなコマンドと一緒に使うこともある。

◆使用例

freebsd# \underline{ps auxww} （←全プロセスの表示）												
USER	PID	%CPU	%MEM	VSZ	RSS	TT	STAT	STARTED	TIME	COMMAND		
root	11	100.0	0.0	0	8	??	RL	12:37PM	1263:15.43	[idle]		
root	0	0.0	0.0	0	48	??	DLS	12:37PM	0:00.08	[kernel]		
root	132	0.0	0.0	1536	848	??	Is	12:37PM	0:00.00	adjkerntz -i		
root	449	0.0	0.0	1888	540	??	Is	12:37PM	0:00.00	/sbin/devd		
root	556	0.0	0.1	3344	1308	??	Ss	12:37PM	0:00.11	/usr/sbin/syslogd -s		
root	782	0.0	0.2	6676	3596	??	Is	12:37PM	0:00.00	/usr/sbin/sshd		
root	797	0.0	0.2	6072	3448	??	Ss	12:37PM	0:00.90	sendmail: accepting conn		
smmsp	801	0.0	0.2	6072	3388	??	Is	12:37PM	0:00.02	sendmail: Queue runner@0		
root	807	0.0	0.1	3372	1352	??	Is	12:37PM	0:00.16	/usr/sbin/cron -s		

root	865	0.0	0.2	9400	4416	??	Is	12:37PM	0:03.36	sshd: root@pts/0 (sshd)
root	3217	0.0	0.2	9400	4428	??	Ss	7:58AM	0:00.07	sshd: root@pts/1 (sshd)
root	857	0.0	0.1	3344	1160	v0	Is+	12:37PM	0:00.00	/usr/libexec/getty Pc tt
root	858	0.0	0.1	3344	1160	v1	Is+	12:37PM	0:00.00	/usr/libexec/getty Pc tt
root	3220	0.0	0.1	4664	2588	1	Rs	7:58AM	0:00.04	-csh (csh)
root	3431	0.0	0.1	3424	1144	1	R+	9:41AM	0:00.00	ps aux

サービス利用

B.18 mail

メールの送信と受信を行う

◆書式

```
% mail [-s] [-c mailaccount] [アドレス]
```

s: subject をコマンドラインから入力する。
 c: Carbon Copies で送信する。

◆機能説明

mail コマンドはメールの送受信を行う。引数に送り先のメールアドレスを指定すると送信モードになり、引数をつけないとメール受信モードになる。

メールを送信するにはメール送信モードに入り、Subject と内容を書き込む。

メール受信モードでは & プロンプトが表示されるので読みたいメールの番号を打つ。メールを消す場合には、> を消したいメールに移動させてから [d] と打つ。受信メールがない場合には、You have no mail. と表示される。メール受信モードを終了させるには & プロンプトが表示されているときに [q] と打つ。読み終ったメールは終了する時に mbox というファイルに保管される。

◆使用例 (1)～メール送信

```
% mail xxx@ugs.kochi-tech.ac.jp [Enter]  (←メール送信の例)
Subject: test mail [Enter]  (←Subject を入力)
This is test mail.  (←メールの内容を入力)

%
```

◆使用例 (2)～メール受信

```
% mail [Enter]  (←メール受信の例)
Mail version 5.3 2/18/88. Type ? for help.
"/usr/spool/mail/username": 6 messages
> 1 admrn@abcdef.com Mon Mar 13 02:24 355/22876 "RECRUIT NAVI Mailing "
  2 aaa@ugs.kochi-te Mon Mar 13 18:17 50/1666 "=?ISO-2022-JP?B?GyRCJW"
  3 bbb@info.kochi-t Mon Mar 13 18:40 37/1395 "Re: =?ISO-2022-JP?B?Gy"
  4 ccc@ugs.kochi-te Mon Mar 13 23:25 37/1438 "[admin,00131] =?iso-20"
  5 ddd@ugs.kochi-te Tue Mar 14 10:04 51/1879 "[admin,00132] Re: =?is"
  6 fff@ugs.kochi-te Tue Mar 14 11:43 43/1497 "Re: [admin,00131] 22=?"
& 1 [Enter]  (←1番のメール読む)
Message 2:
```

```
test mail.
```

```
& d [Enter]  (← 1 番のメールを削除する)
```

```
> 2 aaa@ugs.kochi-te Mon Mar 13 18:17 50/1666 "=?ISO-2022-JP?B?GyRCJW"
  3 bbb@info.kochi-t Mon Mar 13 18:40 37/1395 "Re: =?ISO-2022-JP?B?Gy"
  4 ccc@ugs.kochi-te Mon Mar 13 23:25 37/1438 "[admin,00131] =?iso-20"
  5 ddd@ugs.kochi-te Tue Mar 14 10:04 51/1879 "[admin,00132] Re: =?is"
  6 fff@ugs.kochi-te Tue Mar 14 11:43 43/1497 "Re: [admin,00131] 22=?"
```

```
& q [Enter]  (←受信メールモードを終了する)
```

```
%
```

サービス利用

B.19 man

オンラインマニュアルのフォーマット、表示を行なう

◆書式

```
% man コマンド名
```

◆機能説明

引数としてコマンド名を指定して、コマンドのオンラインマニュアルを表示する。

man コマンドは、/usr/share/man に格納されているマニュアルを参照する。

◆使用例

```
% man man

Reformatting page. Wait... done

User Commands                               man(1)

NAME
    man - find and display reference manual pages

SYNOPSIS
    man [ - ] [ -adFlrt ] [ -M path ] [ -T macro-package ]
        [-s section] name ...
    man [ -M path ] -k keyword ...
    man [ -M path ] -f file ...

DESCRIPTION
    The man command displays information from the reference
    manuals. It displays complete manual pages that you select
    by name, or one-line summaries selected either by keyword
    (-k), or by the name of an associated file (-f). If no
    manual page is located, man prints an error message.

Source Format
    Reference Manual pages are marked up with either nroff(1) or
    sgml(5) (Standard Generalized Markup Language) tags. The
--More--(5%)
%
```

サービス利用

B.20 telnet

TELNET プロトコルで通信を行う

◆書式

```
% telnet [ホスト名 or IP アドレス] [ ポート番号 ]
```

◆機能説明

telnet は TELNET プロトコルを用いてネットワーク上の他の端末にアクセスする時に使う。

引数として、ホスト名または IP アドレスを指定する。通常はディフォルトの telnet ポート（ポート番号 23）を叩くが、ポート番号を指定する事で telnet が叩く TCP ポート番号を指定する事ができる。また、接続先のマシンにアカウントがないと接続できないので注意が必要である。自分が使っているマシンから遠隔地にある他のマシンに接続するときによく使う。

◆使用例

```
% telnet sun
Trying 192.168.1.1...
Connected to sun.a360.kochi-tech.ac.jp.
Escape character is '^]'.

SunOS 5.6

login: omori
passwd: ohji

Last login: Thu Mar 23 16:26:22 from omori.star.space
Sun Microsystems Inc. SunOS 5.6 Generic August 1997
%
```

ソフトウェア管理

B.21 tar (Tape ARchive)

複数のファイルを一つのファイルにまとめるアーカイブ化およびアーカイブの展開を行う。

◆書式

```
tar [xtcvzf] filename
x : extract, アーカイブを展開する
t : list, アーカイブの内容（ファイル）を表示する
c : create, アーカイブを作成する
v : verbose, 実行時に詳細の内容を表示する
z : 圧縮を行う（tar 内部で gzip 等の圧縮ソフトウェアを用いる）
f : filename, アーカイブファイル名をこの後に続けて書く
```

◆機能説明

tar は、UNIX で標準のアーカイバソフトウェアである。バックアップの際などに、ディレクトリごとまとめてみたい場合に用いたり、アーカイブ化されたファイルを展開する際に用いる。

サーバ等のフリーソフトウェアのソースコード配布も、tar (+ gzip) 形式が多いので、tar を用いて展開しコンパイルする。

◆使用例

```
freebsd# tar cvzf archive.tar.gz folder
(フォルダ folder 以下の全てのファイルを archive.tar.gz というファイルにアーカイブする。)
freebsd# tar xvzf software.tar.gz
(現在のフォルダに software.tar.gz にアーカイブされている全てのファイルを展開する。注意点として、software.tar.gz が单一フォルダ以下で構成されている場合は問題ないが、そうでない場合は、多くのファイルがカレントフォルダに作成されるので、その際は、mkdir で新しいフォルダを前もって作成し、その中で展開するなどする。単一フォルダか否かの内容の確認は下の例を参照。)
freebsd# tar tvzf software.tar.gz
(内容の確認。実際にはファイルは作成されない。)
```

ソフトウェア管理

B.22 make

プログラムの依存関係をメンテナスする

◆書式

```
% make ターゲット
```

◆機能説明 make は、複数のファイルからなるプログラムのコンパイルを支援する。実行すると、同一フォルダ内の makefile か Makefile というファイルの生成とプログラムの依存関係を記したファイル（両方ある場合は最初に見つかったほう）を読み込み、最初にあるターゲットを読み込む。ターゲットを指定した場合、そのターゲットの内容を実行する。Makefile の内容を簡略的に以下に示す。

変数の設定

(まず、 Makefile 内で使用する変数を定義する)

ターゲット 1 : ファイル, ファイル...

コマンド

コマンド

(ターゲット 1 の依存関係を記述)

ターゲット 2 : ファイル, ファイル...

コマンド

コマンド

(ターゲット 2 の依存関係を記述)

Makefile は、このような書式になっている。make がターゲットなしで実行された場合は、最初のターゲットであるターゲット 1 の内容に従って make が実行される。もしコマンド引数のターゲットにターゲット 2 を指定すると、ターゲット 2 の内容に従って make が実行される。

◆使用例

```
% make [Enter]
```

ソフトウェア管理

B.23 patch

パッチをあてる

◆書式

```
% patch < パッチファイル
```

◆機能説明

diff プログラムにより作成された差分ファイル（diff ファイルをオリジナルのファイルに適応する。パッチは、何というファイルの差分かという情報はパッチの中に含んでいる。

◆使用例

```
% patch < patchfile.patch [Enter]  
patching file 'file.txt'  
%
```

patchfile.patch に含まれる file.txt の差分を file.txt に追加した。

その他

B.24 su(Switch User)

他のユーザになりかわる

◆書式

ルートになる時

% su

他のユーザになる時

% su ユーザ名

◆機能説明

su を行うと、一時的に他のユーザとして作業を行う事ができる。ログインする時にディフォルトでは環境変数 HOME, SHELL, USER(ユーザ ID が 0 の場合のみ) はターゲットとなるログインのディフォルトとなり、それ以外の環境変数は引き継がれる。

特定のユーザ(ルート等)になろうとする時に、現在のユーザがそのユーザになる資格を持ってない場合はログインできないので注意が必要である。

また、ログインを終了させるとときは exit と入力して終了する。

◆使用例

```
% su [Enter]
Password: [パスワードを入力して enter]
Mar 26 17:03:32 group@info su: syu to root on /dev/ttyp0
group@info# exit [Enter]
exit
%
```

ルートとして一時的にログインをして、その後ログアウトをした。

付録 C 章

Cisco IOS コマンド集

C.1 IOS の基本操作

基本オペレーション

後述するモードからコマンドを実行する。

? ヘルプ, タブ補完, 省略形を用いて素早い操作ができる。

? ヘルプ

? キーをタイプすることで、コマンド一覧と簡単な説明が表示される。

? ヘルプ (2)

途中までキータイプしてから ? キーをタイプすると、そこで実行可能なコマンド一覧が表示される。

```
cisco#show ip r?  
redirects  rip  route  rpf  rsvp  
rtp
```

タブ補完

コマンドを途中までキータイプし、そのコマンドのみが補完可能なコマンドであれば、タブを入力することで補完される。

```
cisco#conf [TAB] term [TAB]  
↓  
cisco#configure terminal
```

省略形

コマンドを途中までキータイプし、そのコマンドのみが補完可能なコマンドであれば、補完することなく Enter をすれば、そのコマンドが実行できる。

```
cisco#en  
↓  
enable が実行される  
  
cisco#conf t  
↓  
configure terminal が実行される
```

モード

Cisco IOS にはいくつかのモードがあり、それぞれのモードで必要な設定を行うモーダルなシステムである。設定ファイルの概念はなく、設定はコマンドを実行するごとに内部コンフィグファイル追加されていく。各モードから、前のモードに戻るには、“exit” コマンドを実行する。

一般ユーザモード

ログインしてすぐに入るモード。ごく限られたコマンドしか使えない。

```
User Access Verification  
Password: Kerberos:      No default realm defined for Kerberos!  
Password:  
cisco>
```

特権モード

一般ユーザモードから、“enable” コマンドで特権モードに遷移する。プロンプトが # に変わり、多くのシステム閲覧コマンド (show xxxx ...) が使えるようになる。設定変更は行えない。

```
cisco>enable  
Password:  
cisco#
```

コンフィグモード

設定変更を行うモード。特権モードから“configure terminal”コマンドで遷移する。ルーティングやアクセスコントロールリスト等、様々なシステム全体に及ぶ設定変更コマンドが使える。

```
cisco#configure terminal  
Enter configuration commands, one per line. End with CNTL/Z.  
cisco(config)#
```

インターフェイスモード

各インターフェイスの設定変更を行うモード。コンフィグモードから“interface インターフェイス名”コマンドで遷移する。

IP アドレスやネットマスクの変更、アクセスコントロールリストのインターフェイスへの適用が行える。

```
cisco(config)#interface fastEthernet 0/0  
cisco(config-if)#
```

ルータモード

ルーティングの設定変更を行うモード。コンフィグモードから“router ルーティングプロトコル”コマンドで遷移する。

```
cisco(config)#router rip  
cisco(config-router)#
```

インターフェイスの確認:show interface (sh int)

```
cisco#show interfaces  
FastEthernet0/0 is up, line protocol is up  
Hardware is AmdFE, address is 000f.2309.a620 (bia 000f.2309.a620)  
Internet address is 172.21.10.11/24  
:
```

インターフェイスの確認 2:show ip interface brief (sh ip int br)

インターフェースに線が接続されているか、有効か無効か、IP アドレスは何かなどの基本的な情報を得ることができる。

```
cisco#show ip interface brief
Interface          IP-Address      OK? Method Status      Protocol
FastEthernet0/0    172.21.10.11   YES NVRAM  up           up
FastEthernet0/1    172.21.11.1   YES NVRAM  up           up
```

IP アドレスの設定 : ip address コマンド

インターフェイスモードから行う。

```
cisco#conf t
cisco(config)#int fa0/0
cisco(config-if)#ip address 192.168.1.2 255.255.255.0
                  (IP)           (netmask)
```

IP アドレスの変更や設定取り消し

IP アドレスの変更は、そのまま別の IP を設定すれば、上書きされる。

IP アドレスの無効化は、取り消したいコマンドに no を付けて実行する。

```
cisco#conf t
cisco(config)#int fa0/0
cisco(config-if)#no ip address 192.168.1.2 255.255.255.0
                  (先に設定したコマンド)
```

インターフェイスの無効化 : shutdown コマンド

インターフェイスを無効化 (down) させたい場合に用いる。

Cisco ルータでは、デフォルトでインターフェイスは down になっているので、使用時には後述の “no” を付けた “no shutdown” コマンドを実行し、インターフェイスを有効化する。

```
cisco#conf t
cisco(config)#int fa0/0
cisco(config-if)#shutdown      ( ← down)
cisco(config-if)#no shutdown   ( ← up)
```

現在の設定の閲覧

現在の設定 running-config を表示する.

```
cisco#show running-config    (show run)
```

古い OS の場合

```
cisco#write term
```

起動設定の閲覧

再起動時に読み込まれる設定ファイルを表示する.

```
cisco#show startup-config    (show start = show conf)
```

現在の設定の保存

現在の設定を、再起動時に読み込まれる設定ファイルに保存する.

```
cisco#copy running-config startup-config    (copy run start)
```

古い OS の場合

```
cisco#write memory    (wr)
```

設定の削除（1行単位）

設定の削除は、削除したい行のコマンドの行頭に“no”を付けたコマンドを実行する.

```
#cisco(config-if)#no shutdown  
(shutdown されているインターフェイスを up させる)
```

起動設定の全削除

設定の全部削除し工場出荷時に戻すには、startup-config を削除する.

```
#erase startup-config
```

C.2 ip route

Cisco IOS 上で、静的経路の設定を行う。

◆書式

```
(config)#ip route 宛先 NW アドレス 宛先ネットマスク 次ルータアドレス (メトリック)
```

◆機能説明

Cisco IOS 上で、ルーティングテーブルに静的に経路を追加する。管理者が手動で削除するまで、消去されることはない。

◆使用例 ルーティングテーブルの追加

```
cisco(config)#ip route 192.168.0.0 255.255.255.0 172.21.10.101 2
          ↑          ↑          ↑          ↑
          NW アドレス   マスク    次ルータ    距離
```

◆削除

削除する場合は、行頭に no を付けると削除される。

```
(config)#no ip route 宛先 NW アドレス 宛先ネットマスク 次ルータアドレス (メトリック)
```

C.3 show ip route

Cisco IOS 上で、ルーティングテーブルを閲覧する

◆書式

```
#show ip route
```

◆機能説明

Cisco IOS 上で、ルーティングテーブルの情報を表示する。ルーティングテーブルは、宛先ネットワークのネットワークアドレス、次ホップルータのアドレス、接続インターフェースの情報等が表示される。

◆使用例 ルーティングテーブルの追加

```
cisco1>show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set
      (↑該当する経路が設定されていない場合にデフォルトでパケットを
       転送する先)

172.21.0.0/24 is subnetted, 10 subnets
      (↑クラスB 172.21.0.0 は /24 でサブネット化されていることを示す
       VLSM の場合は、Variably subnetted, サブネット化されていない
       場合は、表示されない)

R    172.21.17.0 [120/1] via 172.21.10.17, 00:00:27, FastEthernet0/0
      (↑ 172.21.17.0/24 の次ホップは 172.21.10.17 であることを示す。
       R は RIP で動的に学習した経路であることを示す)

S    172.21.22.0 [1/0] via 172.21.10.19
      (↑ 172.21.22.0/24 の次ホップは 172.21.10.19 であることを示す。
       S は ip route コマンドで管理者が手動で設定した経路であることを示す)

C    172.21.11.0 is directly connected, FastEthernet0/1
      (↑ 172.21.11.0/24 は、このルータの FastEthernet0/1 インターフェースに
       直接接続されていることを示す。IP アドレスを付与したインターフェースが
       アップしていれば自動的に追加される)

C    172.21.10.0 is directly connected, FastEthernet0/0
      (↑ 172.21.10.0/24 は、このルータの FastEthernet0/1 インターフェースに
       直接接続されていることを示す。IP アドレスを付与したインターフェースが
       アップしていれば自動的に追加される)
```

ルーティングテーブルの経路表示情報

R	172.21.17.0	[120/1]	via	172.21.10.17, 00:00:27,	FastEthernet0/0
↑	↑	↑ ↑	↑	↑	↑
(1)	(2)	(3) (4)	(5)	(6)	(7)

(1) : 経路の学習方式の意味

C: 直接接続された経路 S: 静的経路 R: RIP で学習した経路

O: OSPF で学習した経路 B: BGP で学習した経路

(2) :宛先ネットワークのネットワークアドレス

VLSM などの場合は、172.21.15.0/26 等のようにサブネットマスク情報も
prefix 表記で表示される。

(3) : Administrative Distance

経路の管理距離（学習する方式によって決まる）

同一宛先について複数経路がある場合は値が小さいものが優先される

(4) : メトリック

経路の距離

同一宛先について複数経路があり、(3) の値が等しい場合は、
メトリック値が小さいものが優先される

(5) : 次ホップルータの IP アドレス

パケットを次に転送るべきルータの IP アドレス

(6) : 動的ルーティングで学習した経路の経過時間

動的ルーティングで経路を学習してから経過した時間

(7) : 次ホップルータのインターフェース

次ホップルータが接続されているインターフェース

C.4 show mac-address-table

Cisco IOS 上で、スイッチ（ブリッジ）の MAC アドレステーブルを確認する。

◆書式

ルータの場合

```
Router#show mac-address-table
```

スイッチの場合

```
Switch#show mac address-table
```

◆機能説明

どのポートの先に、どの MAC アドレスの端末がいるかが確認できる。

スイッチやルータで確認できるが、L2 スイッチ機能のないルータでは実行できない（MAC アドレステーブルそのものがない）。

◆使用例

```
Switch#show mac address-table
      Mac Address Table
-----
Vlan   Mac Address        Type      Ports
----  -----
  1    109a.dd4f.0df5    DYNAMIC   Fa0/9
  1    e05f.b90d.47b5    DYNAMIC   Fa0/1
```

スイッチの 9 番ポートの先に、10:9A:DD:4F:0D:F5 の MAC アドレスを持つ端末が存在する。

スイッチの 10 番ポートの先に E0:5F:B9:0D:47:B5 の MAC アドレスを持つ端末が存在する。

C.5 show arp

Cisco IOS 上で、ルータの ARP テーブルを確認する。

◆書式

```
Router>show arp
```

◆機能説明

どの IP アドレスが、どの MAC アドレスかを確認する。

ただし、ARP (Address Resolution Protocol) でこれまで確認できたものか、もしくは静的に arp コマンドで管理者が設定したもののみ確認できる。

◆使用例

```
Router>show arp
```

Protocol	Address	Age (min)	Hardware Addr	Type	Interface
Internet	172.21.22.2	103	e0cb.4e7a.af48	ARPA	FastEthernet1
Internet	172.21.22.9	-	e05f.b90d.47b5	ARPA	FastEthernet1
Internet	172.21.22.10	3	04c5.a47e.a140	ARPA	FastEthernet1
Internet	192.168.0.1	14	0015.17ee.73a0	ARPA	FastEthernet0
Internet	192.168.0.189	-	e05f.b90d.47b4	ARPA	FastEthernet0
Internet	192.168.0.225	13	106f.3f04.2fd0	ARPA	FastEthernet0

Address が IP アドレスを、

Hardware Addr は MAC アドレスを示す。

C.6 show vlan

Cisco IOS 上で、ルータ・スイッチの VLAN 設定状況を確認する。

◆書式

```
Switch>show arp
```

◆機能説明

どのポートが、どの VLAN に所属しているかを確認する。

◆使用例

```
Switch>show vlan
```

VLAN	Name	Status	Ports
1	default	active	Fa0/1, Fa0/2, Fa0/3, Fa0/4
7	VLAN0007	active	
9	group09	active	Fa0/18, Fa0/19
12	group12	active	Fa0/20, Fa0/24
13	teststp	active	

ポート 1-4 が VLAN 1 (デフォルト), 18,19 が VLAN 9,
20, 24 が VLAN 12 である。

C.7 show interface status

Cisco IOS 上で、スイッチのポート（インターフェース）の状況を確認する。

◆書式

```
Switch>show interface status
(sh int status)
```

◆機能説明

静的 VLAN、ポートの up/down (有効・無効、接続有り/無し) などが分かる。

◆使用例

```
Switch>show int status
```

Port	Name	Status	Vlan	Duplex	Speed	Type
Fa0/1		connected	1	a-full	a-100	10/100BaseTX
Fa0/2		notconnect	1	auto	auto	10/100BaseTX
Fa0/3		connected	1	a-full	a-100	10/100BaseTX
Fa0/4		notconnect	1	auto	auto	10/100BaseTX
Fa0/5		connected	1	a-full	a-100	10/100BaseTX
Fa0/6		notconnect	1	auto	auto	10/100BaseTX
Fa0/7		notconnect	1	auto	auto	10/100BaseTX
Fa0/8		notconnect	1	auto	auto	10/100BaseTX
Fa0/9		connected	1	a-full	a-100	10/100BaseTX
Fa0/10		notconnect	1	auto	auto	10/100BaseTX
Fa0/11		notconnect	1	auto	auto	10/100BaseTX
Fa0/12		notconnect	1	auto	auto	10/100BaseTX
Fa0/13		notconnect	1	auto	auto	10/100BaseTX
Fa0/14		notconnect	1	auto	auto	10/100BaseTX
Fa0/15		notconnect	1	auto	auto	10/100BaseTX
Fa0/16		notconnect	1	auto	auto	10/100BaseTX
Fa0/17		notconnect	1	auto	auto	10/100BaseTX
Fa0/18		notconnect	1	auto	auto	10/100BaseTX
Fa0/19		notconnect	1	auto	auto	10/100BaseTX
Fa0/20		notconnect	1	auto	auto	10/100BaseTX
Fa0/21		notconnect	1	auto	auto	10/100BaseTX
Fa0/22		notconnect	1	auto	auto	10/100BaseTX
Fa0/23		notconnect	1	auto	auto	10/100BaseTX
Fa0/24		notconnect	1	auto	auto	10/100BaseTX

connected は端末が接続中であり、a-full、a100 は automatic

(自動認識) で全二重, 100M で接続中であることを示す.
not connected は接続されていない, あるいは
電源が入っていないことを示す.
VLAN は全て 1 である.

C.8 show ip interface brief

Cisco IOS 上で、ルータ・スイッチのインターフェースの IP アドレスとリンク状況を確認する。

◆書式

```
Router>show ip interface brief  
(sh ip int brief)
```

◆機能説明

ルータ・スイッチのインターフェースの IP アドレス、up/down (有効・無効、接続有り / 無し) などが分かる。

C.9 show ip nat translations, ip nat

Cisco IOS 上で、NAT (NAPT, IP masquerading) の設定、および、NAT テーブル確認を行う。

◆書式

静的 NAT の設定

```
Router(config)#ip nat inside source static tcp 172.21.22.100 80 222.229.69.3 8080
```

内側 (inside) の (送信元)IP アドレス 172.21.22.100、TCP ポート 80 を、
外側 (outside、インターネット側) に対して、222.229.69.3 TCP ポート 8080 で
公開する。

NAT テーブルの確認

```
Router#show ip nat translations
```

Pro	Inside global	Inside local	Outside local	Outside global
tcp	222.229.69.3:8080	172.21.22.100:80	---	---

静的 NAT の設定削除

```
Router(config)#no ip nat inside source static tcp 172.21.22.100 80 222.229.69.3 8080
```

NAT テーブルから削除

```
Router#clear ip nat translations *
```

◆機能説明

どのポートが、どの VLAN に所属しているかを確認する。

◆使用例

静的 NAT の設定

```
Router(config)#ip nat inside source static tcp 172.21.22.100 80 222.229.69.3 8080
```

内側 (inside) の (送信元)IP アドレス 172.21.22.100、TCP ポート 80 を、
外側 (outside、インターネット側) に対して、222.229.69.3 TCP ポート 8080 で
公開する。

すなわち外から、222.229.69.3 TCP ポート 8080 へのアクセスがあれば、
内側のサーバ 172.21.22.100、TCP ポート 80 へ転送
(ポートフォワーディング) する。

```
Router#show ip nat translations
```

Pro	Inside global	Inside local	Outside local	Outside global
-----	---------------	--------------	---------------	----------------

tcp 222.229.69.3:8080	172.21.22.100:80	---	---
-----------------------	------------------	-----	-----

NAT 設定の確認
