2 章. ネットワーク TCP/IP・HTTP

序文

今日におけるネットワーク通信のデファクトスタンダードとなったTCP/IPについて取り扱う。TCP/IPそれ自体は、無条件に使用可能なものとして取り扱われることが多いが、その性質を学ぶことによりネットワーク技術の一般的な方法論を理解することができる。

OSI参照モデル

- **第7層** アプリケーション層
  TELNET、SSH、HTTP、SMTP、POP、SSL/TLS、FTP・・・
- **第6層** プレゼンテーション層 - データの表現方法
- **第5層** セッション層 - 通信プログラム間の通信の開始から終了までの手順
- **第4層** トランスポート層
  例. TCP、UDP
- **第3層** ネットワーク層
  例. IPv4、IPv6、ICMP、IPSec
- **第2層** データリンク層
  例. ARP、PPP、Ethernet、IEEE 802.11 無線LAN
- **第1層** 物理層
  例. 物理的な接続。1000BASE-T、ADSL、光ファイバーケーブル、ツイストペアケーブルなど

プロトコルの階層化

ISOは通信プロトコルを設計するときの指標として、OSI 参照モデルを提唱しました。通信に必要な昨日を７つの階層に分け、昨日を分割することで、複雑になりがちなネットワーク・プロトコルを単純化するためのモデルである。

各階層は、下位の層から特定の機能を利用していき、上位の層は下位の層の機能を"前提"とした上でサービスを提供できる。

上位の層と下位の層の間でサービスのやりとりを行うときの取り決めを "インターフェイス" と呼び、同じ階層での通信での取り決めを "プロトコル" という。

TCP/IP

TCP は Transmission Control Protocol 、 IP は Internet Protocol を表します。TCP/IPはインターネット上の通信を実現するためのプロトコルであり、今日最も一般的に用いられているプロトコルと言える。

TCP/IP は歴史的に、大学や研究機関で利用されてきました。そのため、同じく大学や研究機関で利用されていた UNIX 系の OS と相性がよくこれらの OS とともに発展してきました。

TCP/IPの標準化・RFC

TCP/IP のプロトコルの標準化は IETF で議論され標準化されています。標準化しようとするプロトコルは RFC (Request For Comments) と呼ばれるドキュメントとしてインターネット上で公開されています。

RFCとなったドキュメントには番号付けされており、例えばIPの仕様を規定しているのはRFC791、TCPの仕様を規定しているのはRFC793です。

代表的なRFC

protocol	rfc
IPv4	**RFC791、RFC919、FFC922**
IPv6	**RFC2460**
ICMP	**RFC792**
ARP	**RFC826**
TCP	**RFC793**
UDP	**RFC768**

物理層

Ethernet、FTTH

データリンク層

NICなどのドライバが相当する。

IP (ネットワーク層)

ネットワークをまたいでパケットを転送し、インターネット全体にパケットを送り届けるためのプロトコルである。IPを使っていればデータリンク層の存在を抽象化することができる。すなわち、データリンク層が何であってもIPを用いていればIP同士で通信ができるということ。

IPには大きく３つの役割がある。

1. IPのアドレス
2. IPアドレスの先へのパケットの転送
3. IPのパケットの分割と再構築の処理

IPアドレス

ネットワーク通信では通信相手の識別のためにアドレスという識別子を用いる。TCP/IPで通信するすべてのホストやルーターには必ずIPアドレスが設定されていなくてはならない。

IPv4

ネットワーク部とホスト部

IPアドレスはネットワーク部とホスト部に分けられる。
例えば 192.168.0.1/24 とあった場合、*192.168.0* までがネットワーク部、残りの *1* がホスト部といった具合である。

ブロードキャストアドレス

IP アドレスのホスト部のビットをすべて *1* にしたものは、ブロードキャストアドレスとして扱われる。
例えば 192.168.0.255/24 はブロードキャストアドレスである。

ICMP (Internet Control Message Protocol)

IP パケットの配送中に以上が発生した場合などに、パケットの送信元に異常を知らせるために使われるプロトコル。ping もこれを使っている。主としてネットワークの診断のために利用される。

演習1

ping を google.com に対して行ってみよ。

ARP (Address Resolution protocol)

IPアドレスから接続すべきホストのMACアドレスを見つけるときに使用される。ARP要求パケットをブロードキャストし、該当するホストがMACアドレスを応答することにより、接続するべきMACアドレスがわかる。

DNS (Domain Name System)

DNS はホスト名から IP アドレスを検索するシステムである。 www.cyberagent.co.jp というアドレスにブラウザからアクセスしたとき、インターネット上から www.cyberagent.co.jp を名乗るWebサーバーを発見し、Webサイトの情報を受け取れるのはこのDNSサーバーのしくみによる。
また

> www.cyberagent.co.jp

のようなドメイン名の *jp* の部分を特に TLD(Top Level Domain)と呼ぶ

DNSレコードのタイプ

タイプ	内容
A	IPv4アドレス
AAAA	IPv6アドレス
MX	メールサーバーのIPアドレス
TXT	テキスト
SOA	DNS権威サーバーが利用する情報

演習2

www.google.co.jp のドメイン名を解決し、IPアドレスを表示してみよ。

DHCP (Dynamic Host Configuration Protocol)

ホストごとにIPアドレスを設定するのは面倒なので、それを肩代わりしてくれるプロトコルである。DHCPサーバーにブロードキャストを用いて問い合わせ、IPアドレスなどの設定を行う。

NAT (Network Address Translation) と NAPT (Network Address Port Translation)

NATに対応しているゲートウェイが、ネットワーク内部でアドレスの変換テーブルを用いてIPの変換処理を行うこと。また、ポートとIPアドレスの組み合わせを用いて同様の変換を施すものをNAPTと

呼ぶことがある。近頃は両者を指して単にNATと呼ぶことが多い

TCPとUDP

UDP (User Datagram Protocol)

UDPは複雑な機能を提供せず、IPを用いてコネクションレスの通信を行うために使用されます。TCPと違ってコネクションを確立するまでのコストがないため、リアルタイムの通信やパケット損失を許容できるような通信を実現するために使用されます。

TCP (Transmission Control Protocol)

TCPはIPというコネクションレス型のコネクションの上にコネクションの機能と、再送要求の機能を含めたプロトコルである。この仕組みにより高い信頼性の通信を実現することができる。

3-way ハンドシェイク

https://ja.wikipedia.org/wiki/3ウェイ・ハンドシェイク

クライアントは接続要求(SYNパケット)を送信し確認応答をまち、確認応答が完了した場合に、通信可能になります。
終了した場合は切断処理を行います。(FINパケット)

セグメント単位での送信と再送要求

TCPでは送信したデータが受信ホストに到達したとき、受信ホストは送信ホストにデータを到達したことを知らせます。これを確認応答(ACK - Acknowledgement)といいます。TCPではこれをデータのセグメント単位で繰り返します。また、データが喪失した場合や、応答がない場合にはデータを再送することになります。このようにして通信の信頼性を担保しています。

HTTP

HTTP(Hypertext Transfer Protocol)とは、WebサーバとWebブラウザの間で、Web情報をやりとりするためのプロトコル（通信規則）です。私たちが普段、ホームページで情報収集したり、ブログを読んだりする時、このHTTPを使ってサーバとクライアント（ユーザ）間でやり取りが行われます。
HTTPの特徴の1つは動作がとてもシンプルな点です。情報のやり取りは常に、クライアント（Webブラウザなど）が要求を出し、サーバが応答を返します。1つの要求（リクエスト）には1つの応答（レスポンス）を返すルールになっていて、どちらかが多くなることはありません。

URL

https://example.com/abc

HTTPメソッド

メソッド	性質
GET、HEAD	べき等かつ安全
PUT、DELETE	べき等だが安全でない
POST	べき等でも安全でない

HTTPヘッダ

Webコンテンツの伝送に用いられるHTTPで、メッセージの前半にある制御情報を記した領域のこと。WebサーバやWebブラウザが相手方に伝えたい情報を格納する部分で、利用者の目には直接触れない。

ex) App Center API

Statusコード

RFC 7231に更新版の仕様書があります。

1. 情報レスポンス (100–199)
2. 成功レスポンス (200–299)
3. リダイレクト (300–399)
4. クライアントエラー (400–499)
5. サーバエラー (500–599)

curl

```
curl https://www.google.co.jp
```

演習3

1. `curl` で Slack にメッセージを送ってみよ。
2. HEADを標準出力してみよ。

業務的な話

REST

REST(**RE**presentational **S**tate **T**ransfer)はWebサービスの設計モデルです。RESTなWebサービスは、そのサービスのURIにHTTPメソッドでアクセスすることでデータの送受信を行います。

RESTful(RESTの設計原則)

RESTは設計に際し以下を設計原則とするよう提言されています。

1. アドレス指定可能なURIで公開されていること
2. インターフェース(HTTPメソッドの利用)の統一がされていること
3. ステートレスであること
4. 処理結果がHTTPステータスコードで通知されること

gRPC

Googleが開発を開始したオープンソースのリモートプロシージャコール (RPC) システムである。gRPCは、HTTP/2をトランスポートとして利用し、Protocol Buffersをインタフェース記述言語として利用する。gRPCは認証や双方向のストリーミング、フロー制御、ブロッキングとノンブロッキングのバインディング、取り消しとタイムアウトといった機能を提供している。

gRPCのメリット

出典: gRPCって何？ - Qiita

- JSON-RPC と同様に、REST APIよりはURL設計が楽そう
- proto ファイルにAPI仕様を強制的に明文化できる
- REST APIやJSON-RPCとは違ってパラメーターをきちんと型付けした状態で扱える安心感
- GoやJavaなどの静的型付け言語と相性がよさそう
- ボイラープレートコードを自動生成でき、品質も上がりそう
- 自動生成が逆に扱いづらいというケースがあるのか現時点では不明
- HTTP/2を活かした高速な双方向ストリーミング通信が可能

gRPCのデメリット

- REST APIほどは周辺のエコシステム（デバッグツールやAPIドキュメント生成など）が揃っていなさそう
- データがシリアライズされるのでデバッグはやりづらそう
- grpc-gateway を使うとREST APIを受け付けるリバースプロキシサーバーを立てられるみたい
- JSON-RPCなどと比べて学習コストは大きそう

てっくぼっと記事(gRPC)

Unityプロジェクトでの gRPC導入方法について
gRPCのiOS実機環境における速度検証について

バイナリ形式

JSON

JavaScript Object Notation（JSON、ジェイソン）はデータ記述言語の1つです。軽量なテキストベースのデータ交換用フォーマットでありプログラミング言語を問わず利用できます。

Protocol Buffer

インタフェース定義言語(IDL)で構造を定義する通信や永続化での利用を目的としたシリアライズフォーマットであり、Googleにより開発されている。

特徴

1. **静的な構造化データを対象とする**:
   シリアライズを行うためのデータは、事前にその構造を定義しておく必要がある。構造の定義には、専用の書式とデータ型を用いる
2. **拡張可能**:
   元のデータ構造を変更することなく、データ構造を拡張することが可能。例えば、ある開発者が作成したデータ構造に対して、別の開発者が独自のフィールドを追加して利用するといったことができる
3. **コンパクトなバイナリ形式を採用**:
   XMLのようなテキスト形式を用いず、なるべくサイズがコンパクトになるよう設計されたバイナリ形式を採っているため、最低限のリソースしか使用しないうえ、シリアライズ/デシリアライズに要する処理時間も非常に少なくて済む
4. **プログラミング言語に依存しない**:
   バイナリ形式がプログラミング言語から独立して設計されているため、Javaのシリアライズ方式などとは異なり、プログラミング言語に一切依存しない
5. **データアクセスコードを自動生成**:
   Protocにより、様々な言語にコンパイル可能

Message Pack

効率の良いバイナリ形式のオブジェクト・シリアライズ フォーマット。JSONの置き換えとして使うことができ、様々なプログラミング言語をまたいでデータを交換することが可能です。
JSONと比べると、保存した状態の可読性を犠牲にする代わりに、より早くて小さいフォーマットになっています。
MagicOnionではgRPC + MessagePackでリアルタイム通信が実装されている。

てっくぼっと記事(MessagePack)

- Unity & サーバー間通信でのMessagePack導入奮闘記(Unity編)

参考図書

- 3分間ネットワーク基礎講座
- オンラインゲームのしくみ Unityで覚えるネットワークプログラミング
- TCP/IP Illustrated, Volume 1: The Protocols (Addison-Wesley Professional Computing Series)