この文書について

この文書は、東京大学大学院理学系研究科物理学専攻の原子核実験グループのコロキウム「情報セキュリティとクラッキングの実際」に先立ち、予備知識を伝えるために作成したものです。

Power Point のスライドショーにて、クリックしてリンクをたどりながら進むように作成されています。また、一部Web ページへのリンクが含まれています。

この文書の私的利用は自由ですが、筆者は記載内容の正確さ等に関して一切責任を負いませんので、ご了承ください。

2002年9月25日

東大理　　民井　淳

あなたのネットワーク習熟度(あるいはおたく度)チェック

聞いたことが無いもの　0 点
聞いたことはあるが、説明できないもの　1 点
~~どのようなものか~~(おおまかに)説明できるもの　2 点

1) DNS
2) MAC Address
3) IP
4) RFC
5) DHCP
6) CSMA/CD
7) HUB, Switch
8) NAT
9) Well-known Ports
10) Subnet Mask
11) 3-Way Handshake
12) TCP
13) Diffie Hellman
14) 10Base-5
15) OSPF

独断と偏見による採点表

[点数]
0- 3: 計算機をあまり使わない一般人として普通
4- 6: 計算機をあまり使わない一般人としてはやや詳しい
7-11: 理系の研究者として普通
　　計算機を管理するにはもう一歩 Study を
12-17: 個人計算機の管理者として一般的
18-24: 共有計算機/ネットワーク管理者レベル
25-28: かなりのエキスパート
　　NEX の計算機管理に是非加わって欲しい
29-30: 立派なおたくです。脱帽！

この程度の質問では生ぬるいと思った方:
　身も心も完全なおたくです。
　社会から疎外されないように気をつけましょう。

ネットワークのイロハ

― 箱の中で何が行われているのか？ ―

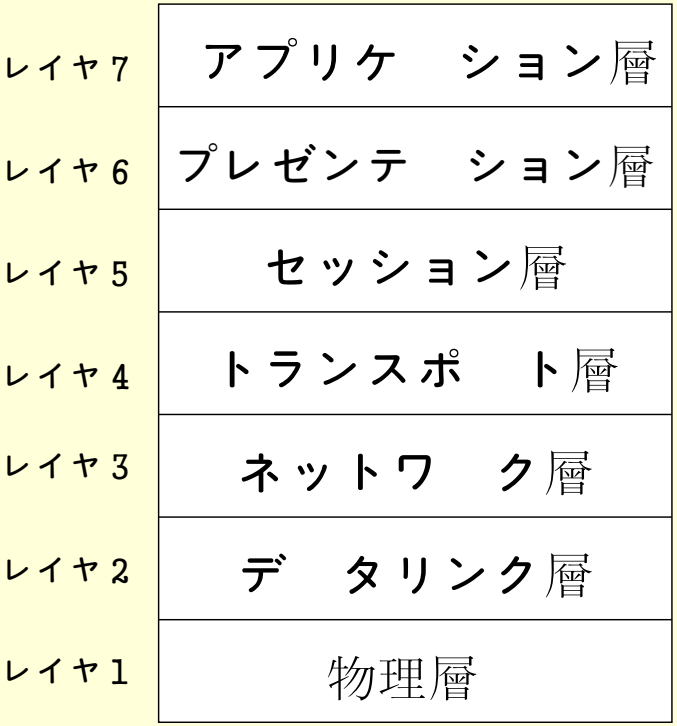
東大理　　民井　淳

目次

OSI 基本参照モデル

(Open Systems Interconnections, basic reference model)

OSI　照モデルの７つのレイヤ



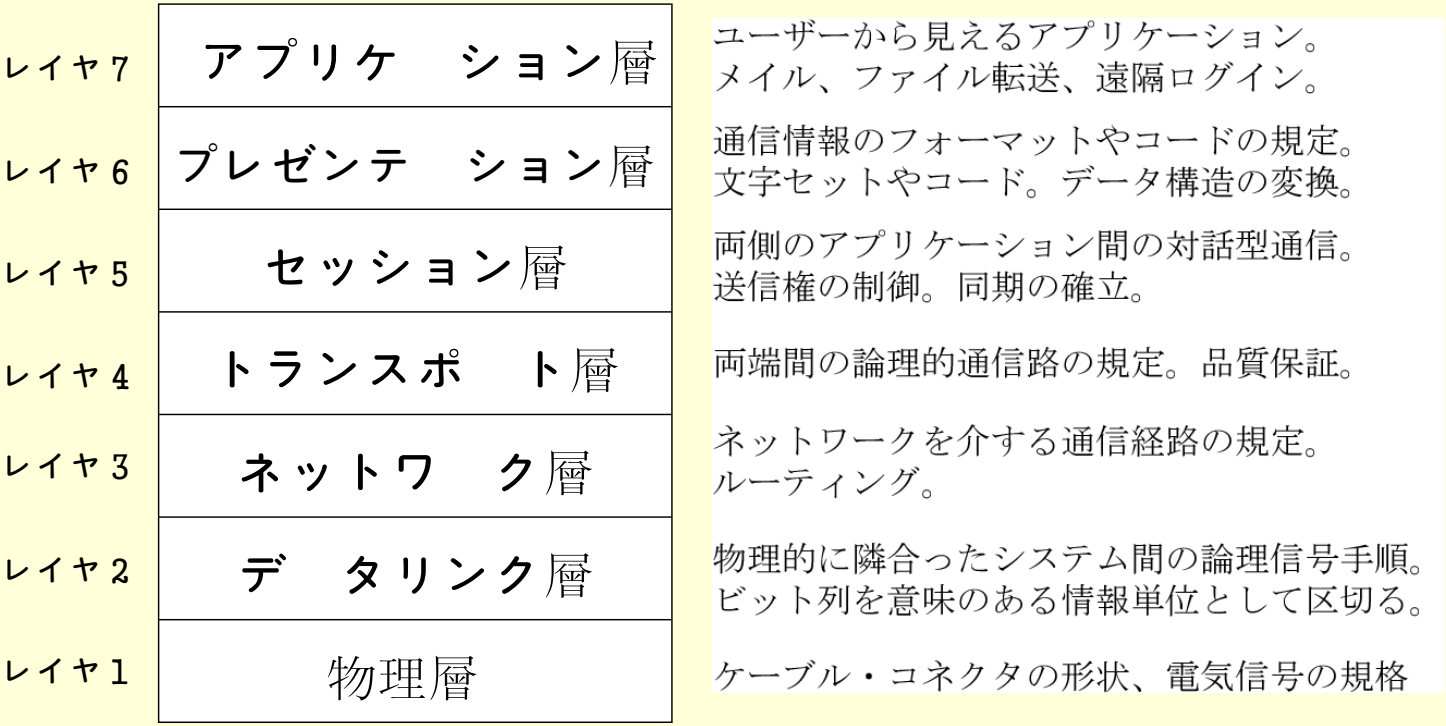
→各層の説明

→TCP/IPの例

OSI 基本参照モデル

(Open Systems Interconnections, basic reference model)

OSI　照モデルの７つのレイヤ

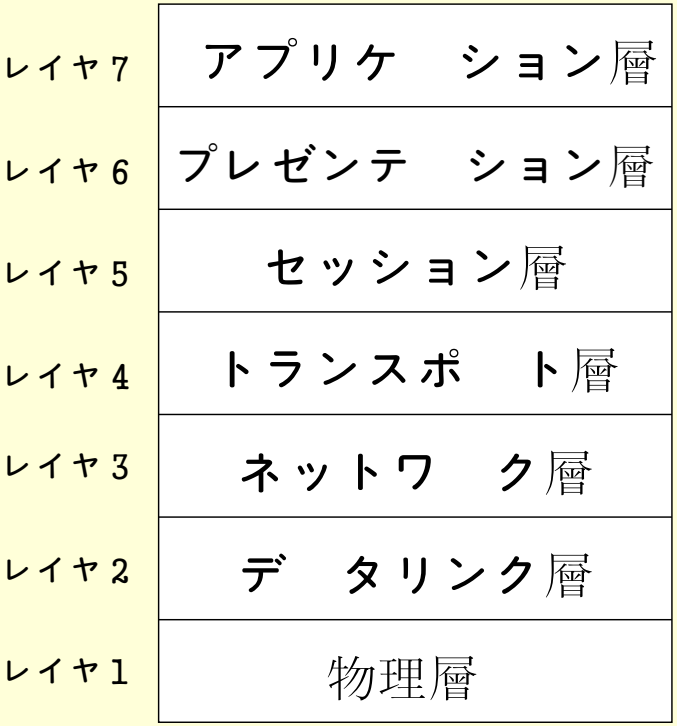


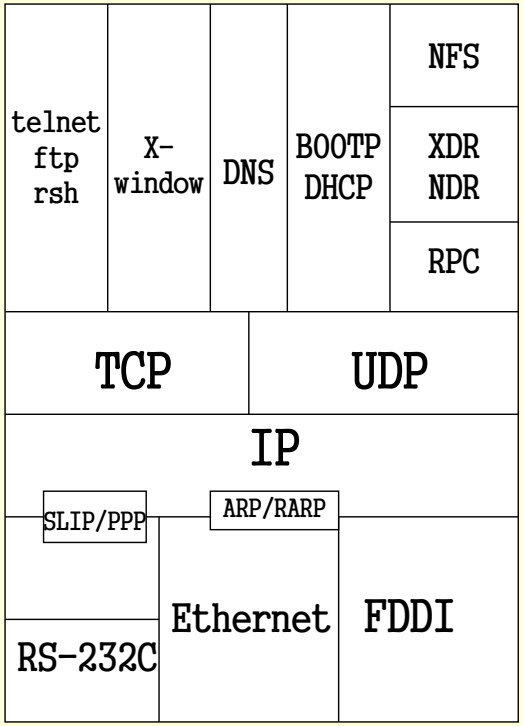
TCP/IPの例

OSI 基本参照モデル

(Open Systems Interconnections, basic reference model)

OSI　照モデルの７つのレイヤ





TCP/IPの場合の例

各層の説明

RFC (Request for Comments)

ISOC (Internet Society) の下部組織である IETF (Internet Engineering Task Force) がインターネット技術に関して発行する文書。

標準化の過程によって、標準プロトコル、ドラフト標準プロトコルなど6つの段階に分かれる。

例:

RFC 791 … IP (Internet Protocol)

RFC 793 … TCP (Transmission Control Protocol)

RFC 2616 … HTTP (HyperText Transfer Protocol)

RFC 3000 … Internet Official Protocol Standards, 2001.

→余談

RFC (余談)

毎年4月1日には数件のジョークRFCが発行されるのが通例。

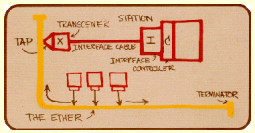
伝書鳩プロトコル(RFC 1149など)

コーヒーポット制御プロトコル(RFC 2324)

など。

Ethernet

IEEE802.3



Ethernet = Ether (エーテル)と Network の合成語

歴史

- 1973年、ゼロックス社PARCコンピューター科学研究所の Robert M. Metcalfe 博士によって発明された。 →論文
- DEC/Intel/Xeron の規格としてまとめられた (DIX-Ethernet)。
- Ethernet をもとに IEEE 802.3 (CSMA/CD) 標準 LANが規格化。

特徴

- CSMA/CD 方式
- MAC (Media Access Control) Address を使用して通信

→ MAC Frame

Ethernet

主な種類	ケーブル	最大長	構造
10Mbps			
10Base5 (thick)	thick coax.	500m	bus
10Base2 (thin, cheaper-net)	thin coax.	185m	bus
10BaseT	UTP (3-5)	100m	star
100Mbps			
100Base-TX	UTP (5)	100m	star
100Base-FX	Fibre-optic	20km	star
1Gbps			
1000Base-T	UTP (5e)	100m	star
1000Base-SX	Fibre-optic (MMF)	550m	star
1000Base-LX	Fibre-optic (SMF)	5km	star

10Gbps Ethernet (IEEE Std 802.3ae) accepted, June-2002

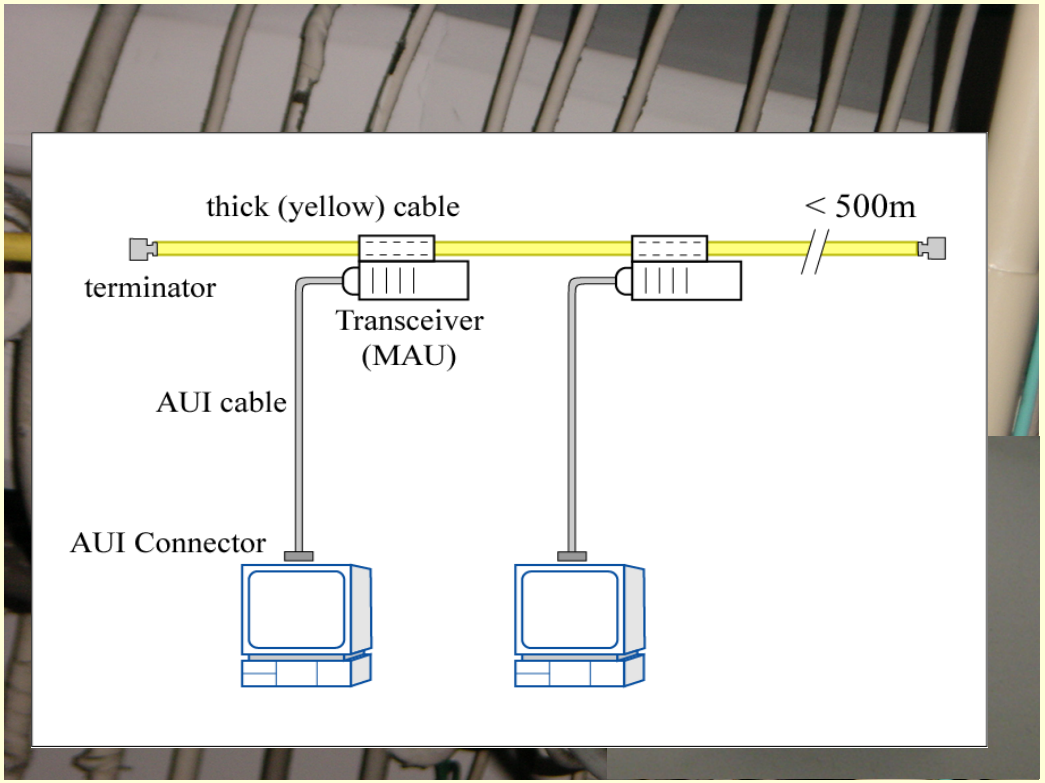
Ethernet

10Base5

- Thick “yellow” cable (2重シールド同軸、特性インピーダンス50Ω)
  N型コネクタ
- Transceiver (MAU), AUI (transceiver) cable を使って接続
- 終端に抵抗を付ける
- 最大 100 station/segment

- 10 Mbps
- Base Band 方式
- 最長 500 m

- マンチェスタ符号化



Ethernet

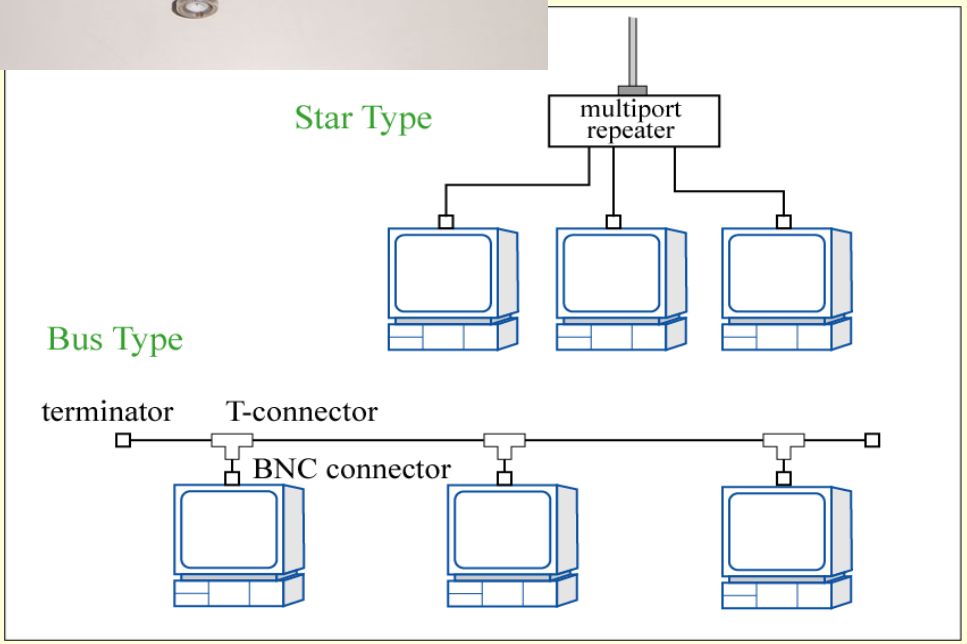
10Base2

- Thin cable (同軸、特性インピーダンス50Ω)
  BNC コネクタ
- ディージーチェーン(bus型) or マルチポートリピータ(star型)
- 終端に抵抗を付ける
- 最大 30 station/segment

- 10 Mbps
- Base Band 方式
- 最長 185 m

- マンチェスタ符号化

10Base5より低コスト (cheaper net)
ケーブルの引き回しが容易



Ethernet

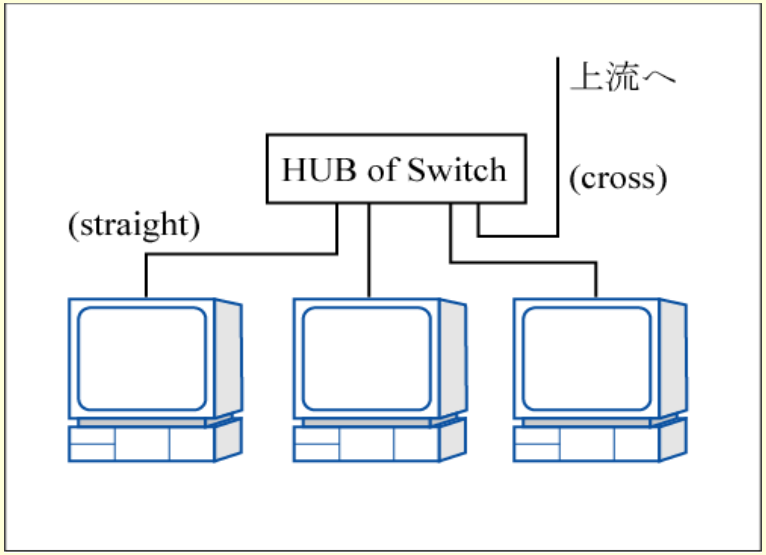
10BaseT

- UTP (Unshield Twisted Pair) cable（特性インピーダンス100Ω）
  カテゴリ 3-5
  4対8芯 (実際に使用しているのは2対)
  RJ45 モジュラーコネクタ
  ストレート接続 (Hub-PC 間)、クロス接続 (Hub-Hub 間)
- Star 型
- 差動式信号

RJ-45

- 10 Mbps
- Base Band 方式
- 最長 100 m (hub-station 間)

- マンチェスタ符号化



最も低コスト
ケーブルの引き回しが容易
全2重化(full-duplex)通信が可能 (20Mbps)

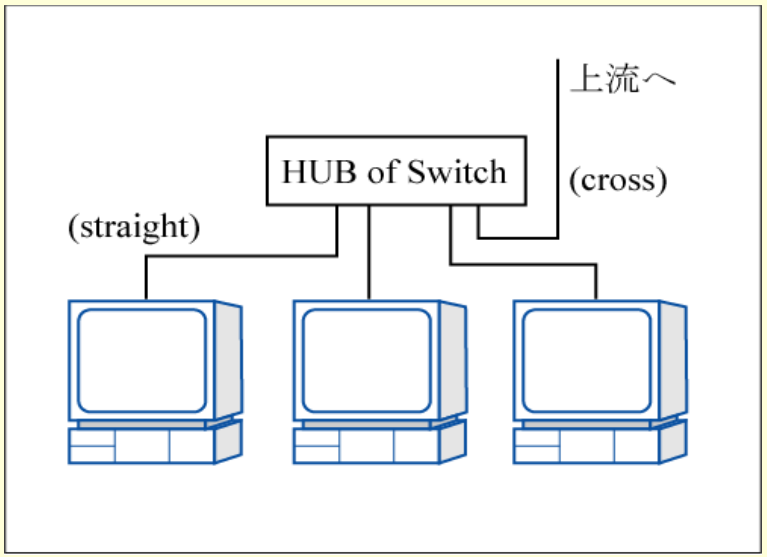
Ethernet

100BaseTX
(Fast Ethernet)

RJ-45

- UTP (Unshield Twisted Pair) cable（特性インピーダンス100Ω）
  カテゴリ 5
  4対8芯 (実際に使用しているのは2対)
  RJ45 モジュラーコネクタ
  ストレート接続 (Hub-PC 間)、クロス接続 (Hub-Hub 間)
- Star 型
- 3値信号

- 100 Mbps
- Base Band 方式
- 最長 100 m (hub-station 間)
- 4B/5B, MLT-3 符号化
- HUB を中継遅延によりクラス分け



低コスト。10BaseT の資産が使える。
ケーブルの引き回しが容易
全2重化(full-duplex)通信が可能 (200Mbps)

Ethernet

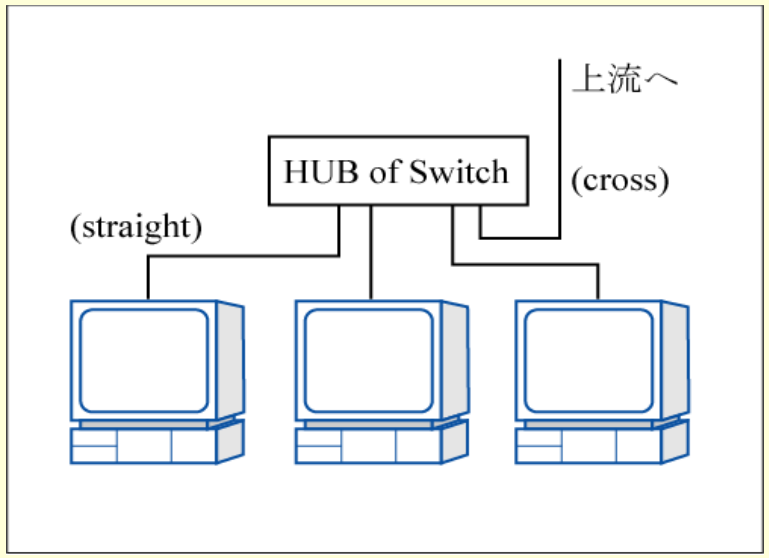
1000BaseT

(Giga Bit Ethernet)

RJ-45

- UTP (Unshield Twisted Pair) cable（特性インピーダンス100Ω）
  カテゴリ 5 (5e)
  4対8芯 (実際に使用しているのは4対)
  RJ45 モジュラーコネクタ
  Hub-PC、Hub-Hub 間 自動検出。
- Star 型
- 5値信号、4対による並列通信

- 1000 Mbps
- Base Band 方式
- 最長 100 m (hub-station 間)
- 8B1Q4 符号化
- Carrier Extension, Frame Burst

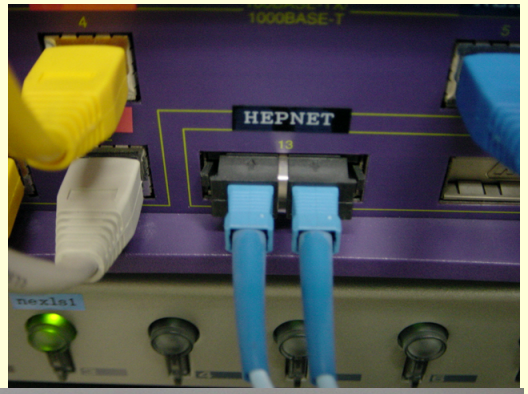


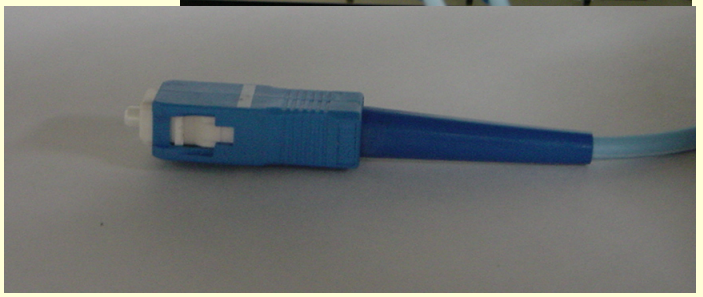
低コスト。10/100BaseT の資産が使える。
全2重化(full-duplex)通信が可能 (2000Mbps)
(但し同じケーブルを双方向に使用する)

Ethernet

1000BaseSX

(Giga Bit Ethernet)





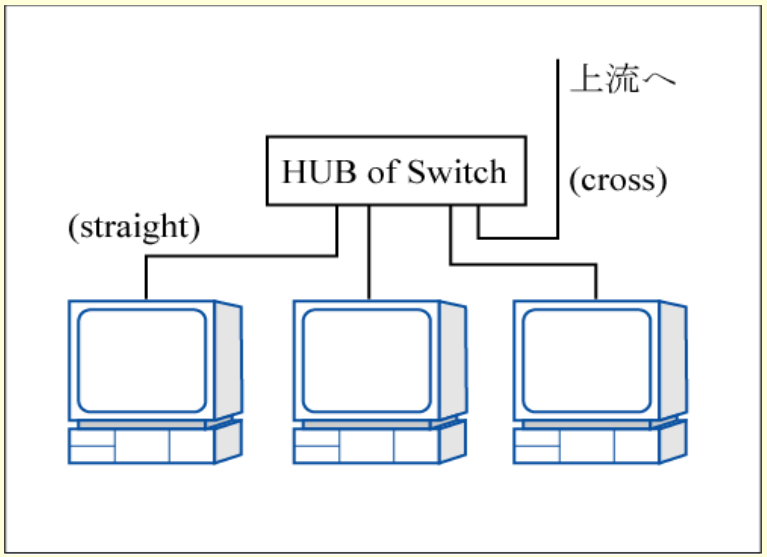
ST Connector

- Fibre-optic cable（Multi Mode） 850nm
  ST コネクタ
  Hub-PC、Hub-Hub 間 自動検出。
- Star 型

- 1000 Mbps （1.25Gbaud)
- Base Band 方式
- 最長 550 m (hub-station 間)

- 8B10B 符号化

ノイズに強い。
全2重化(full-duplex)通信が可能 (2000Mbps)



Ethernet

1000BaseLX

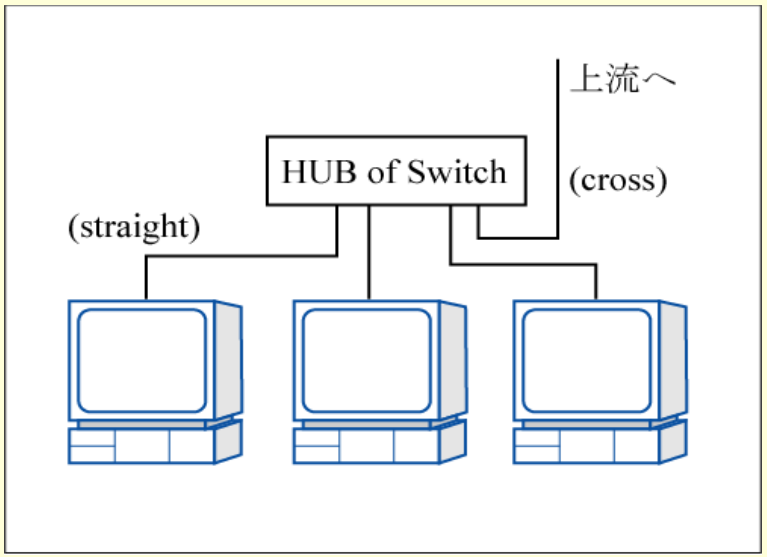
(Giga Bit Ethernet)

- Fibre-optic cable（Single Mode or Multi Mode）1300nm
  ST コネクタ
  Hub-PC、Hub-Hub 間 自動検出。
- Star 型

- 1000 Mbps （1.25Gbaud)
- Base Band 方式
- 最長 5 km (hub-station 間, SMF)

- 8B10B 符号化

ノイズに強い。
長距離接続可能。
全2重化(full-duplex)通信が可能 (2000Mbps)



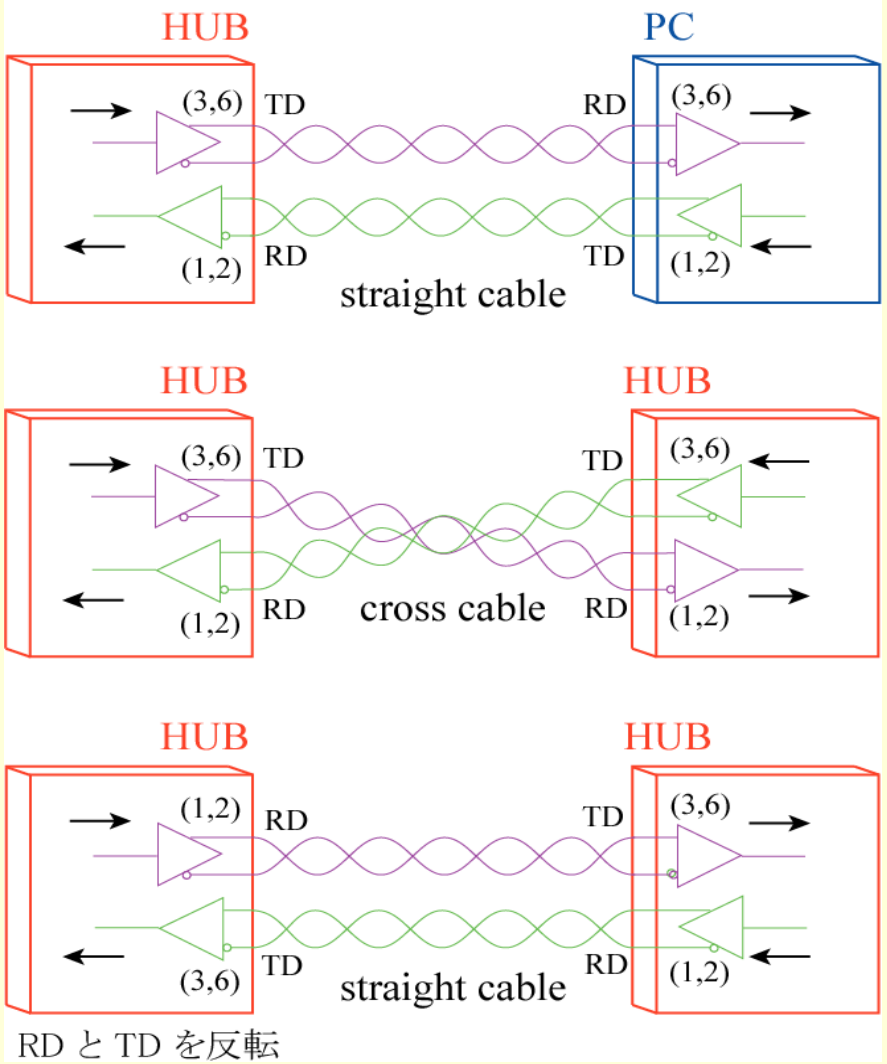
Ethernet

Straight/Cross Connection

RD (受信)と TD (送信)のピンを正しく接続する。

1. HUB ⇔ PC
   Straight Cable で接続

2. HUB ⇔ HUB
   - Cross Cables で接続
   - 片側の HUB の RD, TD を反転し、Straight Cable で接続

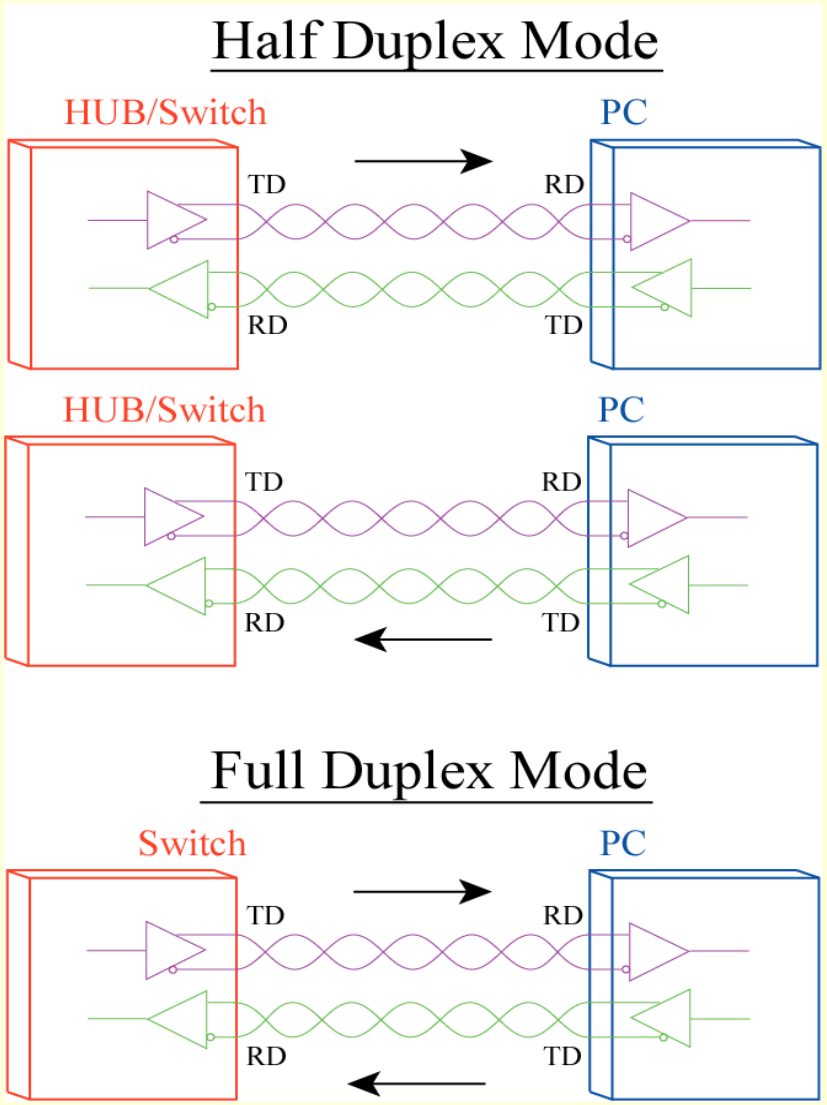


Ethernet

Half Duplex and Full Duplex

1. Half Duplex (半2重) Mode
   - 送信、受信のどちらか一方を行う。
   - Collision Domain に3台以上接続している場合には、CSMA/CD 制御を行うので必須。

2. Full Duplex (全2重) Mode
   - 送信、受信を同時に行うことができる。
   - Switch-PC 間 or Switch-Switch 間の2ノード間のみで通信を行う場合に可能。
   - 通信速度は最大で2倍になる。



CSMA/CD

CSMA/CD
= Carrier Sense Multiple Access with Collision Detection

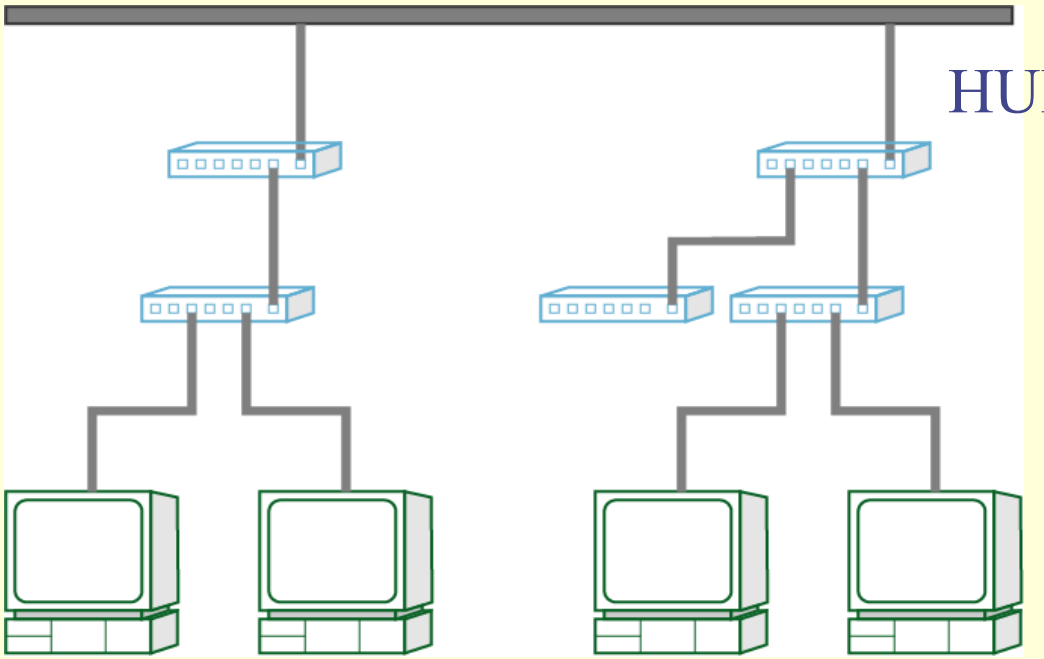
Carrier Sense … キャリア(信号)を監視し、データが流れていない時にアクセス。

Multiple Access … 複数のノードがアクセスできる。

Collision Detection … 信号の衝突を検出する。

→ CSMA/CD の詳細

Four-Repeater Rule



- Ethernet では、始点と終点の間に4台を超えるリピータ(HUB)が入ってはならない。
- → 幹線から端末までの間に入るHUBの数を2台以下に制限すれば、このルールは必ず満たされる。
  (但し間にSwitchが入るとこの数はリセットされる)

Ethernet

HUB, Switch, and Router

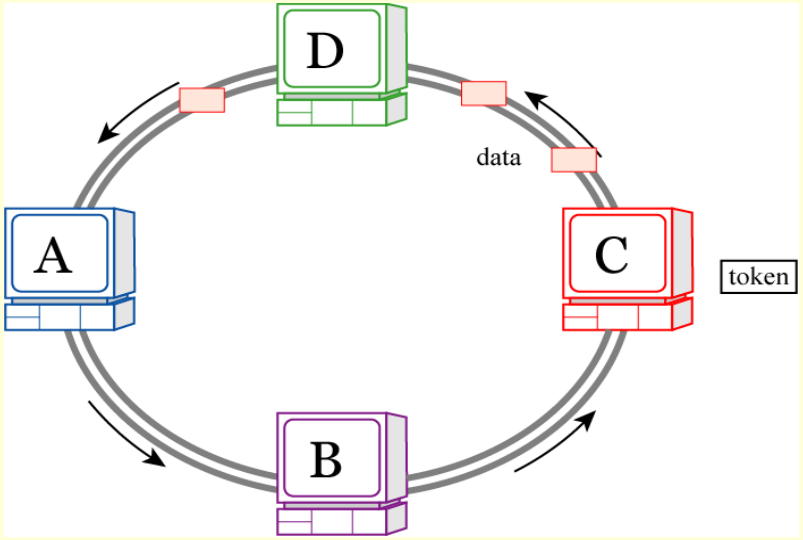
- HUB (multiport repeater)
  電気信号の整形を行う。
  各ポートからの入力情報を全てのポートに送信する。
  CSMA/CD の制御範囲(コリジョンドメイン)を広げる。
  → HUB
- Switch
  各ポートに接続されている MAC Address を記憶し、必要なポートのみに信号を送信する(Bridge)。
  各ポートのデータを一旦記録してから転送する。
  → 速度の異なるネットワーク間を接続できる。
  → コリジョンドメインを切り分けることができる。
  → Switch
  → Layer-3 Switch
- Router
  プロトコルの異なるネットワーク間(WAN⇔LAN間など)を接続できる。
  パケットフィルタリング、アドレス変換
  → Router

FDDI
(token passing network)

ANSI X3T12, ISO 9314

Fiber Distributed Data Interface (FDDI)

- 光ファイバによる2重リング構成 (UTPの規格もある:TP-DDI)。
  片方のリングは障害時に使用。
- Token Ring 形式。Token を持っているノードのみがデータを送信できる。
- Timed Token Protocol
- 4B/5B coding
- 主な仕様
  通信速度　125Mbps
  ノード間＜40km
  リング長　＜100km



→ Frame Format

→ Token Passing, Timed Token Protocol

Other Layer-1,2 Protocols

LAN

- Token Ring
- ATM –LAN
- 無線 LAN (IEEE802.11)
- Fiber Channel (ANSI X3T9.3)
- HIPPI (High Performance Parallel Interface) (ANSI X3T9.3)

WAN

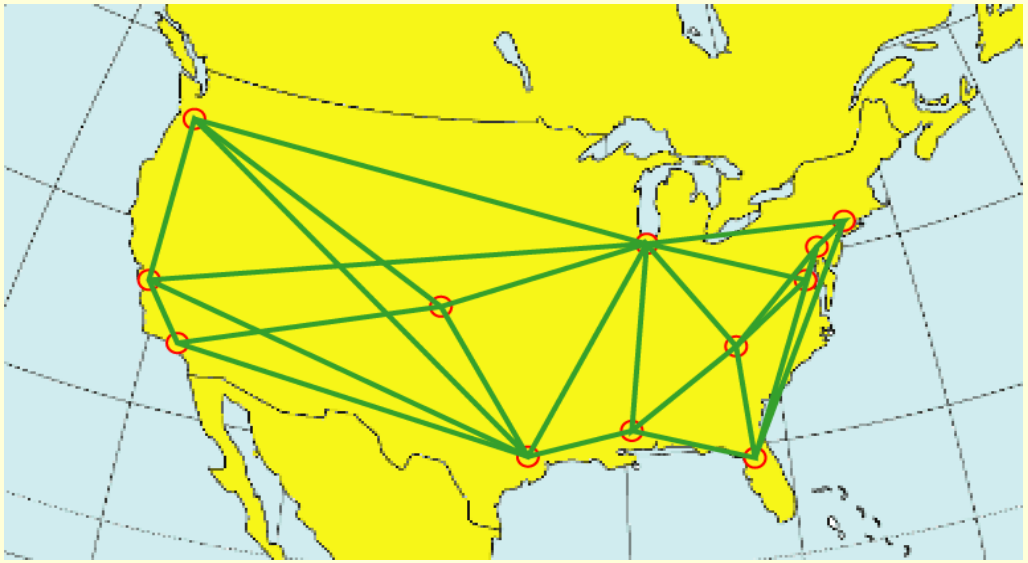
- ATM (Asynchronous Transfer Mode)
- Frame Relay (ANSI X3T11)
- ISDN (Integrated Services Digital Network)
- xDSL (Digital Subscriber Line, HDSL, SDSL, ADSL, VDSL)
- FTTH (Fibre To The Home)

Internet

全世界の LAN を接続した巨大なコンピュータネットワーク。

1969年に米国防総省の高等研究計画局(ARPA)が始めた分散型コンピュータネットワーク研究プロジェクト ARPANET がその起源であるといわれる。

→ ソ連からのミサイル攻撃に対する防衛戦略



図はイメージ図です。実際の計画とは関係ありません。

- 中央の制御ポイントを持たない自律型ネットワーク
- 送信データが障害のあるネットワークを自動的に避けて宛先に到達する

Internet Protocol (IP)

→ RFC 791 (1981年)

インターネットのネットワーク層の中核的プロトコル
ルータによって接続された2点間のコネクションレスなデータグラムの送受信を行う。

- 通信データ
  - IP Datagram と呼ばれるデータの塊の単位で通信を行う。
  - IP Datagram は送信元アドレス(IP Address)、送信先アドレス、オプション情報、およびデータから成る。
- データグラムの配送（経路選択）
  - ルータは IP Datagram の送信先アドレスを元に送信経路を選択する。
  - 経路選択は個々の IP Datagram に対して行う。
- その他特徴
  - Datagram の到着順序は保証されない。
  - ヘッダのエラー検出機能あり。データのエラー検出機能はない。
  - 喪失した Datagram を再送する機能はない。喪失通知機能はある(ICMP)。
  - フラグメント化機能あり。
  - セキュリティに関する機能はない。

IP Address

- 4 つの 8 bit 値を 10 進数で表現したものとして一般に表現される。
  例： 192.168.1.5
- ネットワーク部とホスト部に分かれる。
  例： ネットワーク部: 192.168.1　ホスト部: 5
- Class の割り振り (但しブロードキャストアドレスを除く)
  - Class A　0.0.0.0 – 127.255.255.255　255.0.0.0 (8)
  - Class B　128.0.0.0 – 171.255.255.255　255.255.0.0 (16)
  - Class C　172.0.0.0 – 204.255.255.255　255.255.255.0 (24)
  - Class D　205.0.0.0 – 255.255.255.255　マルチキャストアドレス
  - Class E　240.0.0.0 – 255.255.255.255　実験用
- Private Address → IP Allocation (RFC1918)
  - 10.0.0.0/8
  - 172.16/12
  - 192.168.0.0/16

IP

Subnet and Subnet Mask

アドレス空間を効率よく利用するためクラスのホスト部とさらにサブネット部とホスト部に分ける。

Host IP	192. 168. 1. 5	=	11000000	10101000	00000001	00000101
Netmask	255. 255. 255. 0	=	11111111	11111111	11111111	00000101
Network	192. 168. 1. 0	=	11111111	11111111	11111111	00000000

サブネットマスク以下の 172.168.1.0-172.168.1.255 がブロードキャストドメイン内のホストとなる。

→ CIDR

IP

特別な意味を持つ IP Address

ネットワーク部	ホスト部	
0	0	このネットワークぶのこのホスト。自分の IP Address を調べるような初期化手続きで使われる。
0	ホスト	このネットワーク中の指定されたホスト。フル IP Address を調べる際などに用いられる。
全て1	全て1	制限されたブロードキャスト。このネットワークに属するホストにブロードキャストする。
ネットワーク	全て1	特定のネットワークへのブロードキャスト
127	任意	ループバックアドレス

IP

IP Address はどのようにして割り振られるか？

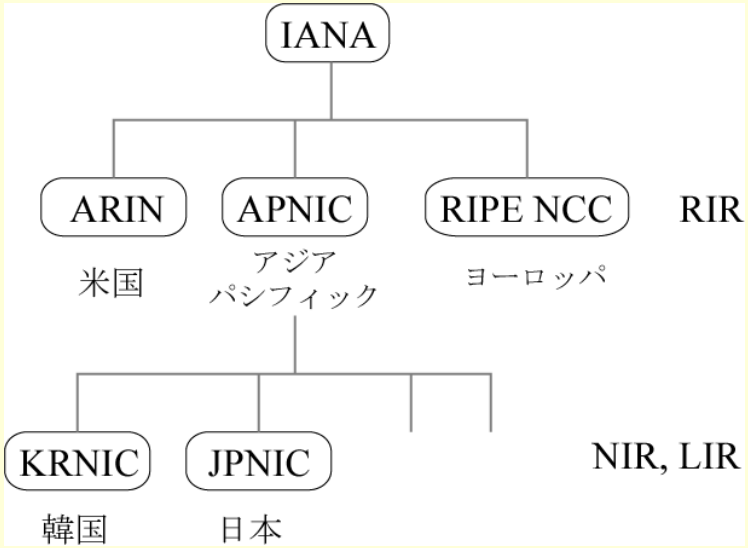
IP Address は、IANA (Internet Assigned Numbers Authority, 現在 ICANN Internet Corporation for Assigned Names and Numbers に機能を移管)を頂点とする階層構造による割り振り・割り当てが行われている。

IANA からは各RIR（地域インターネットレジストリ）へアドレスブロックが割り振られ、RIRは、NIR（国別）またはLIR（ローカル）にアドレスブロックを割り振る。

日本の IP Address は、アジア太平洋地域におけるRIRであるAPNICに属する JPNIC (Japan Network Information Center) が管理を行ってる。

DNS の階層構造とは基本的に無関係

→ Address Space



Classless Inter-Domain Routing (CIDR)

→ RFC 1518, 1519, 1817

IP Address の枯渇に対応するため、従来のクラス分けにとらわれない柔軟なアドレス方式が導入された。

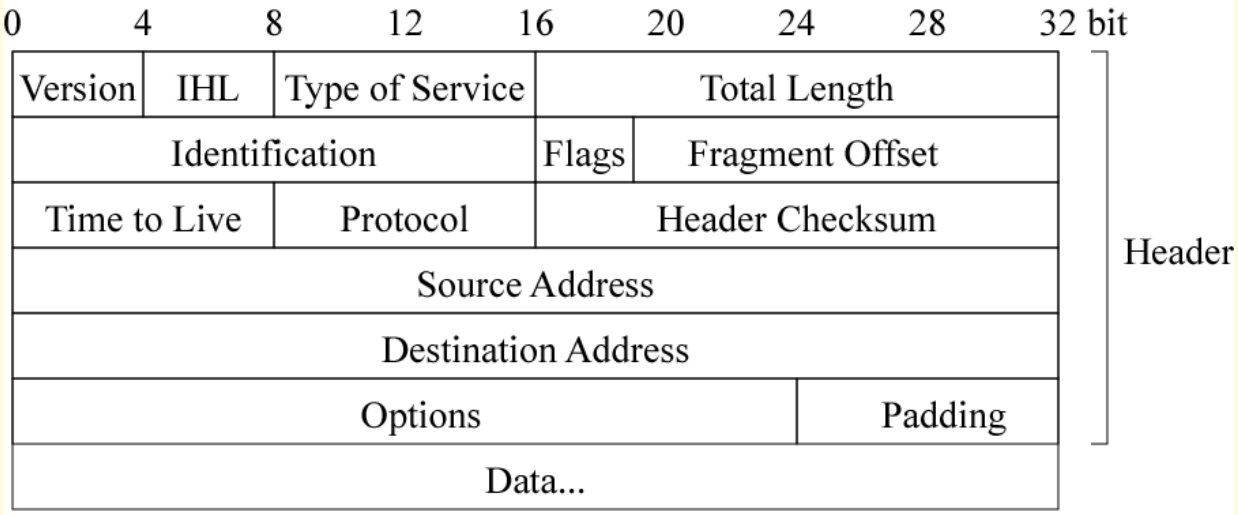
- ネットワーク部とホスト部をネットワークの規模に合わせて適当なビット境界で区切る。
- 経路情報と一緒にネットワーク部を示す情報を送り、他のゲートウェイにネットワーク部のビットを知らせる。

表現の例： 192.168.1.5/24 … 先頭 24 bit (192.168.1) がネットワーク部

→ IP Allocation (RFC1518)

IP

IP Datagram



Version:	IP datagram version (=4)
IHL:	IP header の長さ (32bit 単位)
Type of Service:	サービスの質 (delay, throughput, reliability)
Total Length:	ヘッダとデータの長さ (8bit単位)
Flags:	フラグメントに関するフラグ
Fragment Offset:	元データ中のフラグメントの位置
Time to Live (TTL):	生存時間
Protocol:	ICMP, TCP, UDP, etc.
Header Checksum:	ヘッダのチェックサム
Source Address:	発信元 IP アドレス
Destination Address:	送信先 IP アドレス
Options:	経路指定など(可変長)
Padding:	パディング(0で埋める)

→ RFC 791

TTL は 1 秒もしくはルーターを1段通る度に 1 減らされ、0 になった時点で Datagram が破棄される。

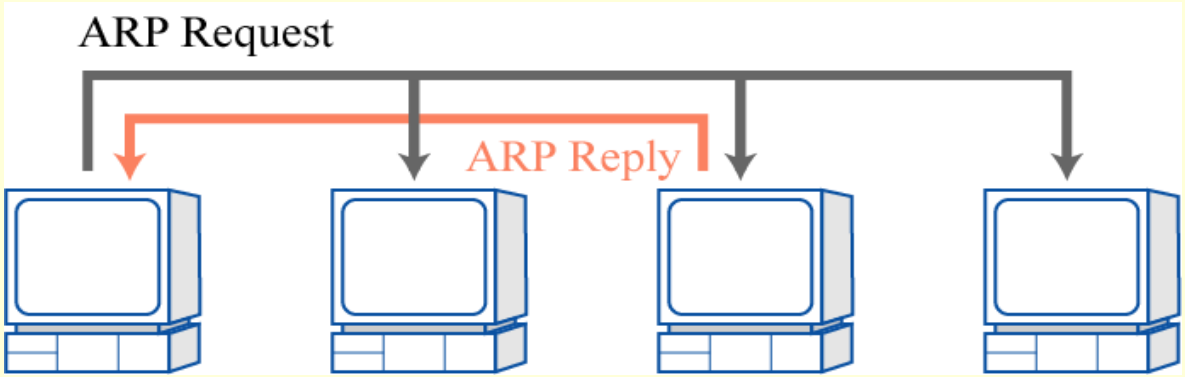
Address Resolution Protocol (ARP)

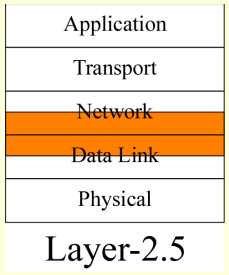
→ RFC 826

IP Address を MAC Address に変換するプロトコル。

相手の IP Address を指定した ARP 要求メッセージをネットワーク(Broadcast Domain)内の全てのホストにブロードキャストする。

指定されたアドレスに対応するホストは、物理アドレス (MAC Address) と IP Address を組にした応答メッセージを送り返す。





RARP Reversed Address Resolution Protocol (RARP)

→ RFC 903

ARP とは逆に、物理アドレス (MAC Address)がわかっているシステム(自システムを含む)の IP Address を調べる為に使われるプロトコル。

IP Address を知りたいシステムの物理アドレスをセットした RARP 要求メッセージをブロードキャストする。

ネットワーク上に RARPサーバーがあれば、要求元システムに物理アドレスと IP Address の組を送り返す。

ディスクレスシステムの起動時などに用いられる。

Dynamic Host Configuration Protocol (DHCP)

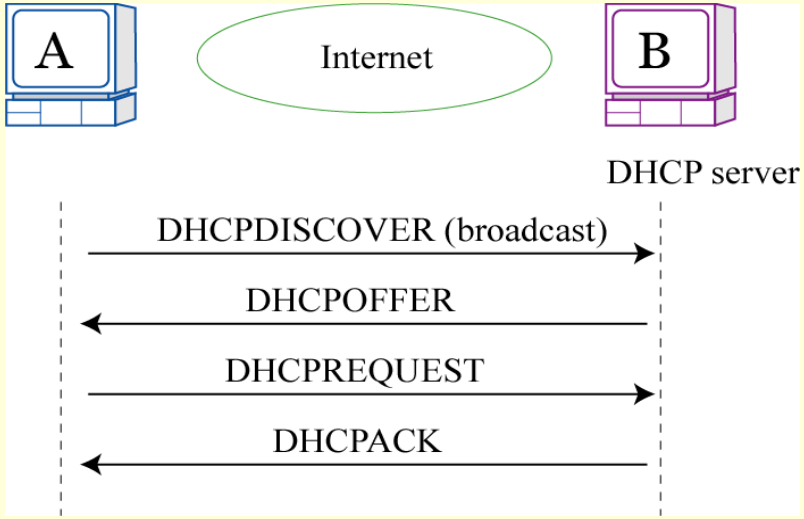
→ RFC 2131

インターネットに一時的に接続するコンピュータに、IPアドレスなど必要な情報を自動的に割り当てるプロトコル。

他に、ゲートウェイやDNSサーバのIPアドレスや、サブネットマスクなどの情報を提供。

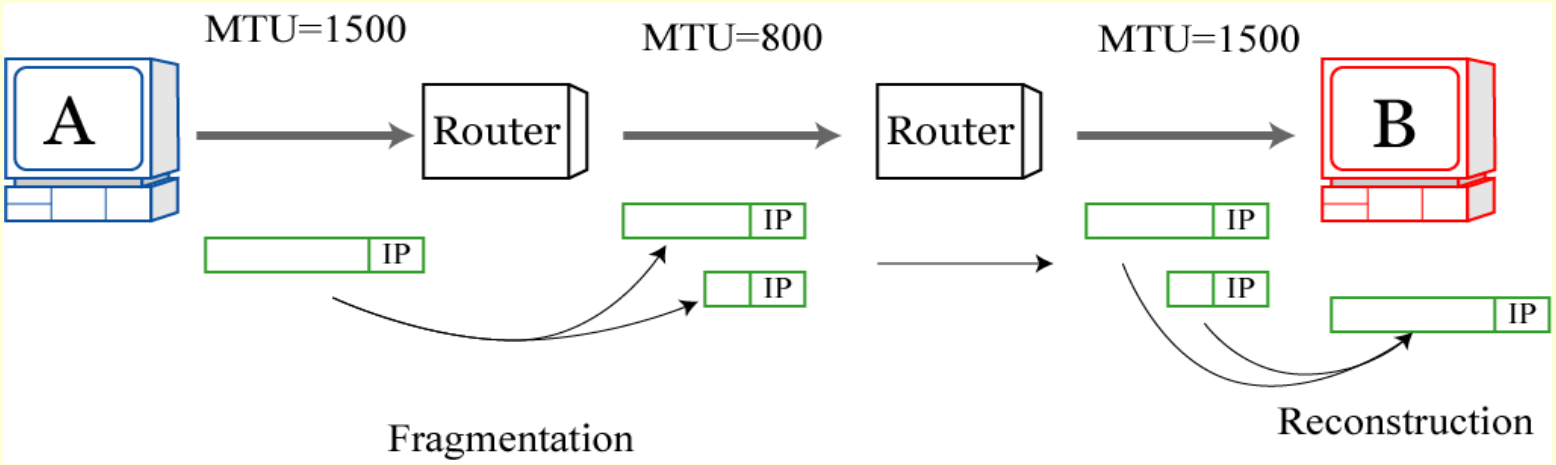
IPアドレスの有効利用。

貸し出す IP アドレスには
有効期限を付けることができる。



IP

Fragmentation of IP Datagrams



- 通信路の許容する最大データサイズ(MTU)よりも大きい IP Datagram が送られて来た場合、ルータは Datagram を分割(Fragmentation)して送信する。
- 再構成のための情報は IP Header の Fragment Offset に格納される。
- 分割されたデータは最終到着点にて再構成される。

(分割後の Datagram の順序が変わったり、転送経路が異なったりすることがある)

IP Datagram の経路選択

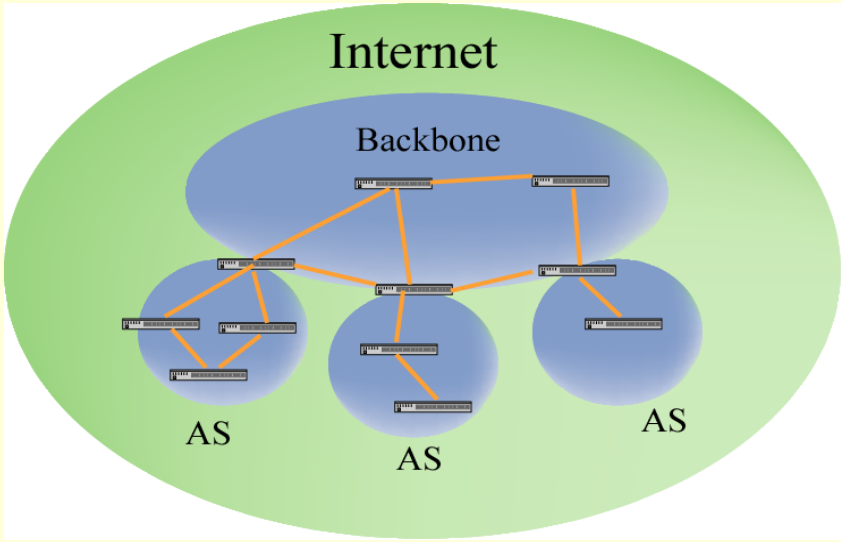
IP Datagram の経路は中継点であるルータが決める。

→ 受け取った Datagram を次に誰に渡すかを決める。

＝ 全世界規模のバケツリレー方式

動的経路制御プロトコル(自律システム間)

- BGP4 (Border Gateway Protocol ver 4, RFC 1771)
  TCP による peer to peer 通信
  Incremental な情報交換
  Path Vector 方式
  CIDR のサポート



自律システム(Autonomous System)

基本的に ISP(Internet Service Provider)
が構成要素。他に大学や企業など

IP Datagram の経路選択

動的経路制御プロトコル(自律システム内)

- RIP2 (Routing Information Protocol ver 2, RFC 2453)

  比較的小規模の自律システム内ルーティングに向いている(Distance Vector 型)。
  隣接ルータ間で互いに経路表を交換し、目的ネットワークへの“距離”に
  基づいて最短経路を決定する。
  トポロジ変化による再構成のコンバージェンス時間が長い。

- OSPF (Open Shortest Path First)

  大規模の自律システム内ルーティングに向いている(Link State 型) 。
  自律システム内をエリアという領域に分ける。
  接続しているネットワーク情報をルータ間で互いに交換。
  自分をルートとする Shortest Path Tree というトポロジーデータベースを作成して最短経路を決定する。

ICMP

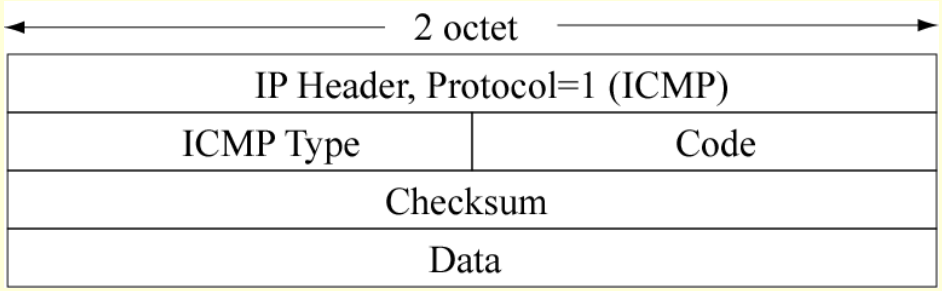
Internet Control Message Protocol (ICMP)

→ RFC 792 (1981)

IP を補完するプロトコル。IP 処理中に発生したエラーを通知したり、
要求に応じてアドレスやタイムスタンプ情報などを応答する。

主なICMP メッセージ (type)

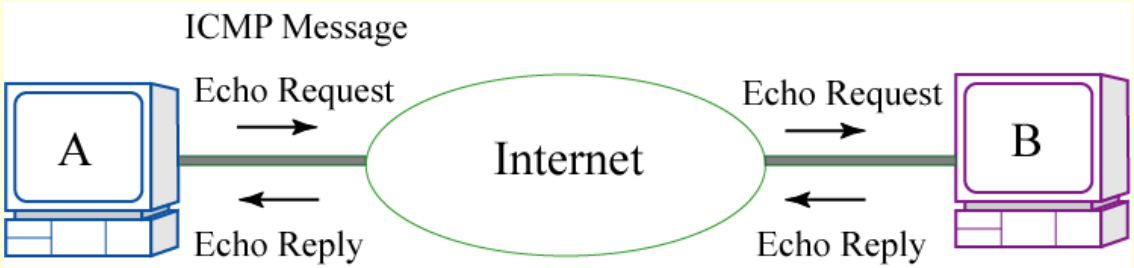
- Echo request (8)
  返答要求。ping や traceroute に使われる。
- Echo reply (0)
  Echo request への返答。
- Destination Unreachable (3)
  宛先まで IP Datagram を送れなかった場合に送信元に送る。



ICMP

ping

ICMP の Echo Request メッセージを使って、標的ホストまでの通信状態確認する。



A→B　ICMP Echo Request Message を送る。

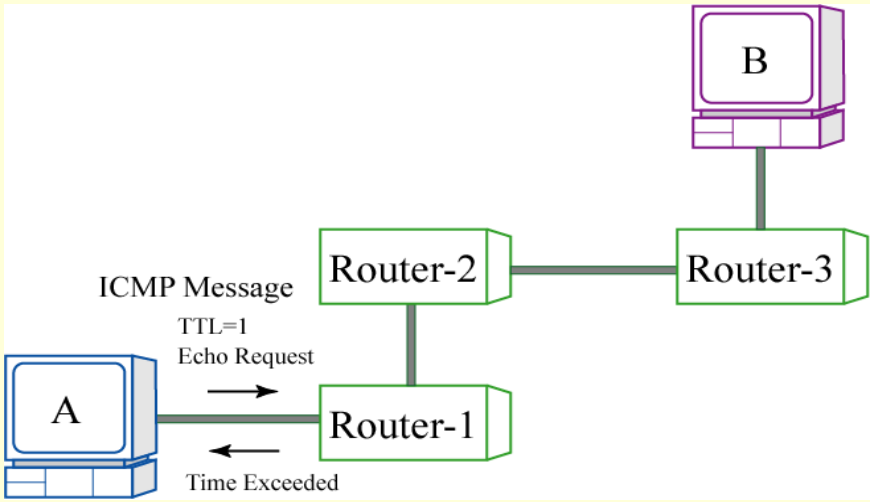
B→A　ICMP Echo Reply Message を返す。

ICMP

traceroute

ICMP Message を利用して標的ホストまでの経路を調べる。
TTL を 1 から順に大きくして、返ってくる Message の送信先を調べる。

1. TTLを1にして、BへICMP Echo Request (or UDP Datagram)を送る。
   → Router-1 から ICMP Time Exceeded が返る。
2. TTLを2にして送る。
   → Router-2 から ICMP Time Exceeded が返る。
3. TTLを3にして送る。
   → Router-3 から Time Exceeded が返る。
4. TTLを4にして送る。
   → B から Echo Reply (or ICMP Port Unreachable) が返る。



User Datagram Protocol (UDP)

→ RFC 768

“相手にデータを投げつける”通信

- コネクションレス(Connectionless)
- データの到着順序は保証されない
- 誤り検出機能あり。再送機能なし。データの到着は保証されない。
- フロー制御機能なし

→ リアルタイム性のある通信、高速性重視の通信など。

但し、

・データの到着および正しさが保証されない。

必要があれば UDP を使うアプリケーション層で対応。

UDPを使う通信

UDP を使う通信の例:

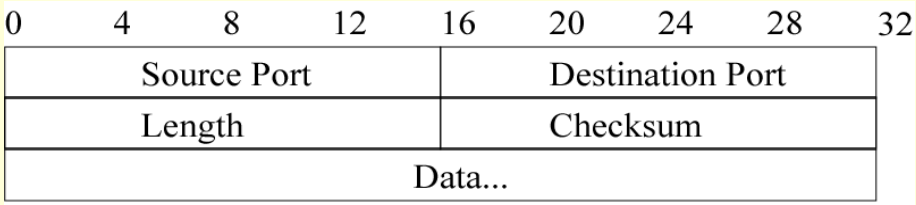
nfs (rpc)	2049	Disk 共有
tftp	69	簡単なファイル転送プロトコル
ntp	123	ネットワーク時刻調整
domain	53	DNS (query)
bootp	67,68	Diskless システムのプブートなど
dhcp	546,547	IP Address の動的割り当てなど

→ Well-known ports

UDP

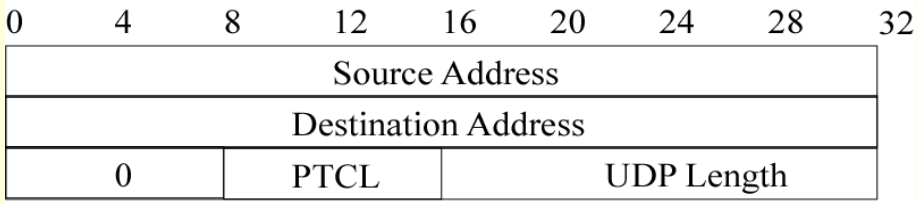
UDP Datagram

UDP Datagram のフォーマット



Source Port:	送信元ポート
Destination Port:	送信先モード
Length:	ヘッダとデータを含む長さ (octet 単位)
Checksum:	擬似ヘッダ、UDPヘッダ、データを含むチェックサム

擬似ヘッダのフォーマット



Source Address:	送信元 IP Address
Destination Address:	送信先 IP Address
PTCL:	プロトコル番号 (UDP=17)
UDP Length:	UDP ヘッダとデータを含む長さ (octet 単位)

→ RFC 768

Transmission Control Protocol (TCP)

→ RFC 793 (1981)

“互いにデータのやりとりを確認しあう”通信

- コネクション型 (Connection-Oriented)
  3-Way Handshake によるコネクション確立
- ストリーム型通信
- 通信データの順序制御
- 誤り検出機能・再送機能
- 相手へのデータの到着確認
- フロー制御
- ポートによるコネクション多重化

→ 様々な伝送路上で、効率が良く信頼性のある通信を確立する。
但し、

・プロトコルが複雑

・UDP に比べ通信が遅い (リアルタイム性が損なわれる可能性)

UDP

TCPを使う通信

→ RFC 768

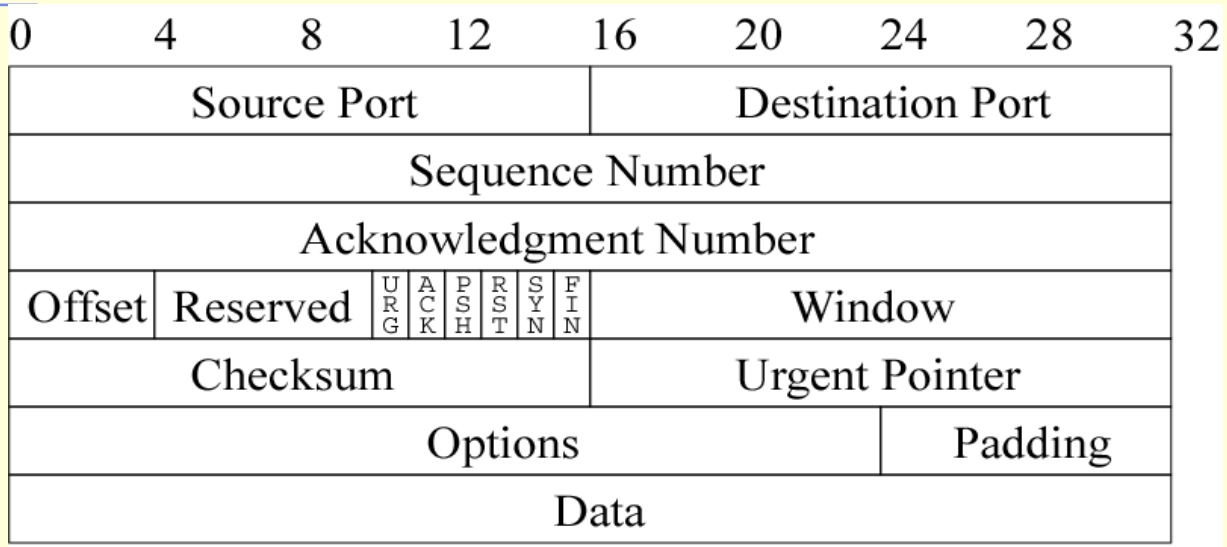
UDP を使う通信の例:

telnet	23	telnet によるログイン
ftp	20,21	ftp によるデータ転送
ssh	22	ssh (secure shell) によるログイン
smtp	25	メイル転送
http	80	WWW
domain	53	DNS (ゾーン転送)
printer	515	lpd による印刷

→ Well-known ports

TCP

TCP Segment



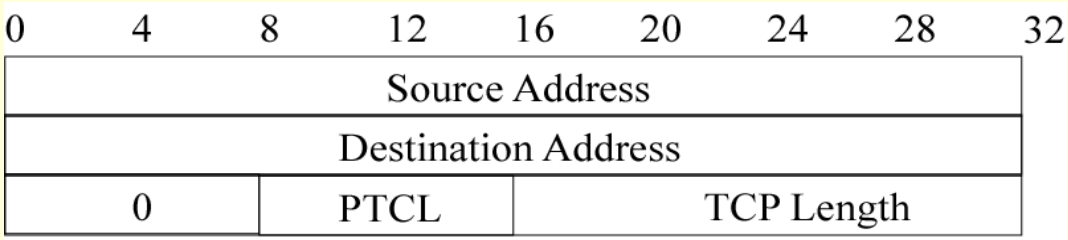
Source Port:	送信元ポート
Destination Port:	送信先モード
Sequence Number:	データ順序番号
Acknowledgment Number:	次に受け取ることを期待している順序番号
Offset:	TCP ヘッダの長さ (32bit単位)
Flags:	接続要求(SYN)、Acknowledge (ACK)、リセット(RST)、送信終了(RST)、緊急情報(URG)、Push (PSH)
Window:	受信可能なデータ量(octet単位)
Checksum:	擬似ヘッダ、TCPヘッダ、データを含むチェックサム
Urgent Pointer:	緊急データポインタ
Options:	オプションリスト(可変長)
Padding:	パディング(0)
Data:	送信データ

→ RFC 793

TCP

TCP Pseudo Header

TCP 擬似ヘッダフォーマット



Source Address:	送信元 IP Address
Destination Address:	送信先 IP Address
PTCL:	プロトコル番号 (TCP=6)
TCP Length:	TCP ヘッダとデータを含む長さ (octet 単位)

→ RFC 793

TCP による通信手順

1. 接続の開始
   - 3-Way Handshake
     - port number and connection
     - well-known ports

2. データの転送
   - Sliding Window
   - Retransmission
     - Checksum

→ Ref. 輻輳制御の詳細

3. 接続の終了
   - FIN

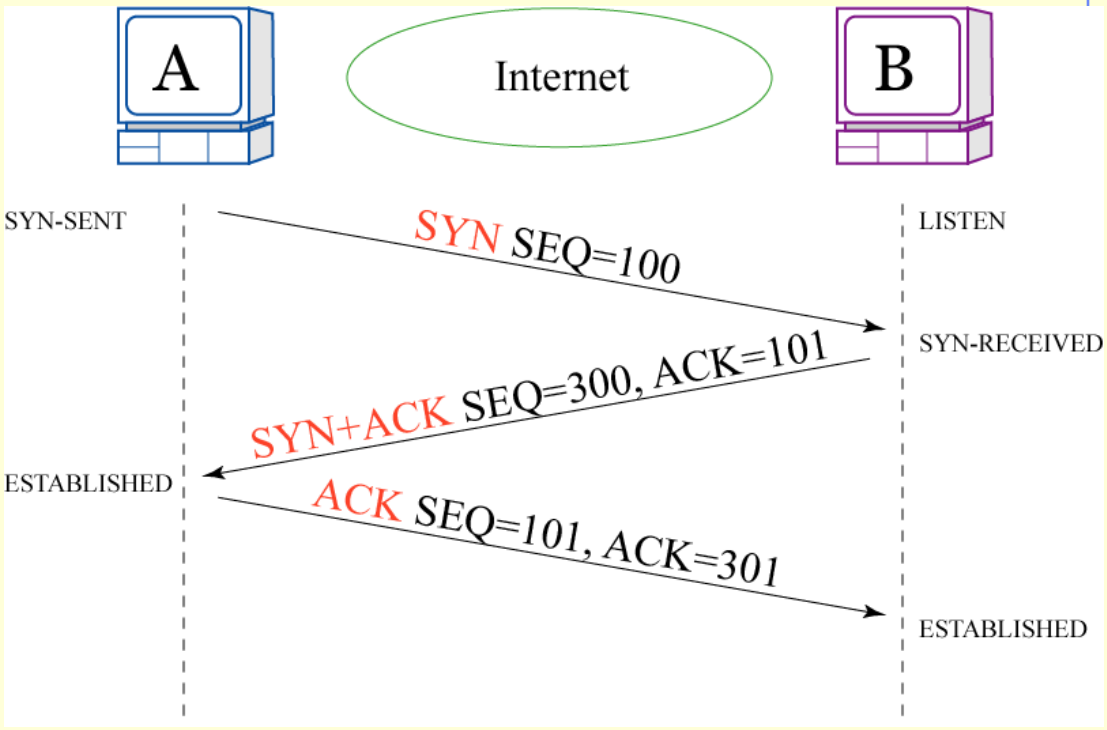
TCP

3-Way Handshake

→ RFC 793 (1981)

TCP の接続開始手順

1. A→B 接続要求
   SYN
2. B→A 応答
   SYN+ACK
3. A→B 応答
   ACK



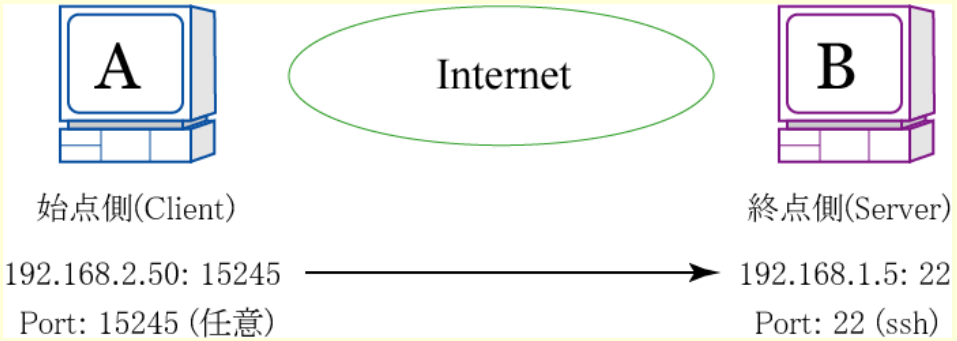
目的:

- 通信路の確認
- 互いの順序番号の交換

TCP/UDP

Port numbers and Connection

- 接続要求側(クライアント)は任意の始点側ポート番号(通常1024-65535)をアサインし、相手(サーバ)の終点側ポート番号を指定して接続要求を送る。
- クライアントが受けられるサービス(ssh, http, ftp, etc)はサーバのどのポートに接続するかによって決まる。 → port numbers
- サーバ側は提供するサービスに対応するポートの接続要求を待つ(LISTEN)デーモンを用意する。
- コネクションは以下の4つの値の組を用いて識別する。
  1. 自分の IP アドレス
  2. 自分の ポート番号
  3. 相手の IP アドレス
  4. 相手のポート番号



> netstat -n

TCP/UDP

Well-known ports

- サーバー側の標準的なサービスについては事前にポート番号が割り当てられている (1-1023)。
  → well-known ports (IANA)
  例： ssh(22/tcp), telnet (23/tcp), smtp (25/tcp),
  www(80/tcp), ntp(123/udp), domain (53/udp)

  TCP と UDP のポート番号は基本的に独立。

- アプリケーションによっては、独自のポート番号を使用するものがある。(≧1024)
  → assigned ports (IANA)

TCP

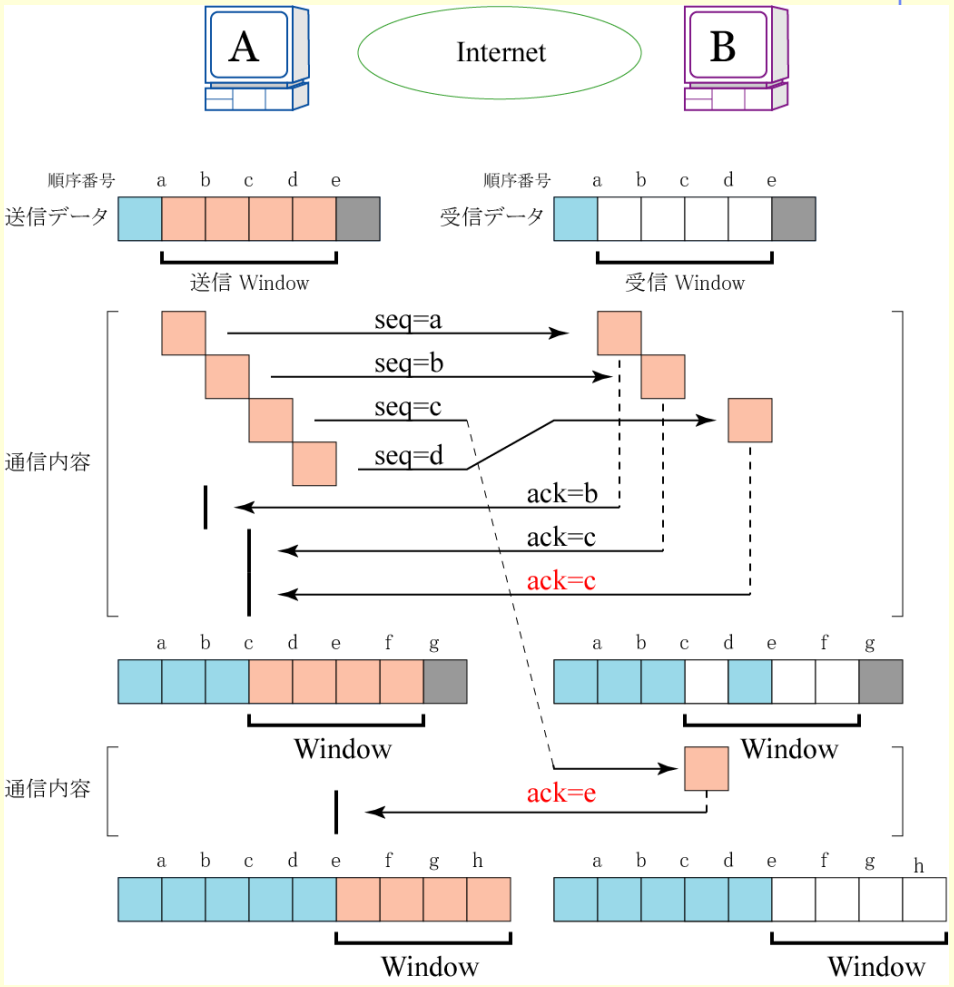
Sliding Window

必ずAcknowledge が返って来てから次のセグメントを送るという方式では、転送データ量が多い場合に高速通信ができない。

↓

- Sliding Window 方式により、複数のセグメントの同時送信を行い、またフロー制御も行う。
- ネットワークの品質に合わせて、適切な Window サイズを動的に決定する。

→ Adjustment of the window size



例： 到着順序がずれた場合

注: 実際には、Window がスライドするのは受け取ったデータが処理された後である。

TCP

Adjustment of the Window Size (1/2)

データ受信側は送信側に受信バッファの使用可能な大きさを TCPセグメントの Window 値として通知する。
→ 送信側はこれに応じて送信 Window の大きさを調整する。

しかし単純にこの方式を実行すると、徐々に Window サイズが低下してしまう (Silly Window Syndrome: SWS)。

これを避けるために以下の操作を行う (RFC 813, 1122)。

- 受信側
  - 連続して受信したセグメントに対して1つの ACK を返す(delayed ACK)。
  - Window がある値よりも小さくなった時実際よりも小さい Window サイズを送る。
- 送信側
  - 送信可能な Window が小さくなったとき、一定の大きさになるまで送信を保留する。(nagle のアルゴリズム: RFC 896)

TCP

Adjustment of the Window Size (2/2)

セグメントの再送が発生した場合には、これを輻輳の兆候ととらえ、IP Datagram が消失するたびに以下の操作を行う。
(半減輻輳回避: Muticable Decrease Congestion Avoidance)。

- 送信 Window サイズを半分にする。

再送が発生しなくなれば以下の操作を行う。
(スロースタート回復: Slow Start Additive Recovery)

- 相手から ACK を受けるごとに 送信 Window サイズを 1 ずつ増やす。
- 送信 Window が本来の大きさの半分になった時点で 送信 Window の 増加量を抑え、Window 内の全てのデータに対する ACK を受けるごとに送信 Window サイズを 1 増やす。

→ RFC 1122 (参考 RFC 2001)

TCP

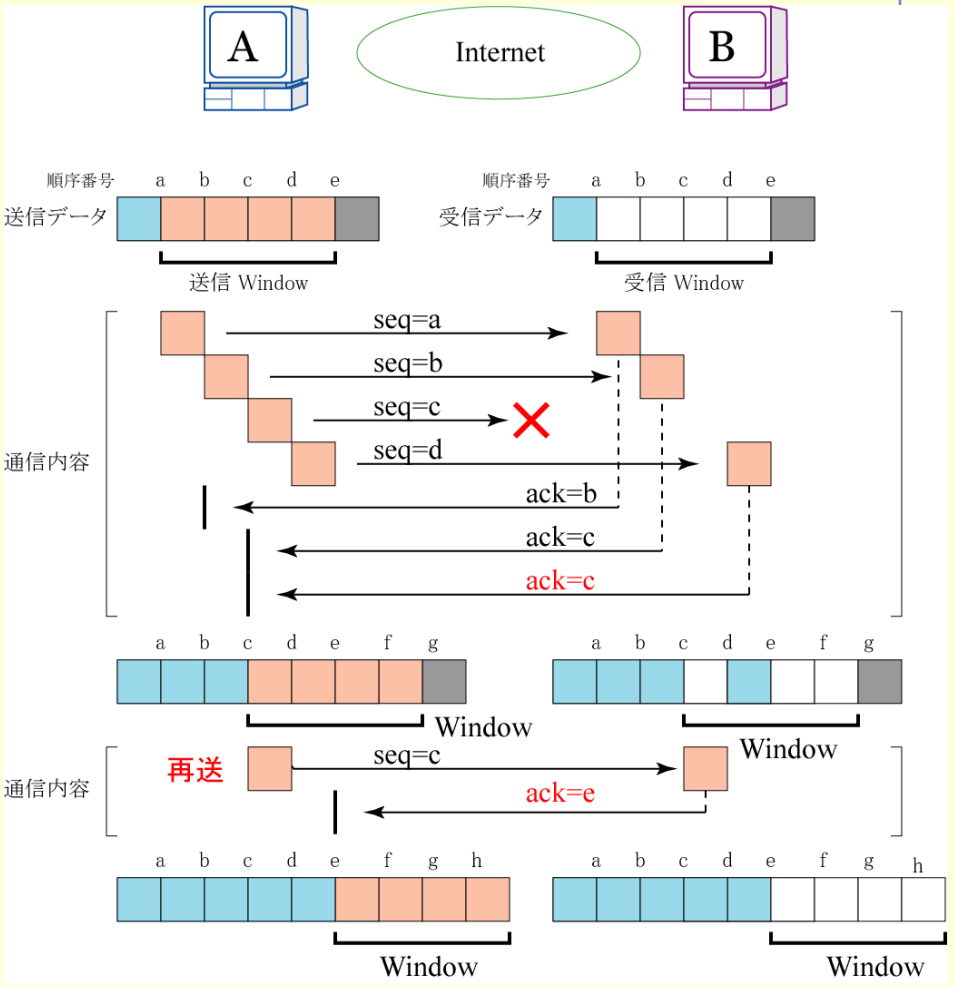
Retransmission of a lost segment

セグメントがデータ受信側に届かなかった場合、または Checksum にエラーが生じた場合には、セグメントが失われる。

- 受信側は、受け取っていないデータの最小順序番号を送信側に返す(ACK)。
- 送信側は、ある時間の間にデータが到着しなかったと判断した場合(timeout)には、データの再送を行う(retransmission)。
- timeout までの時間は、ネットワークの品質により、動的に決定される。

→ Retransmission Timeout

注: SYN と FIN は暗黙の順序番号を持ち、再送の対象となる。



例: セグメント再送の様子

TCP

Retransmission Timeout

再送タイムアウト時間を以下のように計算する。(Jacobson のアルゴリズム RFC 1122)

- RTT = データ送信から ACK 到着までにかかった時間 (Round Trip Time)
- SRTT' = α× SRTT + (1-α)×RTT (Smoothed RTT)
- diff = RTT-SRTT
- VAR' = (1-δ)×VAR + δ×| diff | (diff の分散)
- RTO = SRTT + 2×VAR (Retransmission Timeout)

  α= 0.8~0.9, δ~1/8

  initial values: RTT = 0 sec, RTO = 3 sec.

RTT を平滑化し、その予想値より 2×分散 だけ大きい値をタイムアウト値とする。ただし下限値(数分の1秒)と上限値(240秒)を設定する。

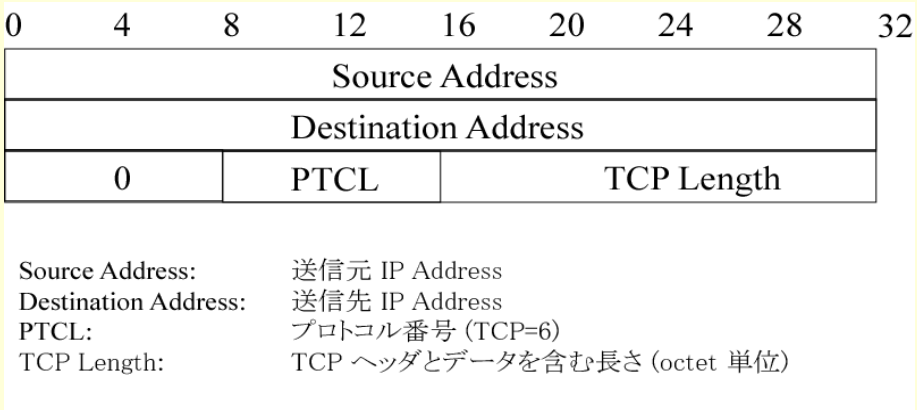
但し、実際に再送が発生した場合には RTT 値を捨て、以下の式で次の RTO を計算する (指数的バックオフ: Karn のアルゴリズム RFC 1122)

- 新 RTO = 2 × 旧 RTO

TCP

Check Sum

TCP ヘッダの前に擬似ヘッダを付けて Check Sum を計算する
→ 配送ミスを検出するため



TCP 擬似ヘッダ

Check Sum の計算：
擬似ヘッダ、TCP ヘッダ、データに対し、16ビット毎に1の補数をとり、その和の1の補数を Check Sum とする。

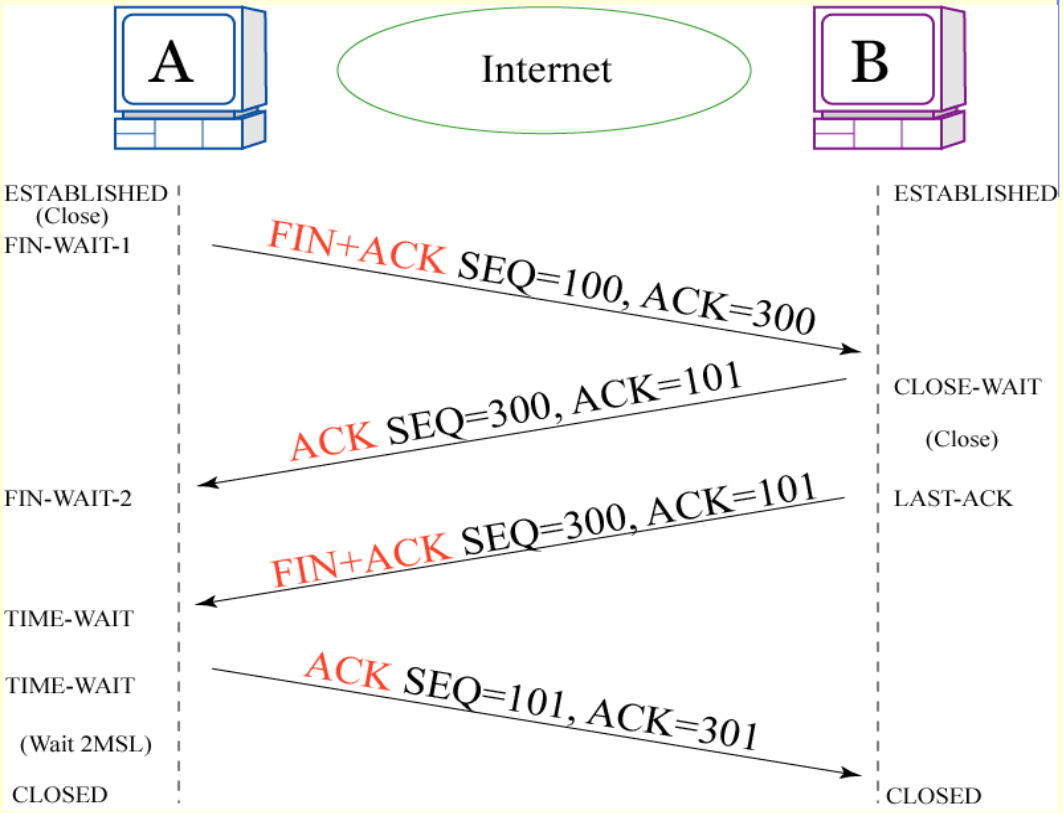
TCP

FIN

TCP 接続の切断手順

双方の切断(FIN)要求
が揃ってから切断する。

1. A→B 切断要求(FIN)
2. B→A 応答
   (この時点ではまだ B→A の
   データ送信は可能)
3. B→A 切断要求(FIN)
4. A→B 応答



B は直後に CLOSE。
A は MSL の2倍の時間待ってから CLOSEする。

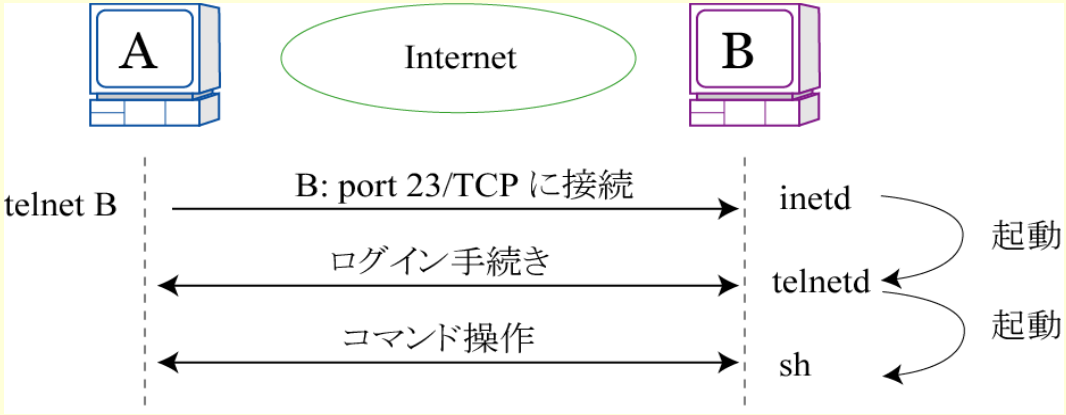
注: MSL (Maximum Segment Lifetime) は RFC793 では 2分と定義されている。
実際には 30 秒の実装も多い。

telnet

→ RFC 854

TCP/IPを用い、ネットワークにつながれたコンピュータを遠隔操作するための標準方式、またプロトコル。

デフォルトではポート 23 (telnet) に接続するが、他のポートにつなぐことも可能。

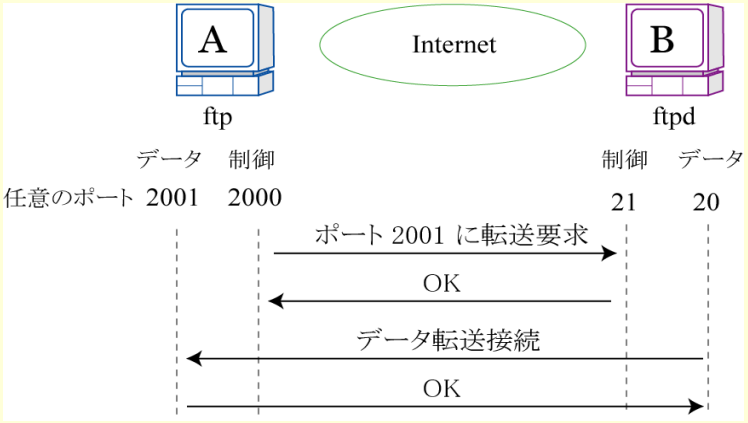


File Transfer Protocol (FTP)

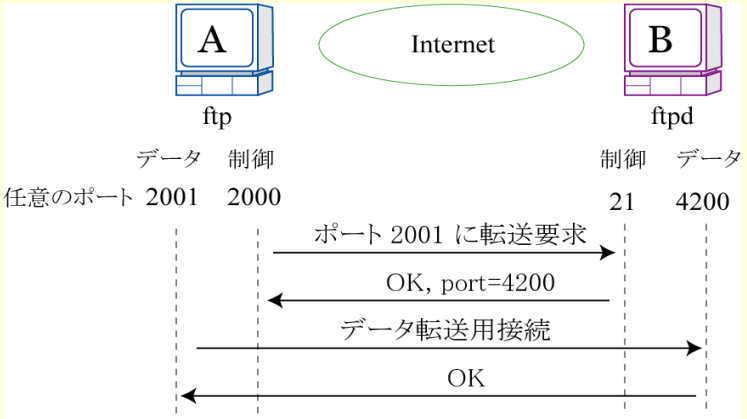
→ RFC 959

TCP/IPによりファイルを転送する時にに使われるプロトコル。
制御用とデータ用の2つの接続を使用する。

通常の FTP 接続



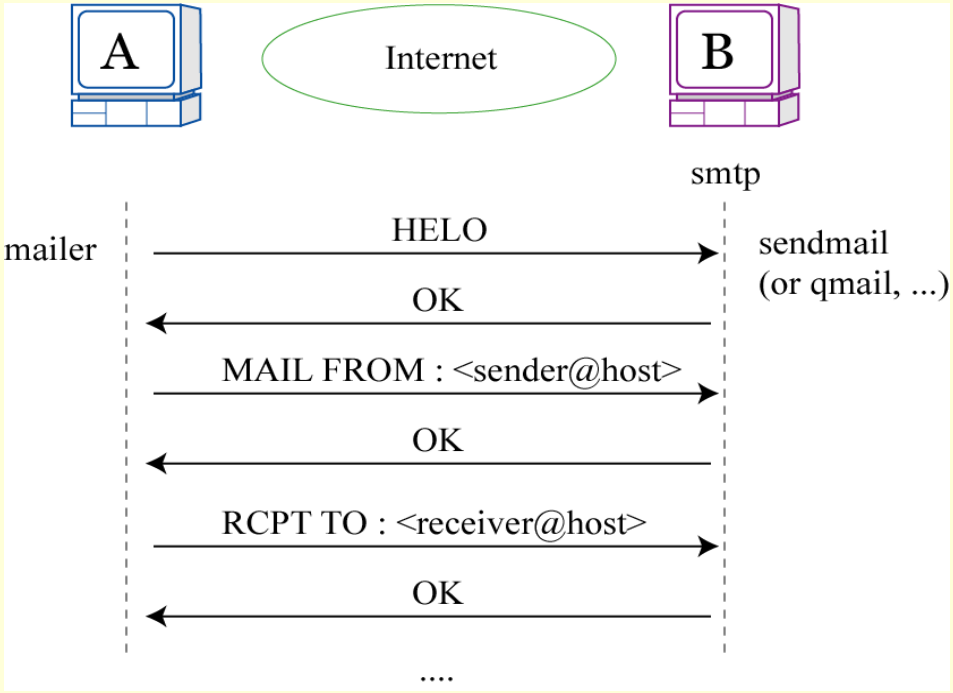
Passive FTP 接続



Simple Mail Transfer Protocol (SMTP)

→ RFC 821, 1869

TCP/IP で電子メールを送信するためのプロトコル。



Application
Transport
Network
Data Link
Physical

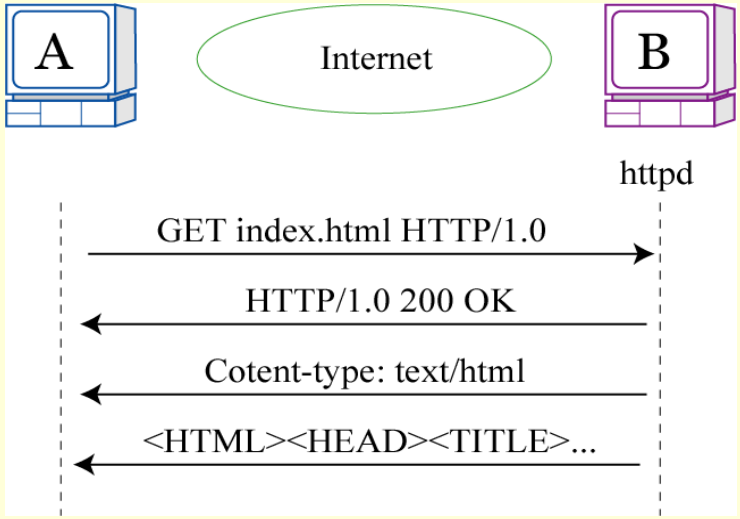
Layer-5-7

HyperText Transfer Protocaol (HTTP)

→ RFC 1945, 2616

Webサーバとクライアント(ブラウザ)がデータを送受信するのに使うプロトコル。

HTML文書や、画像、音声、動画などのファイルを、表現形式の情報を含めてやり取りする。



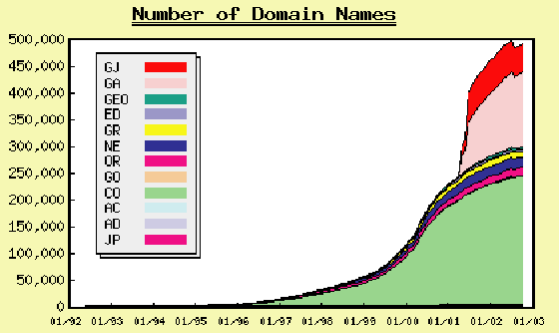
Domain Name System (DNS)

→ RFC 1034, 1035

ホスト名とIPアドレスを対応させるシステム。
全世界のDNSサーバが協調して動作する分散型データベース。

ホスト名→IPアドレス　　正引き
IPアドレス→ホスト名　　逆引き

日本語ドメイン名や個人名などの
新しい運用の開発も進められている。



ドメイン名の種類: http://www.nic.ad.jp/ja/dom/basics.html

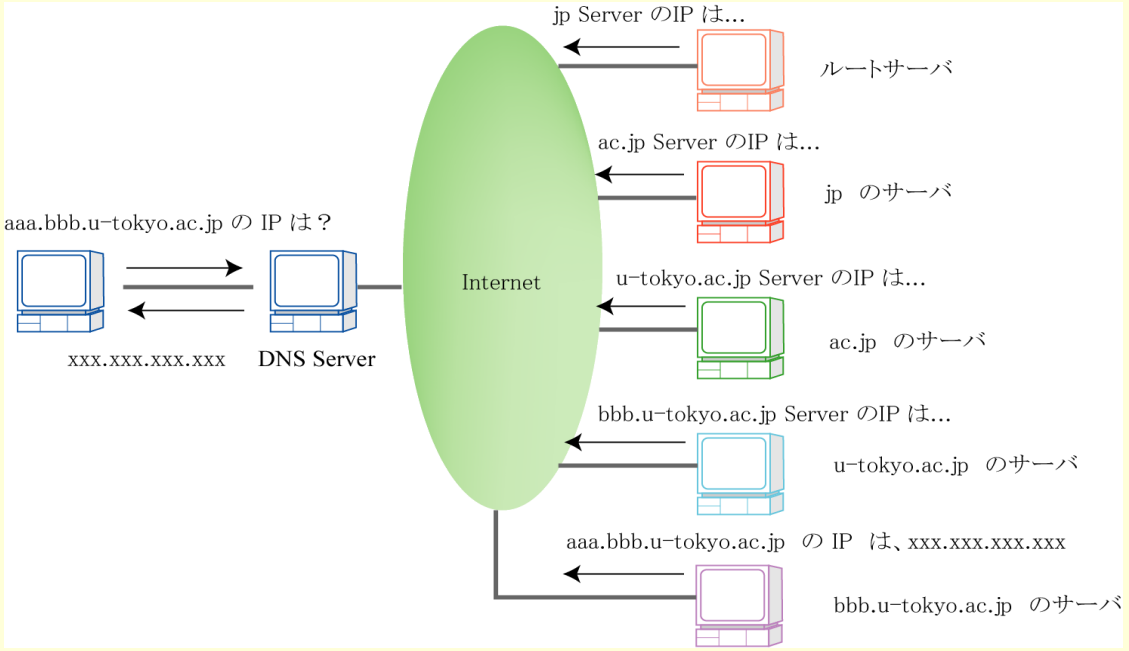
Domain Name System (DNS)

IPアドレス検索の原理 → たらいまわし的解決

1） クライアントが DNS サーバに名前の解決を要求 (query)
2） DNS サーバは、自分の管轄内のホスト名、キャッシュにあって既に知っているホスト名の場合はそのまま返答する。
3） 知らない場合は、ルートサーバから順に聞いていく。

ルートサーバの IP はサーバ上のルートキャッシュと呼ばれるファイルに登録されている。
現在13台

逆引きの場合は
xxx.xxx.xxx.xxx.in-addr.arpa
を検索する。

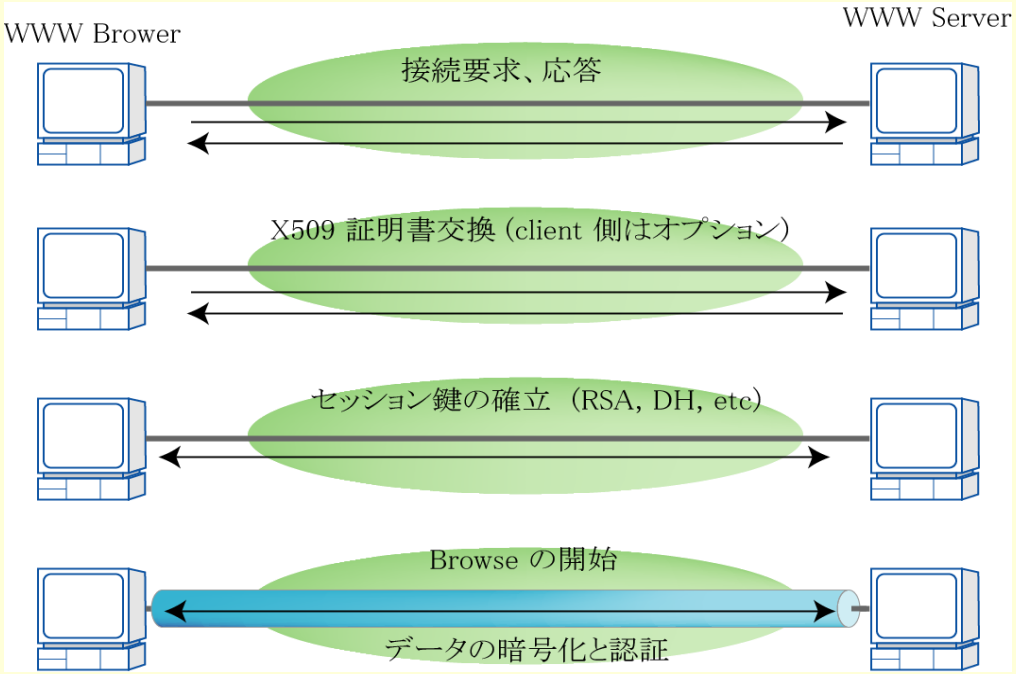


Secure Socket Layer (SSL)

Netscape 社が開発した、インターネット上で情報を暗号化して送受信するプロトコル。

主に WEB Server と WEB Browser 間の暗号通信に用いられている。

SSL 3.0をもとにTLS 1.0が標準化された（RFC 2246）。



IPv6 (IP version 6)

現在の IPv4 のアドレス枯渇問題に対処するため、1992 年に検討を開始した次世代 IP 技術。

特徴:

- IP アドレスを 32bit から 128 ビットへ拡張
  → 全ての家電、モバイル製品を IP で接続
- 階層的なアドレス体系
  → 効率的な経路情報処理
- 簡素化された固定長ヘッダ、フラグメント廃止
  → ルーティングの効率化
- IPsec 標準実装によるセキュリティの強化
- NDP (Neighbor Discovery Protocol) によるアドレス解決
- IP アドレスの自動生成、自動付け替え機能 → Plug and Play
- マルチキャストのサポート、ブロードキャストの廃止
- QoS (Quality of Service) 機能

IPv6 の時代はもう始まっている

- 多くの OS が IPv6 対応
  HPUX, Solaris, Linux, FreeBSD, Windows XP, Solaris
- IPv6 対応のネットワーク製品
- 世界的な IPv6 バックボーンの構成
- ISP による商用ネットワークの IPv6 対応

→ すでに導入の検討をすべき時期に来ている?!

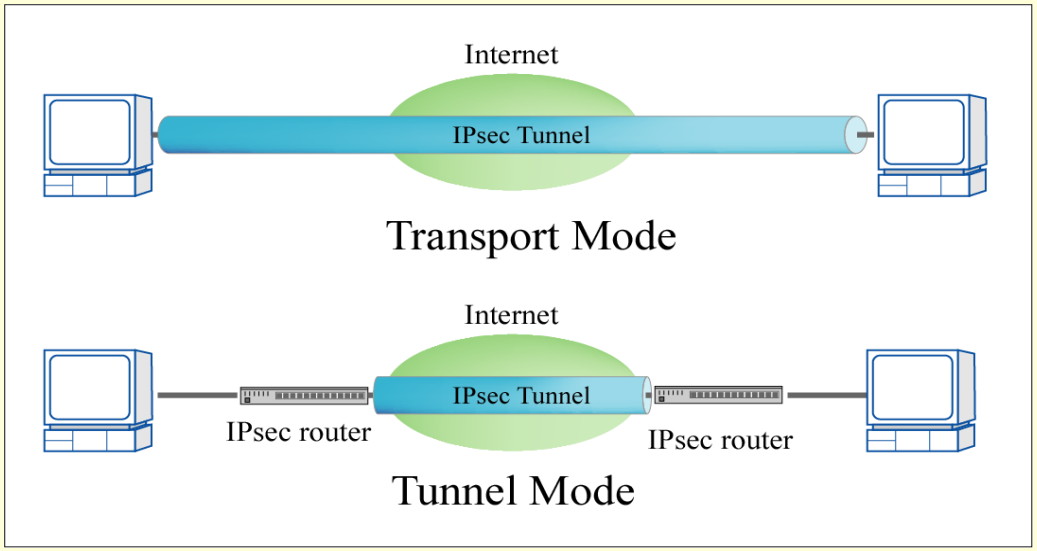
IPsec (IP Security)

IPec: 暗号化技術や認証技術を使用し、Internet を通して IP パケットを安全に運ぶ技術。

- 暗号化によって通信内容の秘密性を確保する。
- 通信内容の完全性を確保する(改竄を防ぐ)。
- 通信相手(計算機)が正しいことを認証する。

2つのモードがある

- Transport Mode
  PC 間の直接通信
- Tunnel Mode
  中継ルータが Ipsec 化を行う。



IPsec (IP Security)

Security に関するアルゴリズム:

- 本人性の認証
  Pre-Shared Key 認証、デジタル署名認証、改良型公開鍵認証
- 公開鍵暗号技術による共通鍵交換
  Diffie-Hellman
- 秘密対称鍵による暗号化
  3DES-CBC
- 鍵付きハッシュ関数による完全性の認証
  HMAC-SHA-1, HMAC-MD5-96

IPsec (IP Security)

- IKE (Internet Key Exchange)
  自動鍵交換機能
  SA (Security Association) の生成
  通信相手の本人性の認証

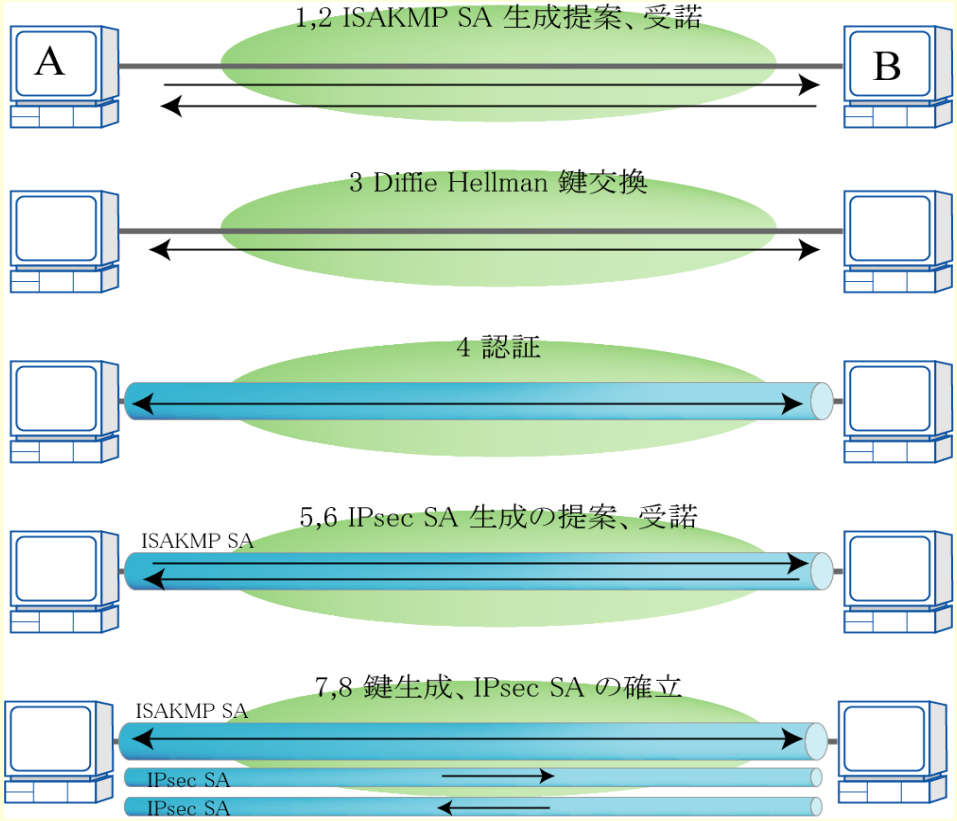
- ESP (Encapsulating Security Paryload)
  暗号化機能、(データの)完全性保証

- AH (Authentication Header)
  (パケット全体の)完全性保証

SA (Security Association) 生成手順

A から B への SA 生成手順
IKE (Internet Key Exchange) Protocol

- A→B ISAKMP SA 生成提案
- B→A ISAKMP SA 生成受諾
- Diffie Hellman 鍵交換
- 認証値交換(本人性確認)
- A→B IPsec SA 生成提案
- B→A IPsec SA 生成受諾
- IPsec SA 鍵生成
- IPsec SA 確立



SA 通信の Security

- 暗号化(秘密性)
  - SA 確立時に交換された共通鍵を用いて暗号化を行う。
    DES, 3DES, RC5, IDEA, ...
  - 暗号化を行う前に、1つ前に暗号化したブロックとの XOR をとる。
    CBC (Cipher Block Chaining)
    → 同じ平文を暗号化しても出力が異なるようにするため。
  - 最初に暗号化を行うブロックは Initiation Vector (IV) と呼ばれる乱数値との XOR をとってから暗号化する。
- 認証(データの完全性)
  - 鍵付き一方向ハッシュ値との整合性をとることにより認証を行う。
    ハッシュ関数: HMAC-MD5, HMAC-SHA-1
- 鍵の安全性
  - 暗号化の鍵は定期的に自動更新する(rekey)=SA の再生成。

通信セキュリティに関する予備知識

- ハッシュ関数
- 鍵交換技術 (Diffie Hellman)
- 共通鍵暗号
- 公開鍵暗号
- 公開鍵証明書(X509)

Diffie Hellman

Diffie Hellman: (Whitfield Diffie and Martin E. Hellman, 1976)

公開鍵暗号技術による共通鍵交換アルゴリズム

（公開鍵暗号技術開発の先駆的研究となった）

以下の2つの要件を満たす：

1. 離れた2者間が通信を行い、両者が知る鍵（となる数値）を生成する。
2. 2者間の通信を全て傍受しても鍵を知ることは事実上不可能。

この鍵交換技術は、離散対数問題を解くことが困難であることに立脚している。

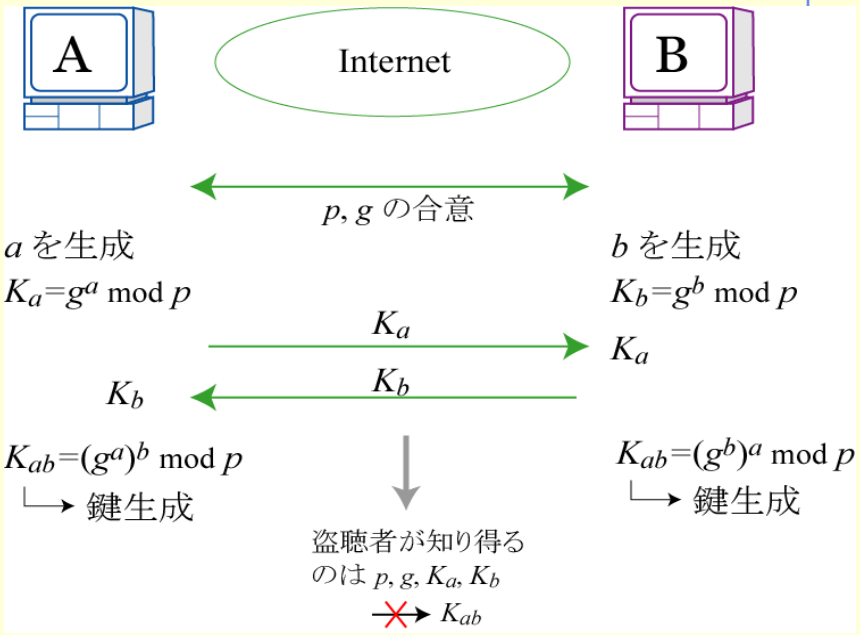
Diffie Hellman Algorithm

あらかじめ法となる素数 p、底となる数 g の組を決めておく。

A は乱数により秘密キー a $(0 \leqq a < p)$を生成、
$K_a = g^a \bmod p$ により公開キー K_a を作る。

B は乱数により秘密キー b $(0 \leqq b < p)$を生成、
$K_b = g^b \bmod p$ により公開キー K_b を作る。

A → B: K_aを送信
B → A: K_bを送信



A, B はそれぞれ、$K_{ab} \equiv g^{ab} \bmod p = (K_a)^b \bmod p = (K_b)^a \bmod p$ の式により
K_{ab}を計算、それをもとに(ハッシュ関数などを用いて)共通鍵を生成する。

盗聴者は、p, g, K_a, K_bの全てを知っていても(盗聴しても) K_{ab}を計算することは困難。
$a = \log_g K_a \,(mod\, p)$ が解ければ計算できる (離散対数問題)。

Diffie Hellman 公開値

IKE で使用している DH 公開値の組

DH グループ1

g = 2

p = 155251809230070893513091813125848175563133404943451431320235
119490296623994910210725866945387659164244291000768028886422
915080371891804634263272761303128298374438082089019628850917
069131659317536746955176311984337163722100721057791 9

DH グループ2

g = 2

p = 179769313486231590770839156793787453197860296048756011706444
423684197180216158519368947833795864925541502180565485980503
646440548199239100050792877003355816639229553136239076508735
759914822574862575007425302077447712589550957937778424442426
617334727629299387668709205606050270810842907692932019128194
467627007

Hash Function

(一方向)ハッシュ関数

- 与えられた任意の文章から、固定長のハッシュ値を計算する関数
- 元文の1ビットを変更してもハッシュ値が大きく変化する
- ハッシュ値から元の文章を導き出すことはできない (一方向性)
- 同じハッシュ値を持つ別の文章を作ることはきわめて困難

代表的なハッシュ関数

- MD5 (Message Digest 5)
  128 bit ハッシュ関数。
  Ronald Rivest らが開発。RFC 1321。
- SHA-1 (Secure Hash Algorithm 1)
  160 bit ハッシュ関数。
  1995年に米国標準技術局(NIST)によりアメリカ政府の標準ハッシュ関数として採用された。

鍵付きハッシュ関数

- 鍵を元文に加えてからハッシュ値を生成するアルゴリズム。
- 鍵を知らなければハッシュ値を生成することができない。
- データ通信の改竄検出に用いられる。

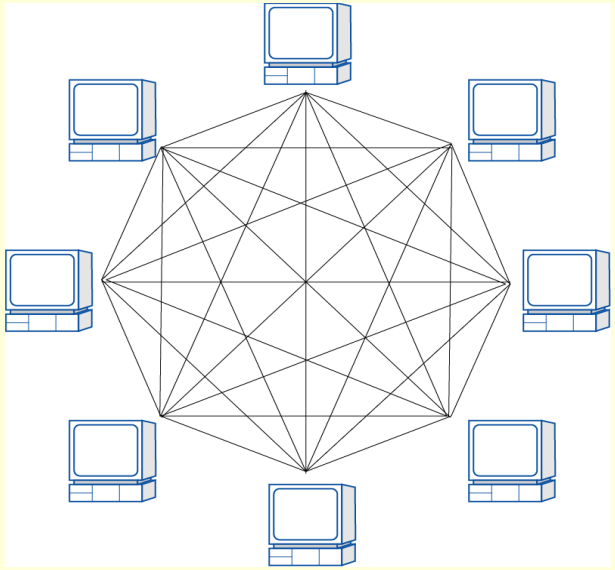
Common Key Encryptosystem

共通鍵暗号(対称鍵暗号)
平文の暗号化と復号に同じ鍵を使用する暗号化アルゴリズム。

特徴:

- 暗号化・復号の速度が速い。チップ化しやすい。
- 鍵を秘密にしつつ共有しなければならない。
  → どうやってキーを相手に送るか?
- 秘密通信を行うグループ毎に鍵が必要。
  → 大量のキーを秘密管理できるか?

例えば N 人の間で全ての 1:1 通信を行う場合
N(N-1)/2 個の暗号鍵が必要。
各人が (N-1) この秘密鍵を管理しなければならない。



Common Key Encryptosystem

代表的な共通鍵暗号

- DES (Data Encryption Statndard)
  IBM社が提案、1976 年米国の標準暗号化アルゴリズムとして採用される。
  56 bit の鍵を用いる 64 bit ブロック暗号。
- 3DES (Triple DES)
  168 (または 112) bit 鍵による 64 bit ブロック暗号。DES を強化盤。
  鍵を 3つの 56bit 鍵に分け、暗号化、復号、暗号化の順に DES を3回適用する。
- AES (Advanced Encription Standard)
  NIST が公募した次世代暗号化標準。
  2000 年にベルギーの数学者の開発したRijndaelに決定。
- IDEA (International Data Encryption Algorithm)
  1992年にスイス工科大学のJ. L.MasseyとX. Laiによって開発された。
- RC5 (Rivest's Cipher 5)
  Ron Rivestによって開発された。RC2の後継。
  RSA Security社によって暗号解読コンテストが開催されたことで有名。

Public Key Encryptosystem

公開鍵暗号

- 暗号化と復号の作業を対を成す2つの鍵によりそれぞれ行う暗号化方式。
- 通常2つの鍵の片方を秘密鍵とし、もう一方を公開鍵とする。
- 秘密鍵によって暗号化された文書は公開鍵により、公開鍵により暗号化された文書は秘密鍵により復号する。
- 公開鍵で暗号化することにより、秘密鍵を所持する本人しか見ることのできない暗号化文書となる。
- 秘密鍵で文書のダイジェストを暗号化することにより、秘密鍵を所持する本人が暗号化した文書であることが確認でき、デジタル署名として機能する。

N 人の間で全ての 1:1 通信を行う場合
N 個の暗号鍵が必要。
各人が 1つの秘密鍵を管理すればよい。

- 公開鍵暗号は計算が複雑で時間を要する。

公開鍵暗号による暗号化と署名

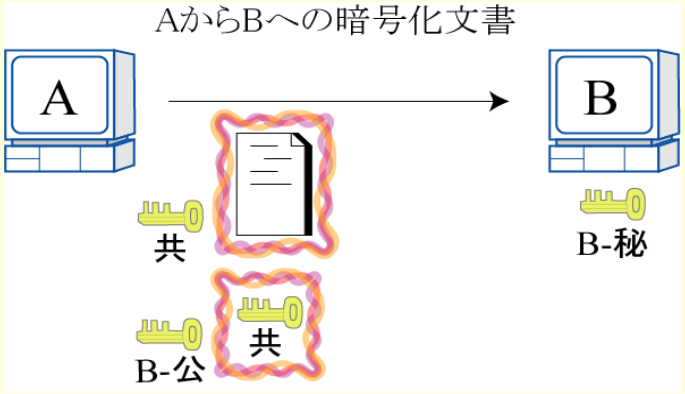
AからBへの暗号化文書送信

A:

1. (乱数生成した)共通鍵で文書を暗号化
2. 共通鍵をBの公開鍵で暗号化
3. 暗号化した文書と鍵をBに送る

B:

1. Bの秘密鍵で共通鍵復号
2. 共通鍵で文書を復号



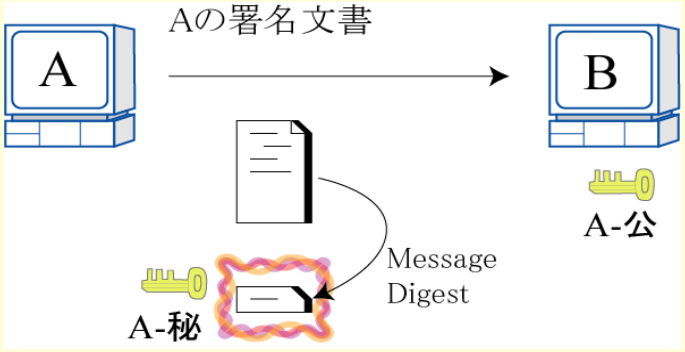
Aの署名文書送信

A:

1. ハッシュ関数により文書のダイジェストを作成
2. ダイジェストをAの秘密鍵で暗号化
3. 平文の文書と暗号化したダイジェストをBに送る

B:

1. ハッシュ関数により文書のダイジェストを作成
2. Aの共通鍵でダイジェストを復号
3. 1と2の値が一致することを確認



Public Key Encryptosystem

代表的な公開鍵暗号

- RSA 暗号
  1978年にRonald Rivest、Adi Shamir、Leonard Adlemanが考案
  巨大な整数の素因数分解の困難さに立脚

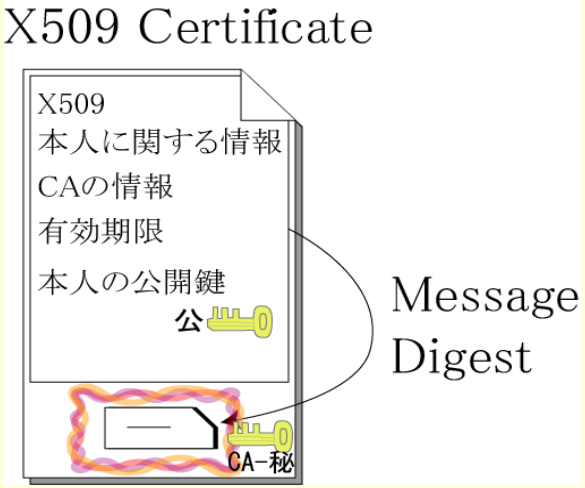
- ElGamal 暗号
  1982年に Netscape 社の Taher ElGamalが開発
  離散対数問題を解く困難さに立脚

- 楕円曲線暗号
  1985年にKoblitzとMillerが独立に考案
  楕円曲線上での離散対数問題を解く困難さに立脚
  短い鍵で高い安全性が確保できる。RSA に比べ高速計算できる。

X509

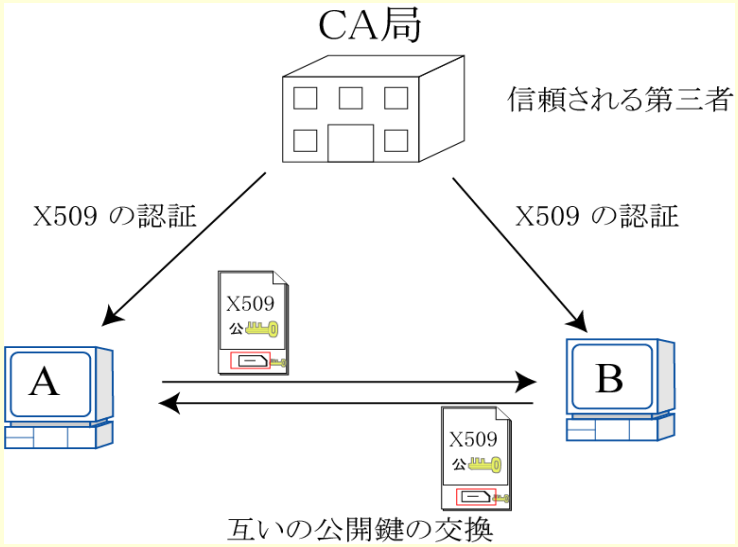
公開鍵証明書および証明書失効リストの標準仕様。
ITU(国際電気通信連合)が1988年に制定した。

記載されている公開鍵の真正性の CA による証明書。
公開鍵暗号による署名技術に基づく。
CA (Certificate Authority): 認証局。信頼される第三者。



主な記載内容

- 公開鍵のバージョン番号
- 証明書のシリアル番号
- 本人の公開鍵情報
- 認証局(CA)情報
- 証明書の有効期間
- 証明される本人の情報

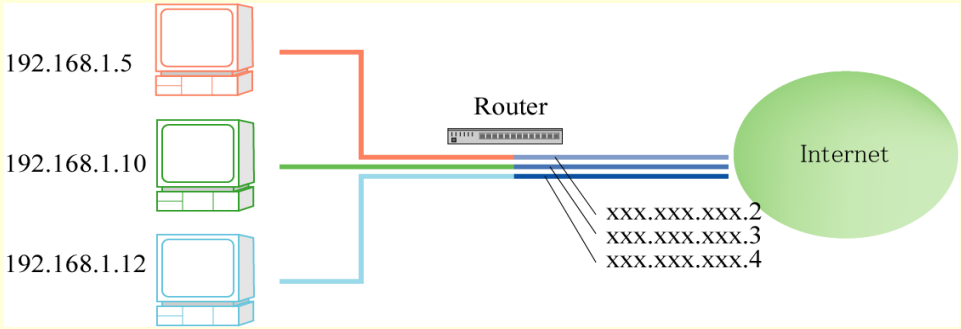


SSL や TSL などの通信における認証を始め幅広く用いられている。

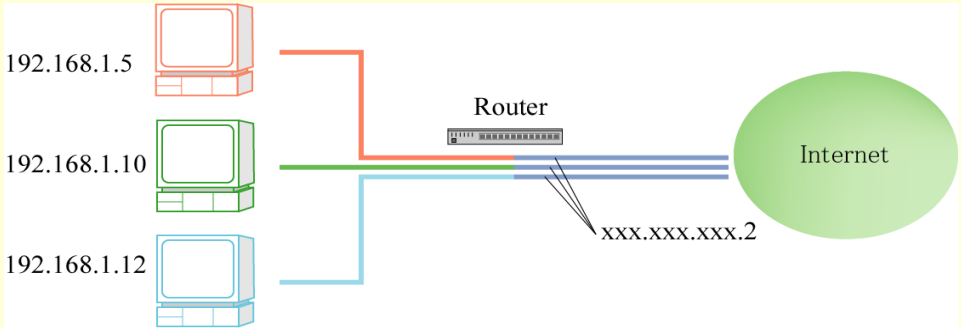
Network Address Translation (NAT)

内部ネットワークの IP アドレスを、グローバル IP に等価的に相互変換して通信する技術。
IP アドレス枯渇に対応するために導入された。
内部ネットワークを外部から隠すセキュリティ上の効果もある。
n:n のマッピング (NAT)、n:1 のマッピング (IP Masquerade) などがある。

NAT



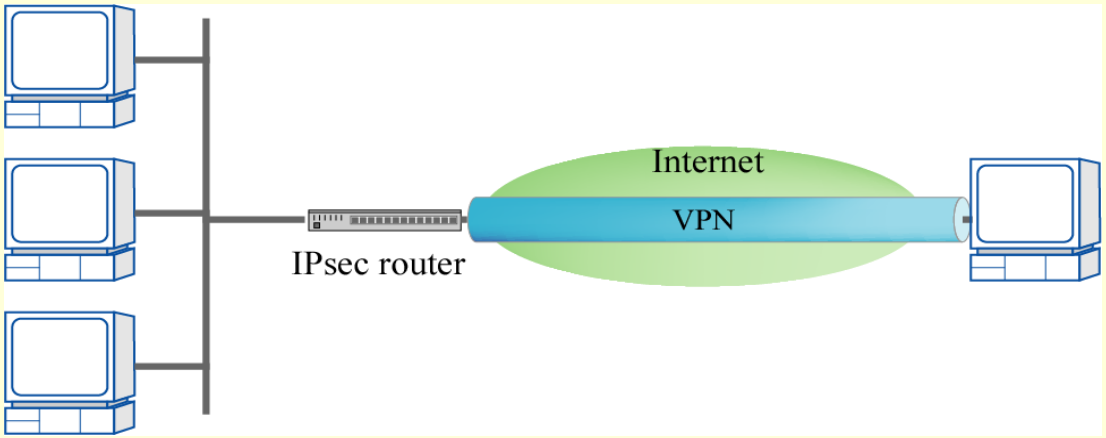
IP Masquerade



Virtual Private Network (VPN)

インターネット上に認証・暗号化技術を用いて仮想的な専用回線を構築する技術。

IPsec, PPTP, L2TP, ssh などがトンネル構築に が用いられる。



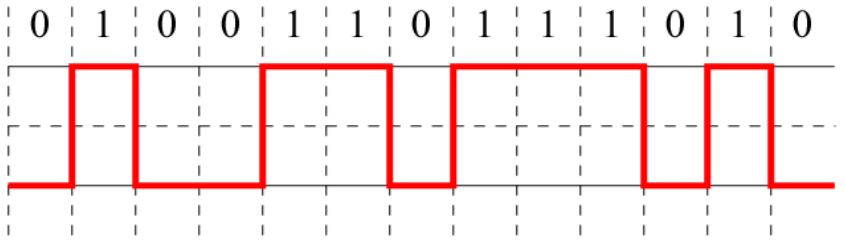
伝送路符号化

目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることに
   より制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号



・1 の時 H, 0 の時 L をとる。
・クロック抽出が困難な場合あり。

使用：RS232C

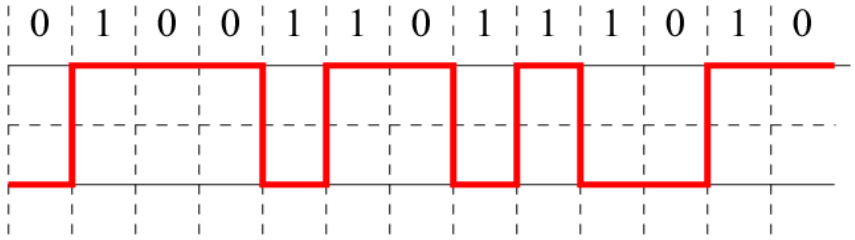
伝送路符号化

目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることに
   より制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号



・1 の時 信号を反転させる。
・クロック抽出が困難な場合あり。

使用：FDDI, 100BaseFX

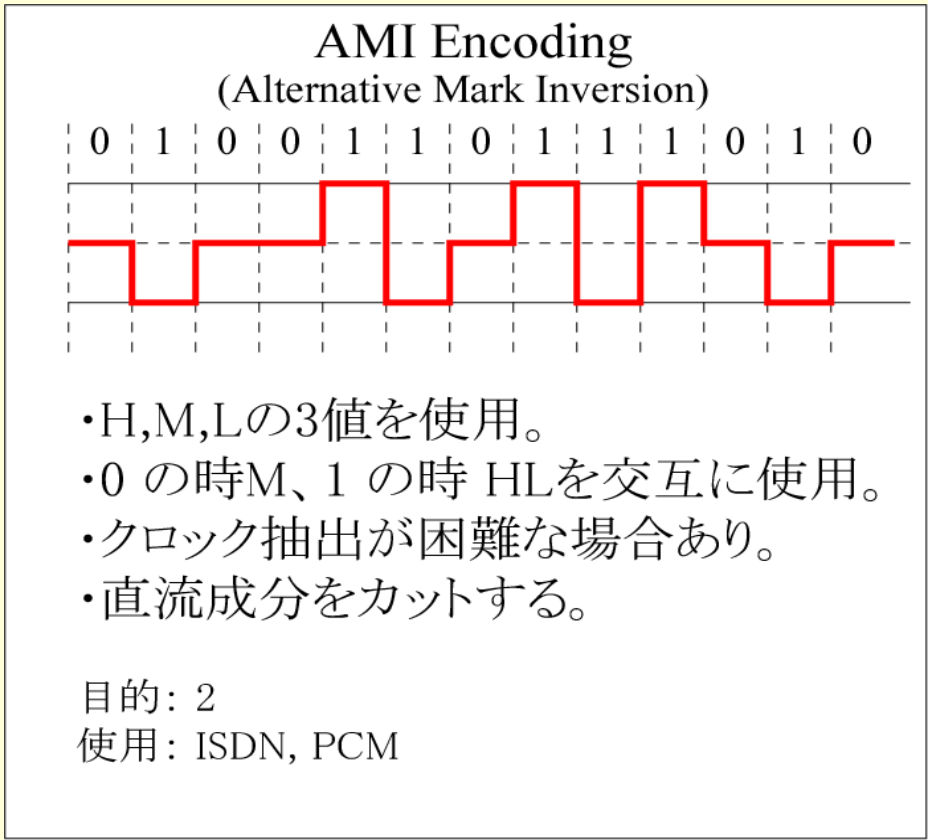
伝送路符号化

目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることに
   より制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号



伝送路符号化

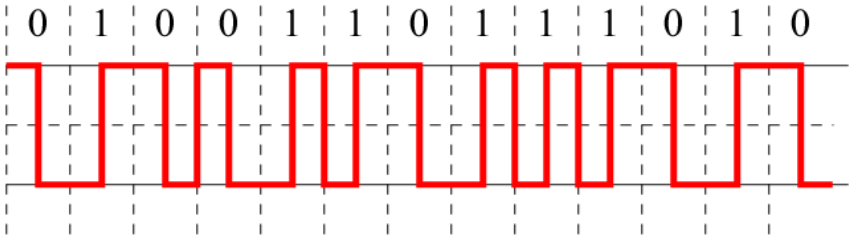
目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることに
   より制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号

Manchester Encoding



・1 の時 L→H, 0 の時 H→ L。
・必ずビットの・途中で信号が反転するようにビット境界で調整。
・クロック抽出が容易。
・高い伝送帯域が必要。
目的:1
使用: Ethernet (10Base)

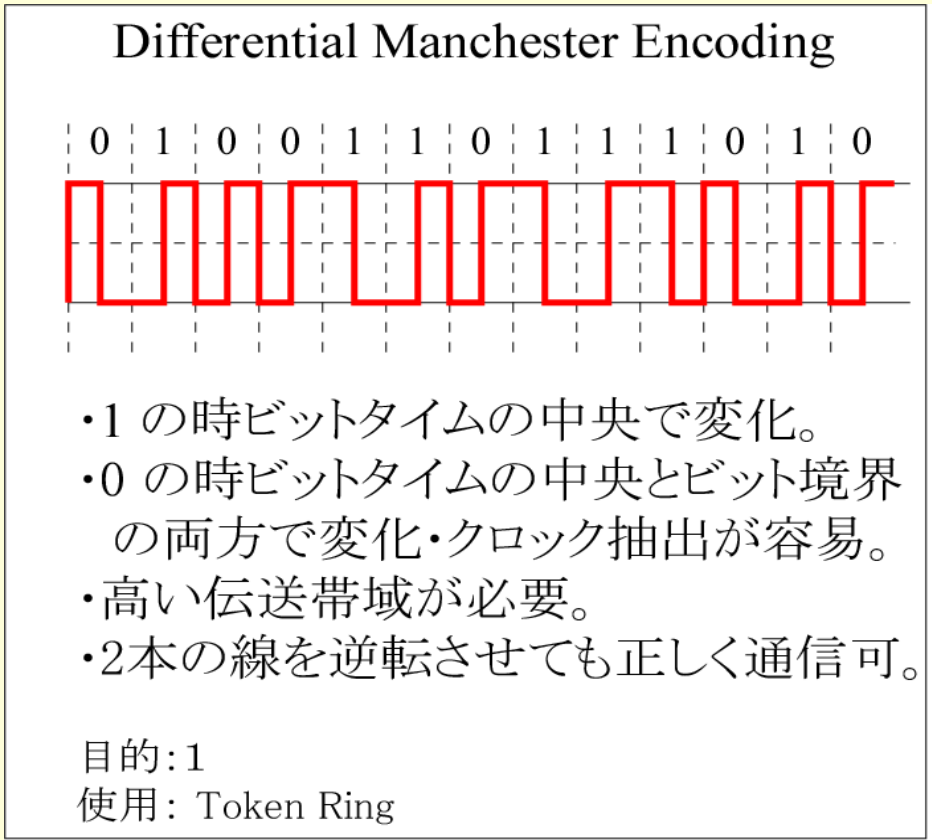
伝送路符号化

目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることにより制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号



伝送路符号化

目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることに
   より制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号

4B/5B Encoding

・4bit の情報を 5bit に拡張。
・1と0の偏りを減らし、低周波成分を削減する。
・0が3個連続するシンボルは制御用を除き使用しない

目的: 2,4
使用: 100Base, FDDI

4 bit data	5 bit code	symbol
	00000	Q (Quit)
	11111	I (Idle)
	00100	H (Halt)
	11000	J
	10001	K
	00101	L
	01101	T
	00111	R (Reset)
	11001	S (Set)
0000	11110	0
0001	01001	1
0010	10100	2
0011	10101	3
0100	01010	4
0101	01011	5
0110	01110	6
0111	01111	7
1000	10010	8
1001	10011	9
1010	10110	A
1011	10111	B
1100	11010	C
1101	11011	D
1110	11100	E
1111	11101	F

伝送路符号化

目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることに
   より制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号

8B/10B Encoding

・8bit の情報を 10bit に拡張。
・1つのビットパターンに複数の符号を割り当て、効果的に直流成分の抑制を行う。
・エラー検出機能あり。

目的: 2,4
使用: 1000BaseT, Fibre-Channel (ANSI X3T11)

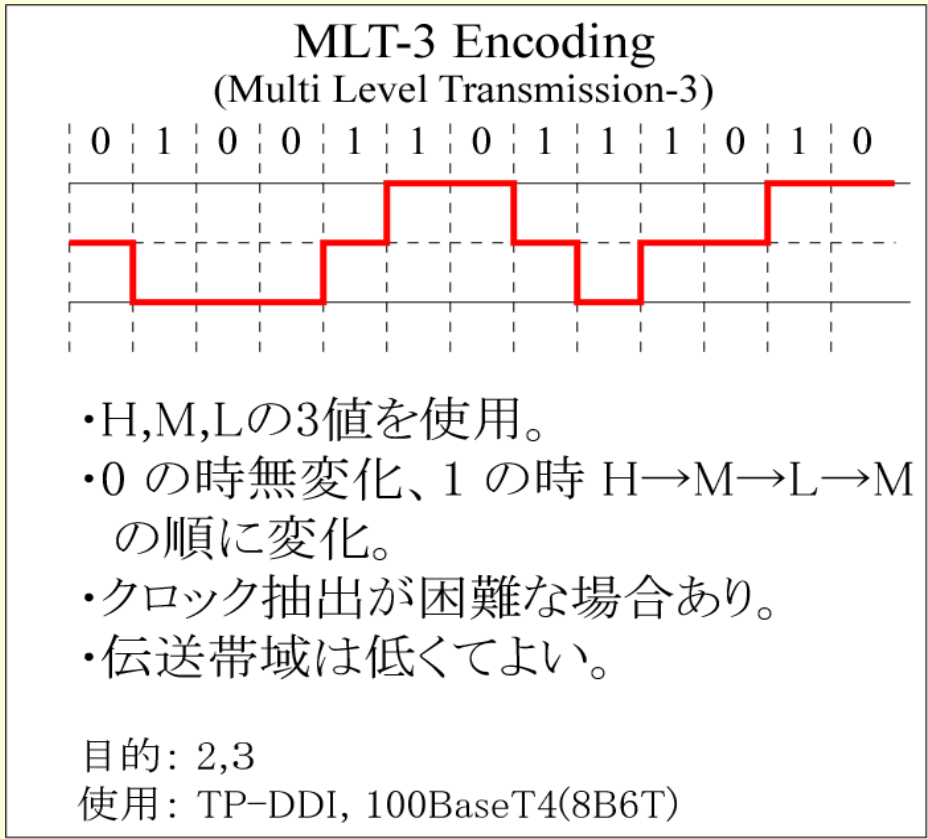
伝送路符号化

目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることにより制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号



伝送路符号化

目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることに
   より制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号

PAM-5 Encoding
(Alternative Mark Inversion)

・+2, +1, 0, −1, −2 の5値を使用。
・連続2ビットのデータを4値に割り当て、
  残る1値を制御用に用いる。

目的: 3,4

伝送路符号化

目的

1. 同期のためのクロックを伝える。
2. 直流成分のカット(ベースラインの安定化)。
3. 伝送速度(ビットレート)の向上。
4. 符号に冗長性を持たせることに
   より制御信号をアサインする。

主な符号化方式

- NRZ符号
- NRZI符号
- AMI符号
- マンチェスタ符号
- 差分マンチェスタ符号
- 4B/5B, 8B/10B 符号拡張
- MLT-3 符号
- PAM-5 符号
- 8B1Q4 符号

8B1Q4 Encoding
(Alternative Mark Inversion)

・+2, +1, 0, −1, −2 の5値を使用。
・8ビットのデータを4本の伝送路で
  並列送信。

目的: 3,4
使用: 1000BaseT

4B/5B 符号表

4 bit data	5 bit code	symbol
	00000	Q (Quit)
	11111	I (Idle)
	00100	H (Halt)
	11000	J
	10001	K
	00101	L
	01101	T
	00111	R (Reset)
	11001	S (Set)
0000	11110	0
0001	01001	1
0010	10100	2
0011	10101	3
0100	01010	4
0101	01011	5
0110	01110	6
0111	01111	7
1000	10010	8
1001	10011	9
1010	10110	A
1011	10111	B
1100	11010	C
1101	11011	D
1110	11100	E
1111	11101	F

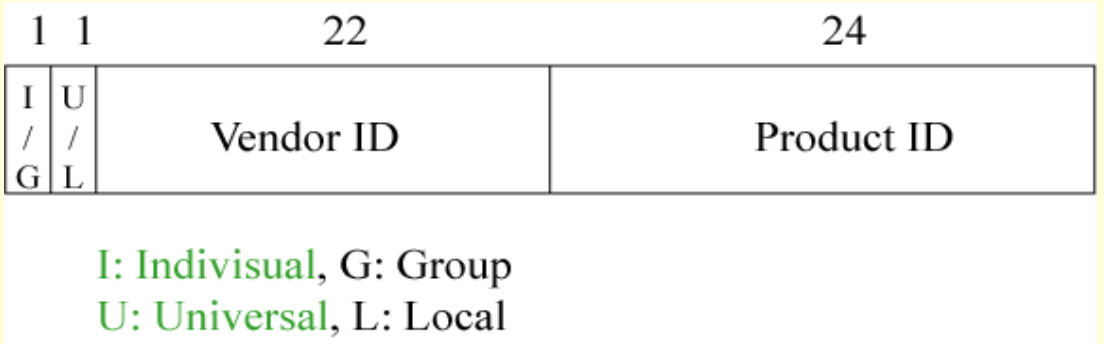
Ethernet

MAC Address

MAC Address ＝ Media Access Control Address,
Ethernet Physical Address

48bit の値を持ち、6バイトの 16 進数値で表現されることが多い。
例: 00:00:C9:08:91:43

Ether カード毎に固有の ID が割り振られ、カードに焼き付けられている。



Ethernet は MAC Address を使用してフレームを送信する。

→ MAC Frame

Ethernet

MAC Frame

(in octet)

8		6	6	2	3	42~1497		4
Preamble	SFD	Dst Addr	Src Addr	Len	LLC Hdr	User Data	Pad	FCS

SFD: フレーム開始デリミタ
Dst Addr: 送信先 MAC Address
Src Addr: 送信元 MAC Address
Len: ヘッダとデータの長さ（8bit単位）
LLC Hdr: 論理リンク制御ヘッダ
User Data: 送信データ
Pad: パディング
FCS: フレーム検査シーケンス

注： FCS には 32bit CRC 値が格納されており、受信側はこれを用いてエラー検出を行う。

LLC は誤データの再送などに用いられる(Token Ring)

→ MAC Address

Ethernet

Cyclic Redundancy Check (CRC)

巡回冗長検査。
連続して出現するデータ誤り(バースト誤り)の検出が可能な誤り検出方式。

Ethernet のCRC生成多項式 (32bit)： (1 00000100 11000001 0001110 1 10110111)

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$$

$M(x)$ を送りたいデータ r を次数(degree=32)とすると、$X^r M(x)$ を $G(x)$ で割った余り $R(X)$

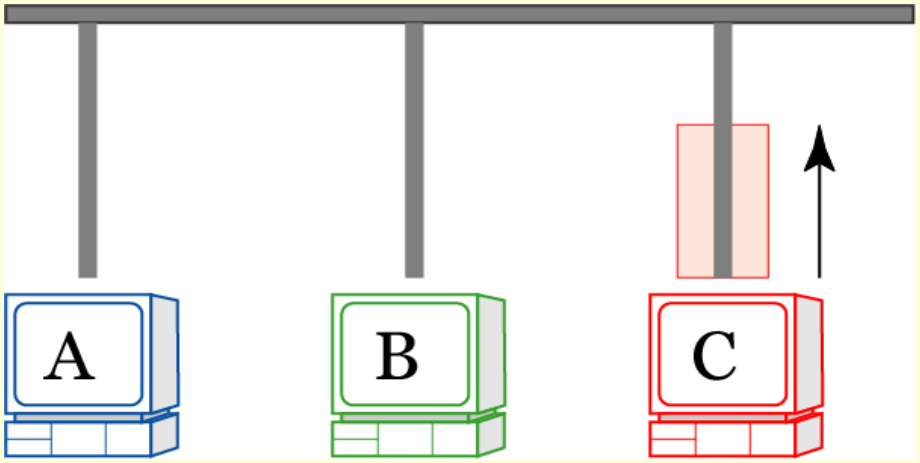
$$R(x) = x^r M(x) \bmod G(x)$$

を計算し、CRC 値 とする。送信されるデータは CRC 値を含めて
$X^r M(x) + R(X)$ となるので $G(x)$ により割り切れる。
このことを受信側が確認することにより、データの誤りを検出する。
但し上記の計算は全て Modulo2 代数にて行う。
エラー訂正はできない。

Ethernet

CSMA/CD

（Carrier Sense Multiple Access）

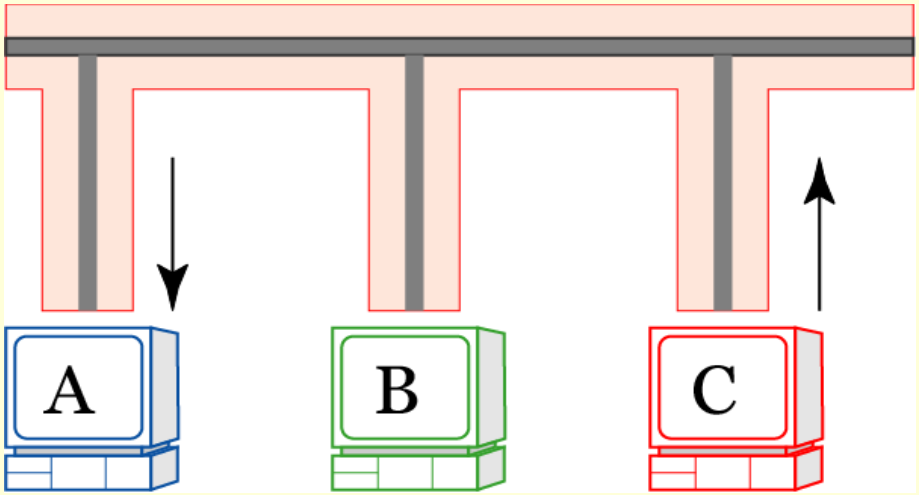


- ネットワーク上にデータが流れていないことを確認し、Cは（Aに対して）ネットワークにフレームを送信し始める。

Ethernet

CSMA/CD

（Carrier Sense Multiple Access）

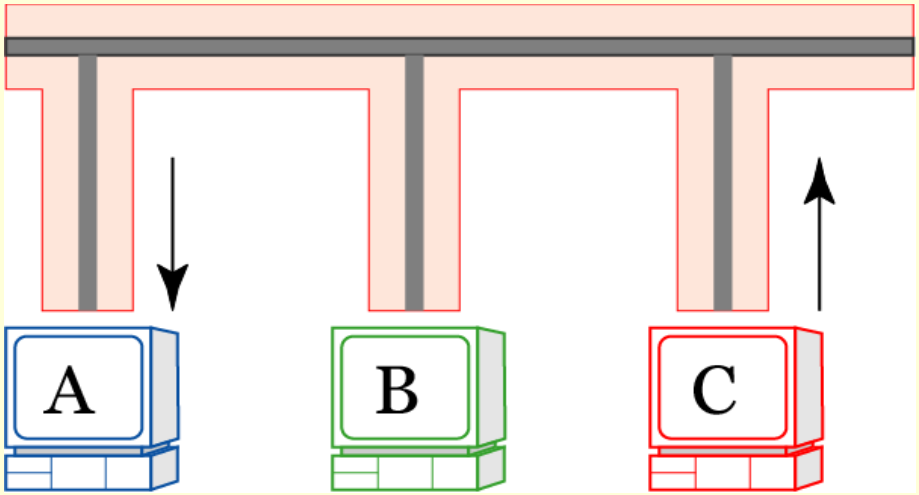


- 全てのホストは常にネートワーク上の信号を監視している。
- 自分の MAC Address 宛てのフレームがあれば受け取って処理する(ホストA)。

Ethernet

CSMA/CD

（Carrier Sense Multiple Access）



- Cは、ネットワーク上で最も遠い2点間の往復時間をえる時間の間フレームを出しつづけなければならない。

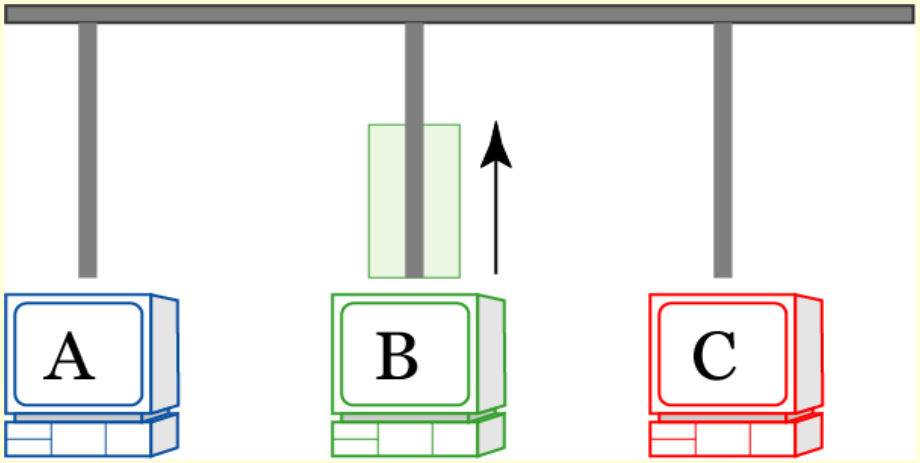
→ フレームは短かすぎてはいけない。
ネットワークケーブル（+リピータの遅延）は長すぎてはいけない。

- ネットワークケーブル上を信号が流れているので、Bはデータを送信できない。

Ethernet

CSMA/CD

（Carrier Sense Multiple Access）

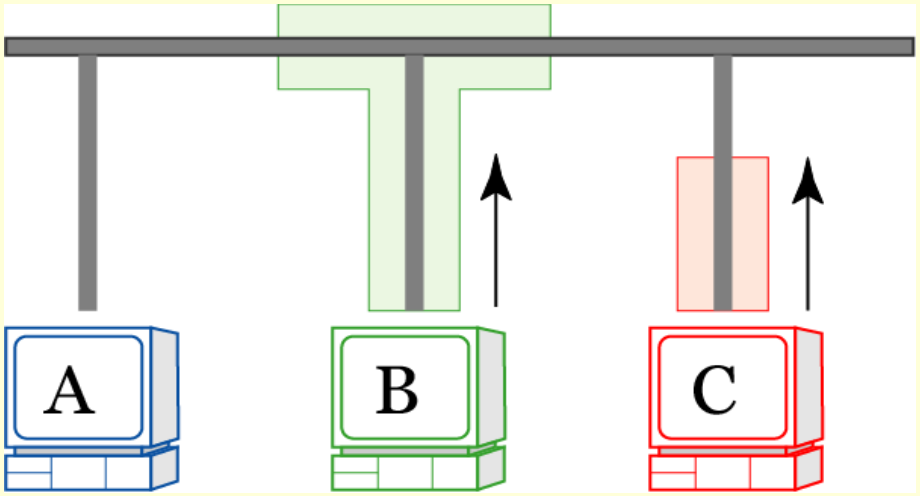


- Cがフレーム送出を終了し、ネットワーク上を流れている信号がなくなる。
- 一定のギャップ時間*の後、B はデータの送信を開始できる。

*10Base の場合で 9.6 μs

Ethernet

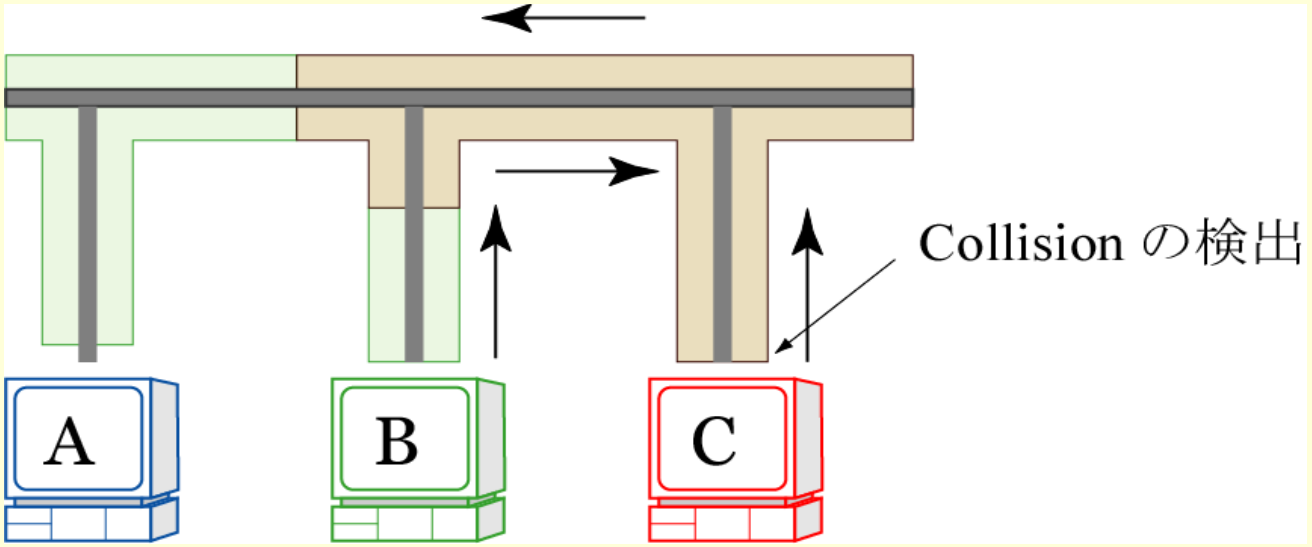
CSMA/CD
(Collision Detection)



- BとCがほぼ同時にフレームを送信し始めた場合

Ethernet

CSMA/CD
(Collision Detection)

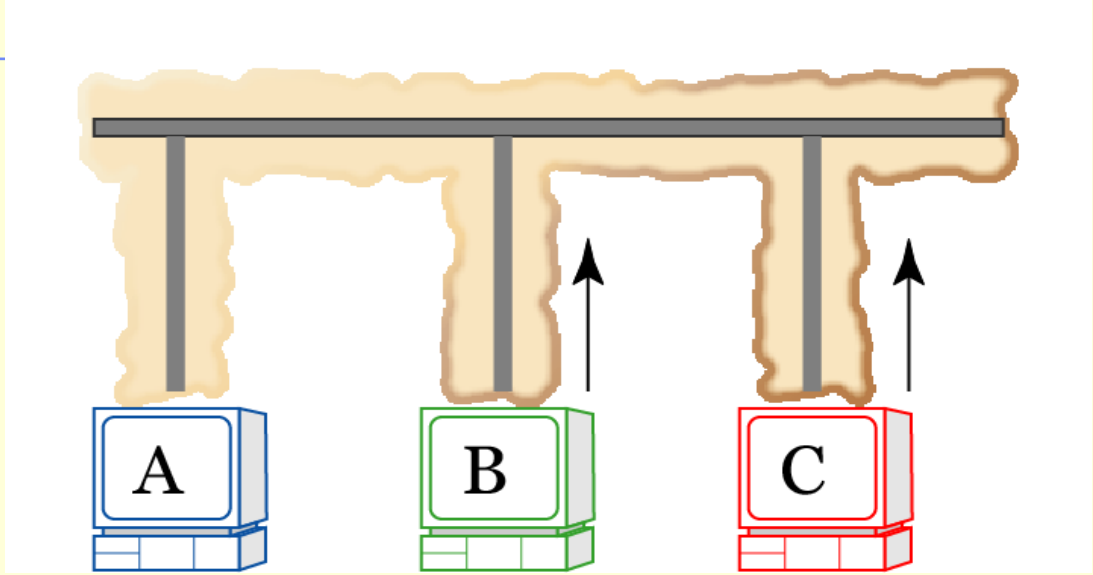


- 衝突(Collision)が発生する。
- 各送信中のノードは、ネットワーク上を流れているデータを監視し、自分が送出した信号と一致しているかどうかを確認している。

→ 衝突の検出

Ethernet

CSMA/CD
(Collision Detection)

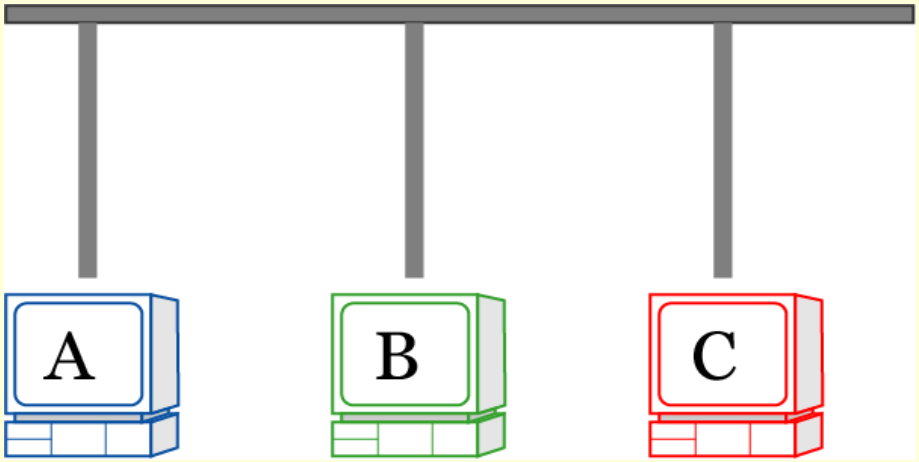


- 衝突を検出したノードは、一定時間* ジャミング信号を出す。

*10Base の場合で 3.2~4.8 μs

Ethernet

CSMA/CD
(Collision Detection)



- ジャミング信号を受け取ったノードは、乱数を振って得られた時間*だけ信号を出さずに待機。

[再送待ち時間] = [スロット時間]×[乱数]
0≦乱数< 2^k
k = min ([再送回数], 10)

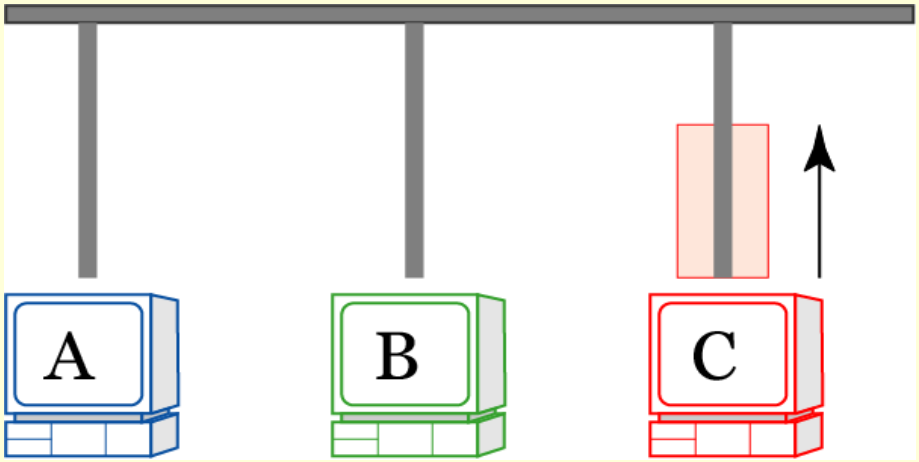
[スロット時間] = 51.2 μs (10Base の場合)

16 回続けて衝突した場合は再送を諦める

*不完全2進指数バックオフアルゴリズム

Ethernet

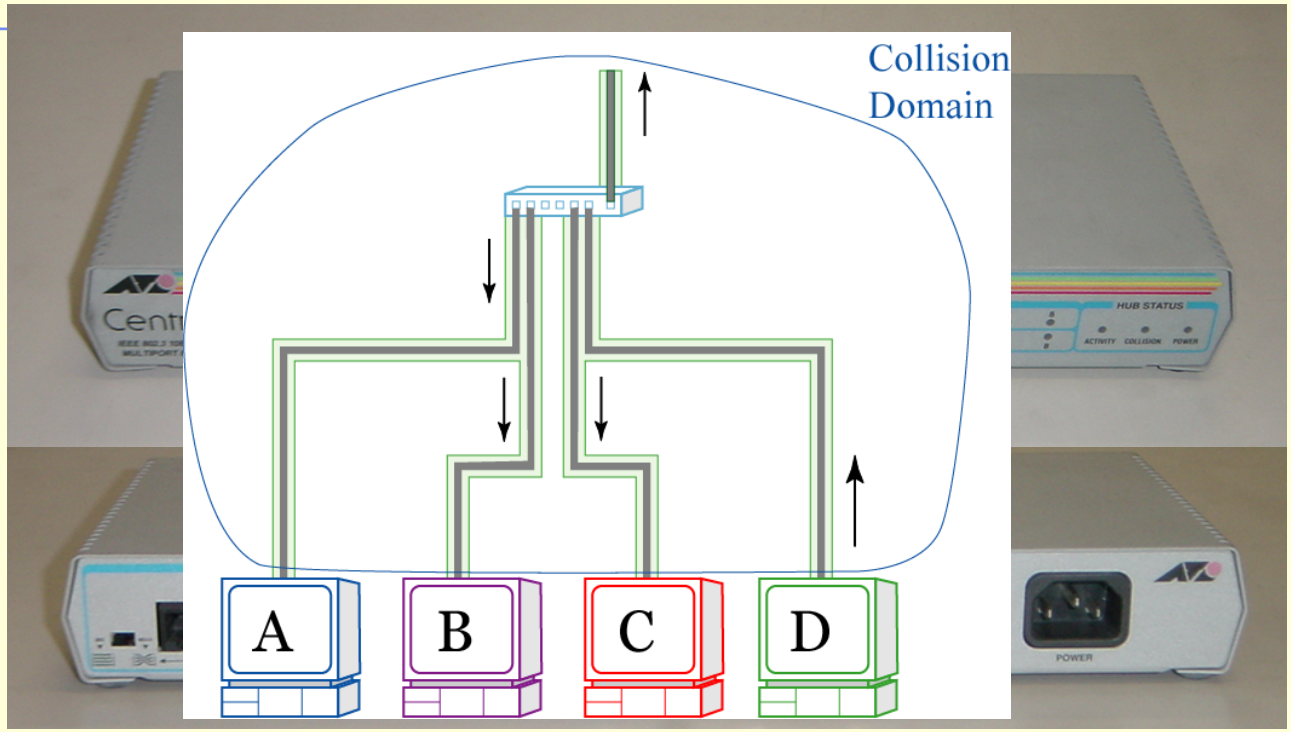
CSMA/CD
(Collision Detection)



- 待機を終了したノードは、フレーム送信を始めることができる。

Ethernet

HUB (multiport repeater)

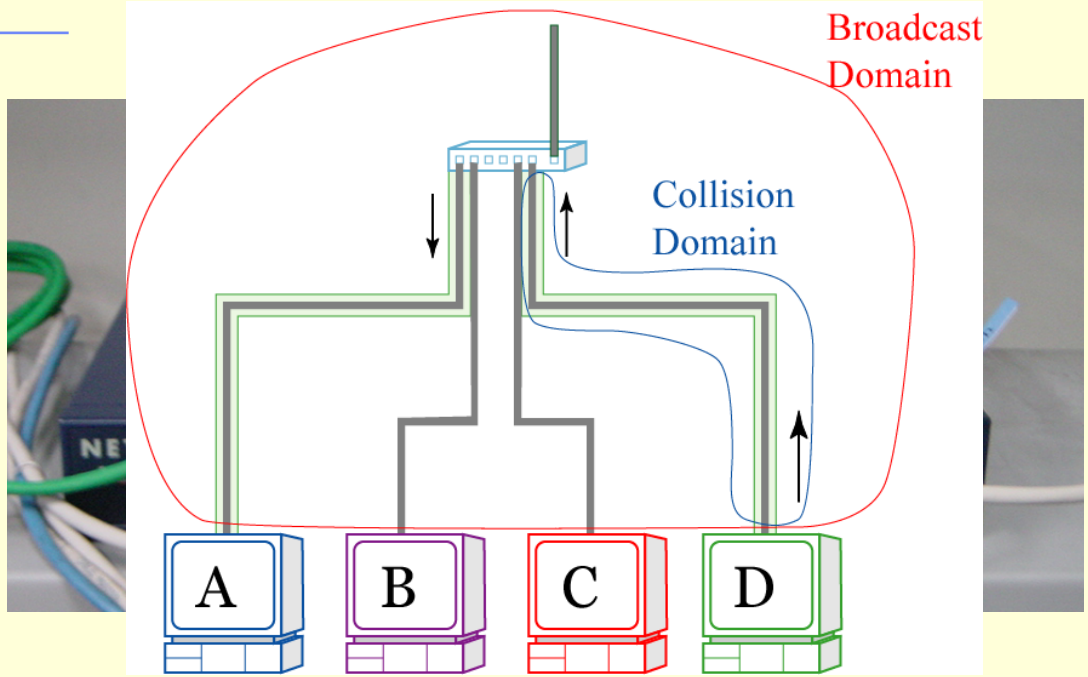


- 1つのポートからの信号を全てのポートに転送する。
- CSMA/CD を行う範囲(Collision Domain)を広げる。
- トラフィックが増えると性能が急激に低下する。
  一般的にはトラフィックを Band Width の30%以下に抑えるべき。

→ 4-Repeater Rule (10Base)

Ethernet

Switch



- 各ポートに繋がっている機器の MAC Address を自動的に学習する。
- 宛先 MAC Address の繋がっているポートのみにフレームを送信する。
  Collision Domain は小さくなる。別ポートのデータ転送に影響されない。
- フレームを一旦記憶してから転送する。
  転送速度の異なるネットワーク間を接続できる。
- 全2重通信(Full-Duplex) が可能になる。

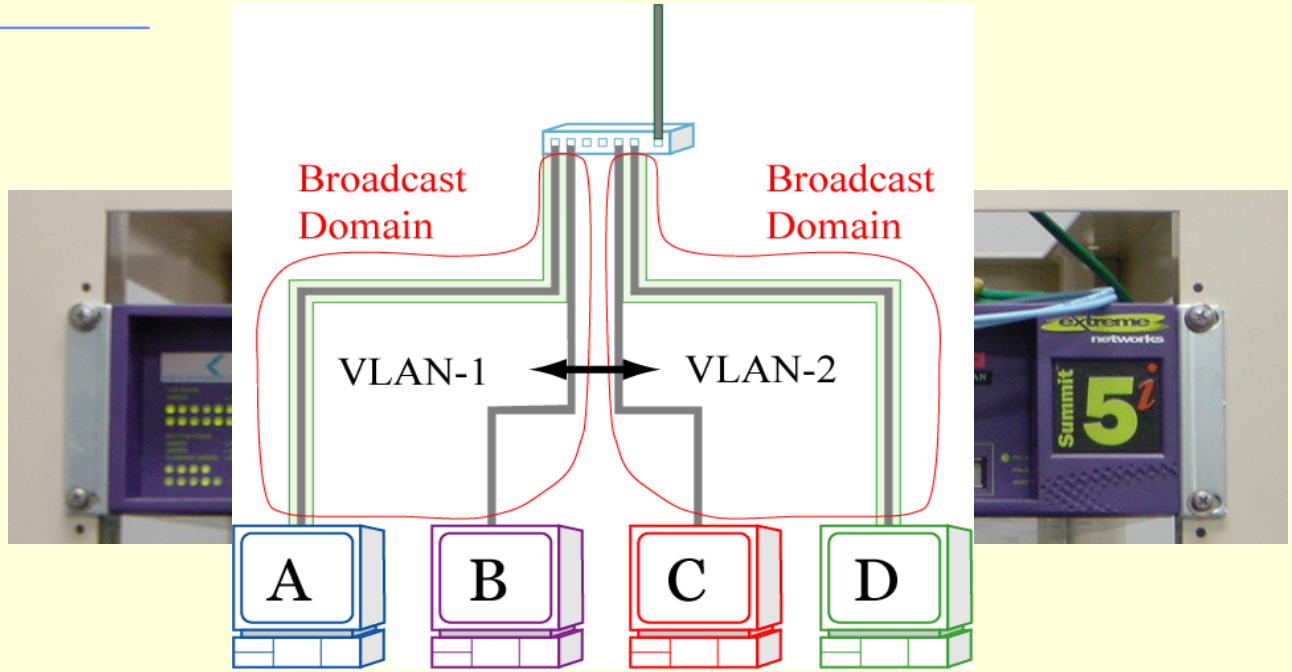
Switching Schemes

Switch によるデータ転送方式
(異なる速度のネットワーク間データ転送においては 1 のみ可能)

1. Store and Forward
   - MAC Frame を完全に受信してから送信
   - Collision や Short Frame をチェックし、エラーのない転送を行う
   - 他の方式に比べ、スループット・遅延に関して劣る
2. Cut Through
   - MAC Frame の先頭数 octet を受信した時点で転送を開始
   - Collision を中継する可能性がある
   - スループットが大きく、遅延が小さい。
3. Fragment Free
   - Collision の可能性がある先頭 64 octet を受信した時点で転送を開始
   - Collision の可能性が下がる。
   - スループット、遅延の劣化が比較的小さく抑えられる。

Ethernet

Layer-3 Switch



- 複数のポートを束ねて1つのネットワーク(VLAN)を構築することが可能。
- VLAN 間の通信を行うことができる。
- 本来ルータが持っていた機能を Switch に移すことで、ネットワーク間通信を高速化できる。

Ethernet

Router

- プロトコルの異なるネットワーク間（WAN など）を接続できる。
- 経路情報を交換し、各パケットの送信経路を決める。
  IP Datagram の TTL を 1 減らす(TTL が 0 になれば廃棄)。
- パケットフィルタリング機能
- アドレス変換機能
- 負荷分散機能
- IP カプセル化機能

FDDI
(4B/5B coding)

- 4 bit のデータを 5 bit に拡張して制御コードを定義。
- さらに NRZ/NRZI コーディングを行った後回線にデータを流す。
- 回線オーバーヘッドのため、物理転送速度 125 Mbps が データ転送速度 100 Mbps になる。

4 bit data	5 bit code	symbol
	00000	Q (Quit)
	11111	I (Idle)
	00100	H (Halt)
	11000	J
	10001	K
	00101	L
	01101	T
	00111	R (Reset)
	11001	S (Set)
0000	11110	0
0001	01001	1
0010	10100	2
0011	10101	3
0100	01010	4
0101	01011	5
0110	01110	6
0111	01111	7
1000	10010	8
1001	10011	9
1010	10110	A
1011	10111	B
1100	11010	C
1101	11011	D
1110	11100	E
1111	11101	F

FDDI

FDDI
(frame format)

Token

Preamble	SD	FC	ED

FCの特定のフラグを立てる

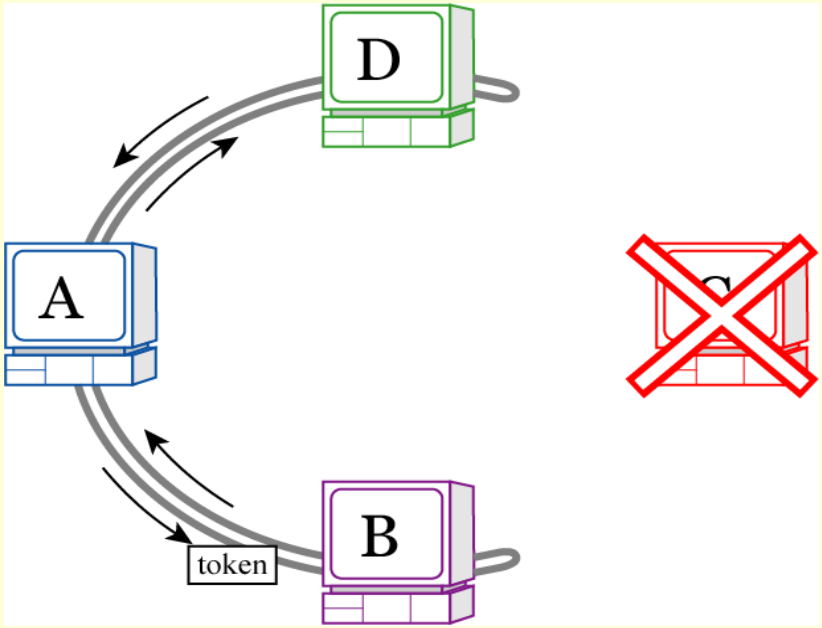
Data Frame

Preamble	SD	FC	DA	SA	INFO	FCS	ED	FS

Preamble: 16個以上のT
SD: Starting Delimiter JとKで表現
FC: Frame Control DAとSAの長さを指示
DA: Destination Address 宛先アドレス
SA: Source Address 送信元アドレス
INFO: Information 情報フィールド
FCS: Frame Check Sequence
ED: Ending Delimiter Tで表現
FS: Frame Status

→4B/5B符号化

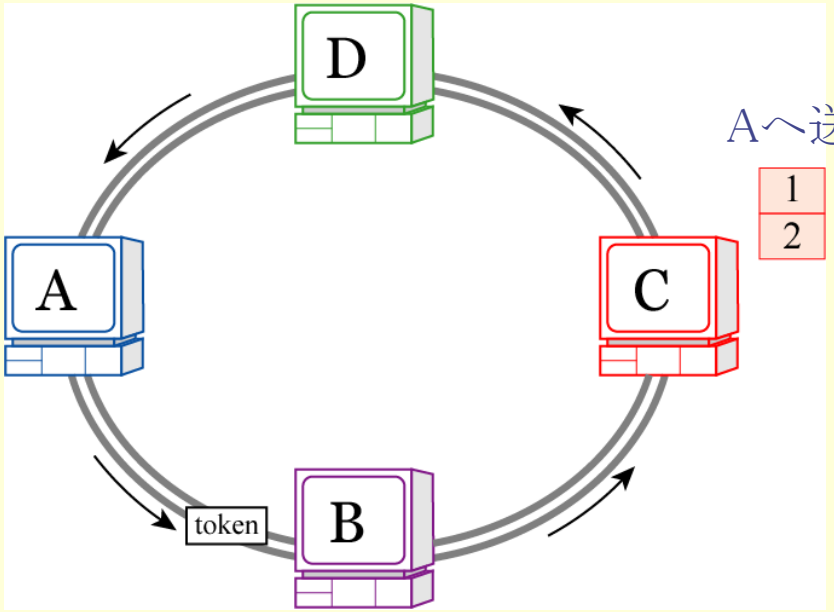
FDDI
(dual ring)



- 1つのノードあるいは1本にケーブルに異常が発見された場合は、自動的に接続を折り返すことによりリングを回復する。

FDDI

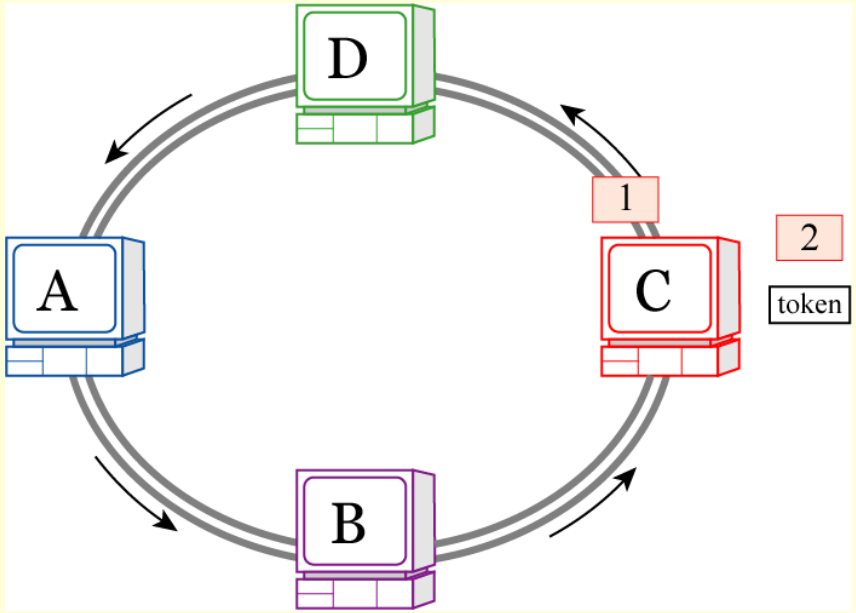
FDDI
(token passing)



- Token は FDDI リングを周回している。

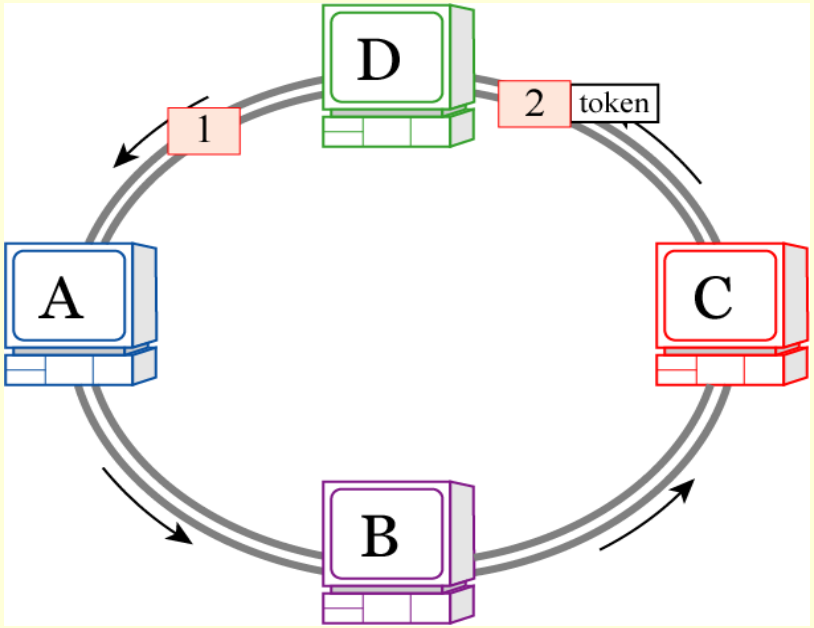
FDDI

FDDI
(token passing)



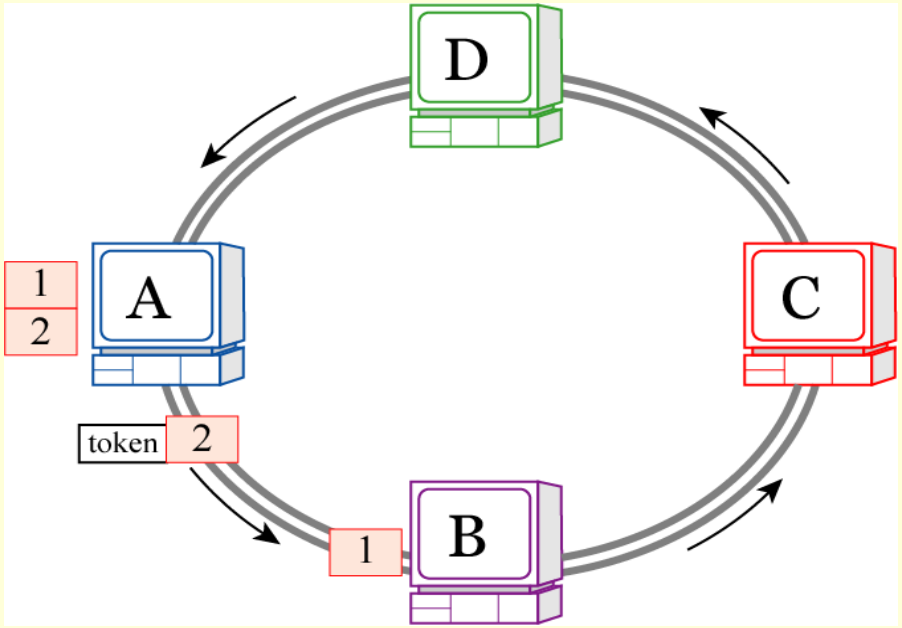
- Token を受け取ったノードはデータを送信できる。
  C は A に向けてデータ送信を始める。

FDDI
(token passing)



- C は 送信が終われば Token を送る。
  最後の送信データに付けて送る (early token release)。
- D は 受け取ったデータが自分宛てではないので、そのまま次に転送する。Token のみを受け取っても良い。

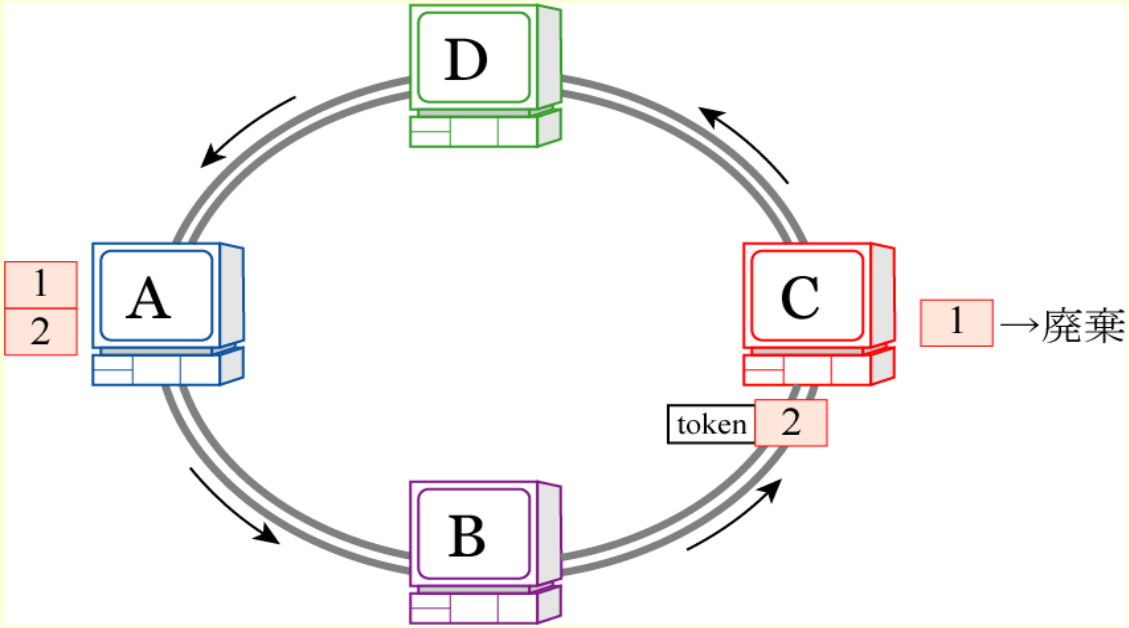
FDDI
(token passing)



- A は 受け取ったデータが自分宛てであるので、受け取ったデータをメモリにコピーして処理する。
- 受け取ったデータは、受信したことを示すフラグを付けて次に送出される。

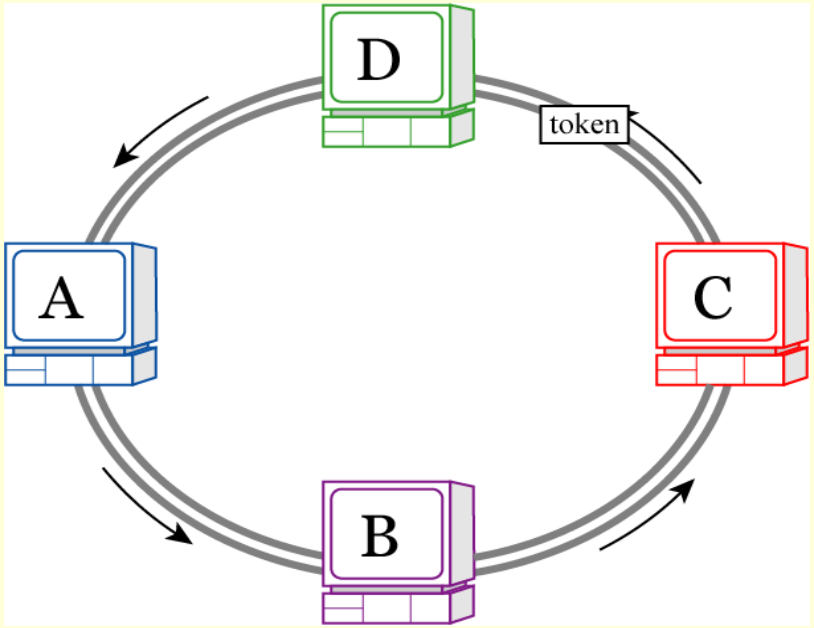
FDDI

FDDI
(token passing)



- C は 自分が送信元であるデータを受け取ると、リングから取り除く。

FDDI
(token passing)

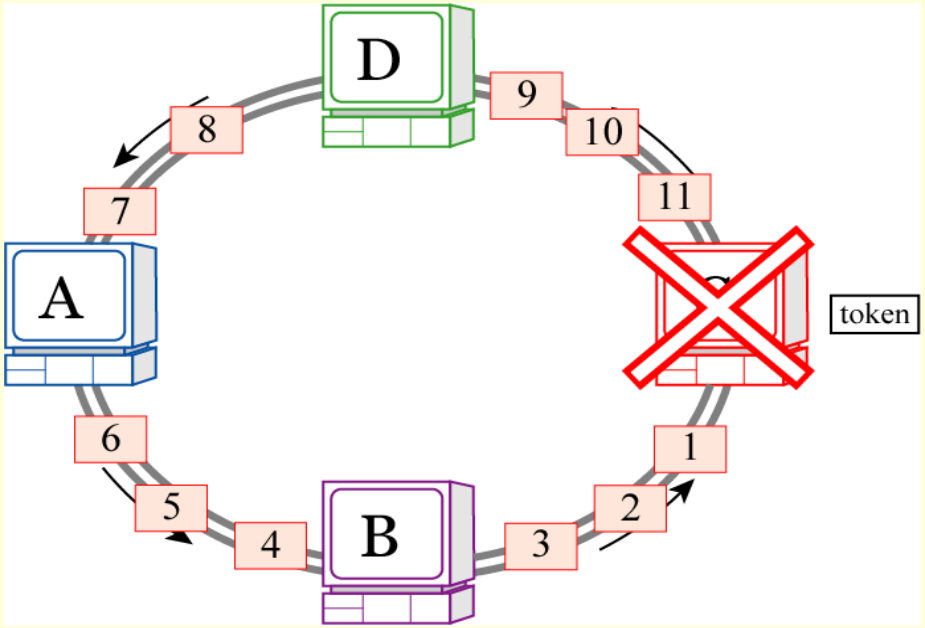


- 再び Token だけが回っている状態に戻る。

FDDI はこの Token Passing 方式により衝突を回避する。

FDDI

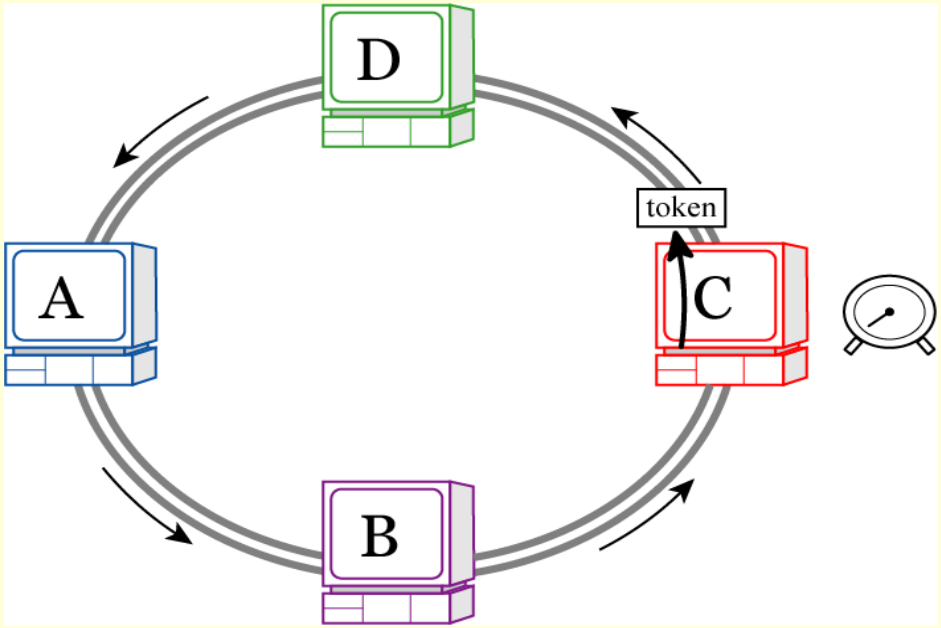
FDDI
(timed token protocol)



1つのノードが長時間 Token を占有してしまい、他のノードに Token が回らないという状況をどう防げばよいか？

1つのノードが Token を占有できる時間の上限を決めるという方式では、ノード数が増えるに従いってToken を受け取るまでの時間が長くなってしまう。

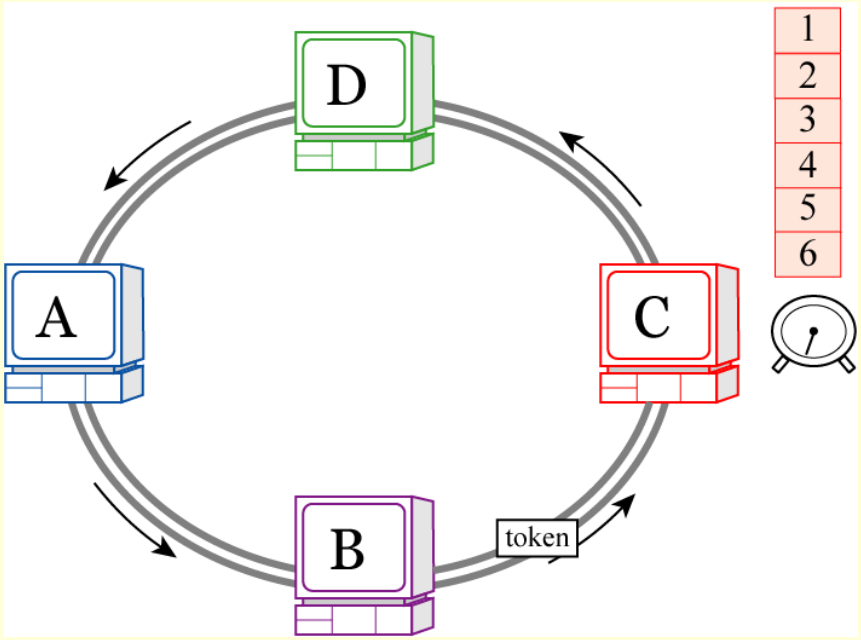
FDDI
(timed token protocol)



- 各ノードはそれぞれのダウンカウントタイマ(TRT)を持っている。
- C はデータを転送することなく Token を通過させたとき、タイマをあらかじめ決められた初期値(Min.TTRT値)に設定する。
- タイマは時間とともにカウントダウンしていく。

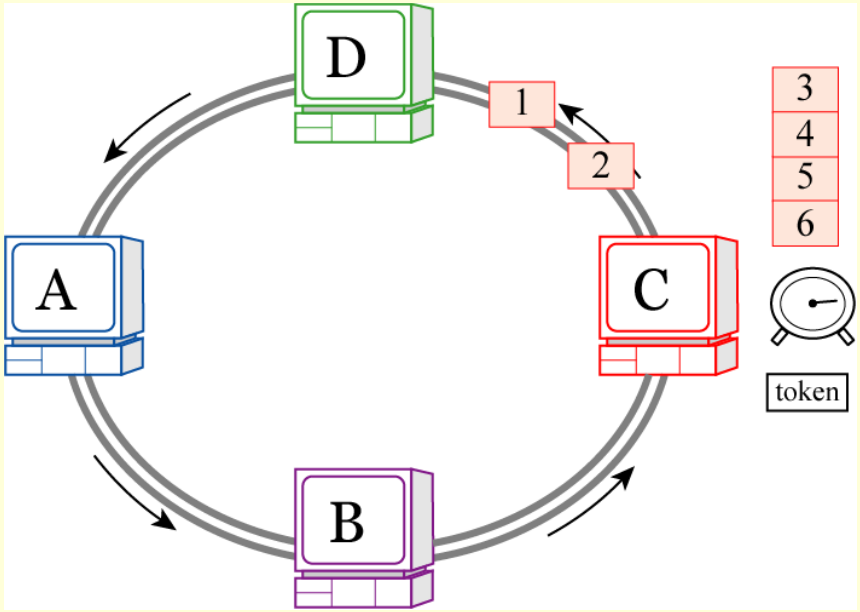
FDDI

FDDI
(timed token protocol)



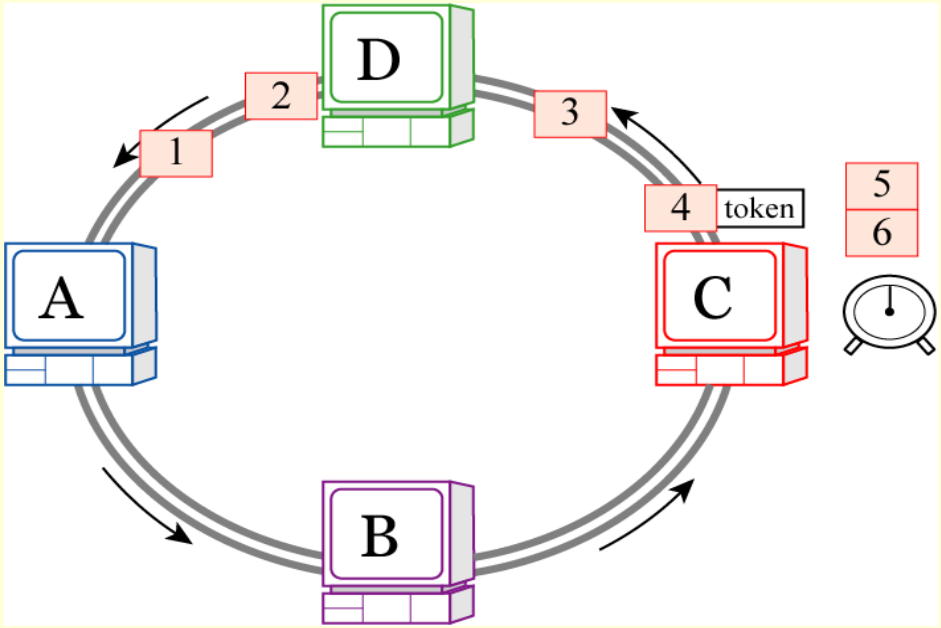
- C に送信したいデータが発生。

FDDI
(timed token protocol)



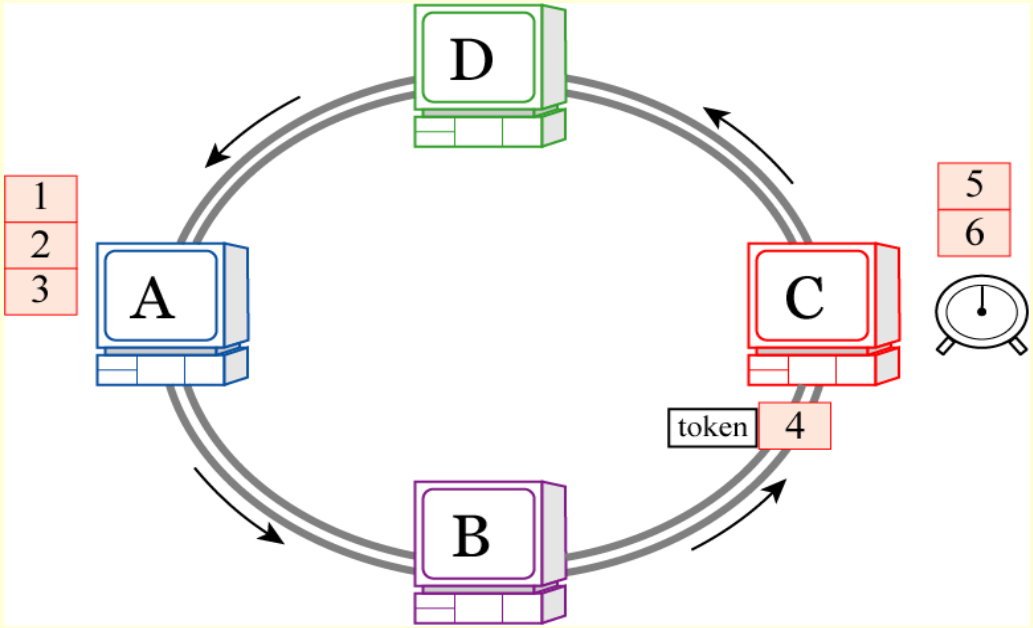
- C は Token を受け取ってデータ送信を始める。
  この時タイマは初期化されない。

FDDI
(timed token protocol)



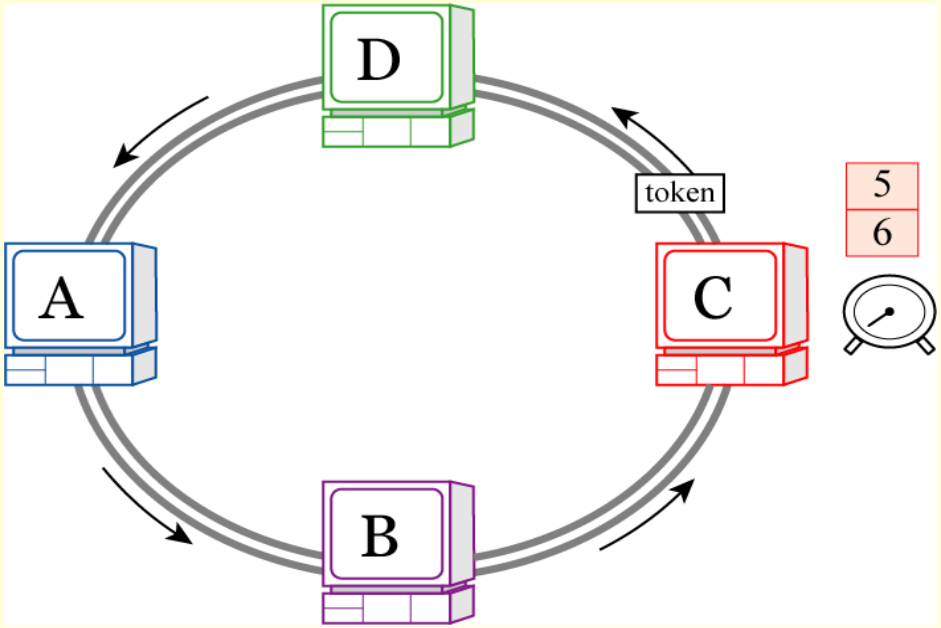
- タイマが 0 になると C は Token を放棄しなければならない。

FDDI
(timed token protocol)



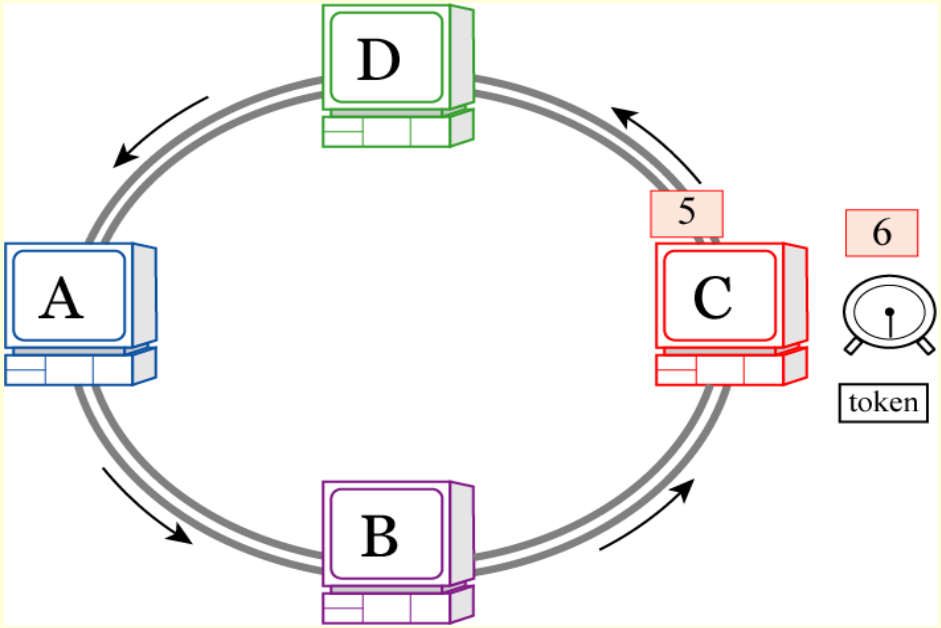
- 再び Token が回ってきたとき、C のタイマは 0 であるので、C は Token を受け取ることができない。

FDDI
(timed token protocol)



- Token をそのまま通過させることにより、C のタイマは初期値に設定し直される。
  (別のノードがこの Token を受け取って送信を開始することができる。)
- C は次回 Token を受け取った時にデータを送信できる。
  (但し、他のノードが時間を使い切っていなければ)

FDDI
(timed token protocol)



この Timed Token Protocol により

- 1つのノードが Token を不当に占有することはない。
- 必ず Min.TTRT×2 の時間の間に1回 Token を受け取ることができる。

Reference Pages

- ICANN (The Internet Corporation for Assigned Names and Numbers)
- IANA (Internet Assigned Numbers Authority)
- JPNIC (Japan Network Information Center)
- RFC (Request for Comments)
- ISOC (Internet Society)
- IETF (Internet Engineering Task Force)
- IEEE (Institute of Electrical and Electronics Engineers)
- ISO (International Organization for Standardization)
- ANSI (American National Standards Institute)

References

- 新プロトコルハンドブック, 朝日新聞社, 1994.
- 通信プロトコル事典, アスキー出版, 1996.
- ギガビットネットワーク, ソフトバンク, 1995.
- 最新 LAN ハンドブック, 宮越・角田, 秀和システム, 2002.
- WIDE, School of Internet, Tutorials , http://www.soi.wide.ad.jp/class/ .
- IPsec 徹底入門、小早川知昭、翔泳社、2002.
- IPv6 教科書、江崎浩監修、IDG ジャパン、2002.
- 分かりやすい暗号学、高田豊、米田出版、2000.
- e-Words 情報通信事典, http://e-words.jp/ .