情報ネットワーク

伊藤　嘉浩

講義内容

- 1）序論
- 2）3）ネットワークアーキテクチャ
- 4）5）伝送路と物理層
- 6）誤り制御方式
- 7）MACプロトコル
- 8）中間試験
- 9）データリンク層プロトコル
- 10）11）データ交換とネットワーク層
- 12）13）14）TCP/IP
- 15）期末試験
- 16）統括

13) TCP/IP (2)

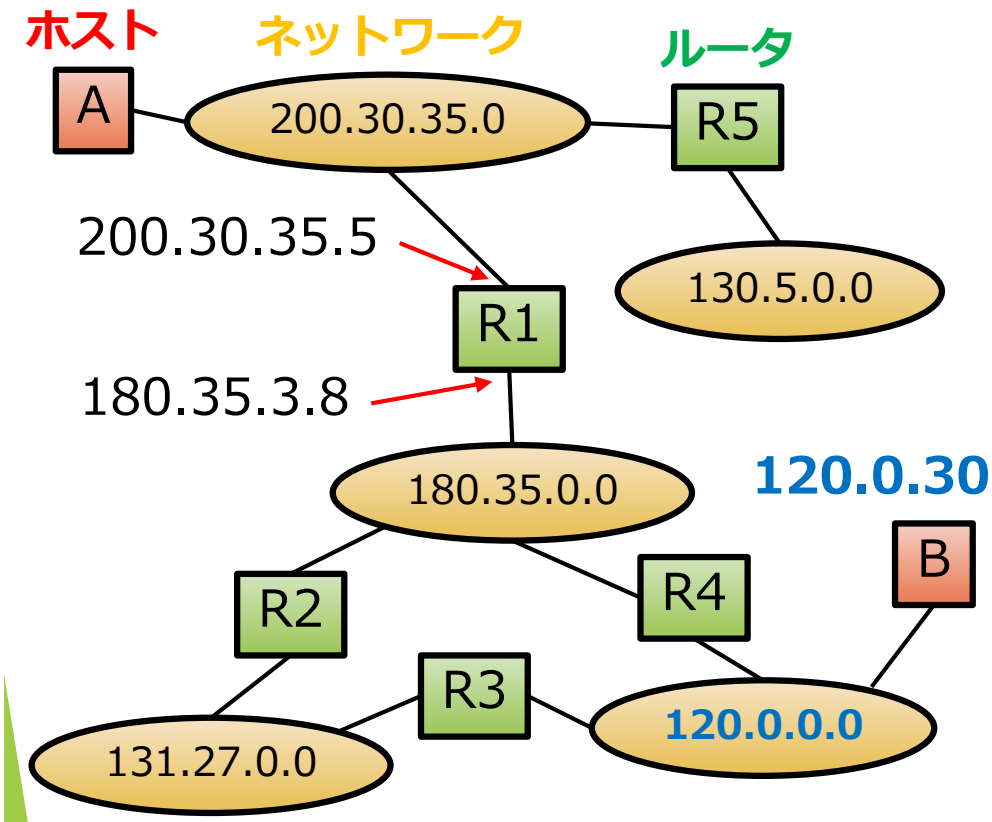
TCP/IP(2)

IPルーティング，ICMP, ARP

IPルーティングの概要

- **ルータ（中継器）**
  - IPデータグラムをIPアドレスによって適切な出線に送り出すノード
  - ネットワーク層のプロトコルに従って転送
  - ヘッダ内のIPアドレスを見て、IPデータグラムをどの経路を通して転送すべきかを判断するルーティング機能を持つ
- **ルーティングテーブル**
  - 中継のノード間でのルート決定に用いる表で、宛先ネットワークのアドレスと次の中継ノードのIPアドレスで構成

ルーティングの例



宛先ネットワーク	次のルータ	インターフェース	ホップ数
200.30.35.0	-	200.30.35.5	1
180.35.0.0	-	180.35.3.8	1
131.27.0.0	R2	180.35.3.8	2
120.0.0.0	R4	180.35.3.8	2
130.5.0.0	R5	200.30.35.5	2

ルータR1のルーティングテーブル

- R1がホストAからホストB(120.0.30.51)宛のデータグラムを受信
- R1はネットワーク番号(120.0.0.0)を見て、ルーティングテーブルの宛先ネットワーク番号のフィールドを参照
- 120.0.0.0のネットワークに配送するにはR4に送ればよいことが判明
- R4に送出するには、180.35.3.8側に出せばよいことが判明
- R4はデータグラムを受信すると、その宛先アドレスのネットワーク番号を見て、自分の管轄しているローカルネットワーク内のホスト宛であると判断

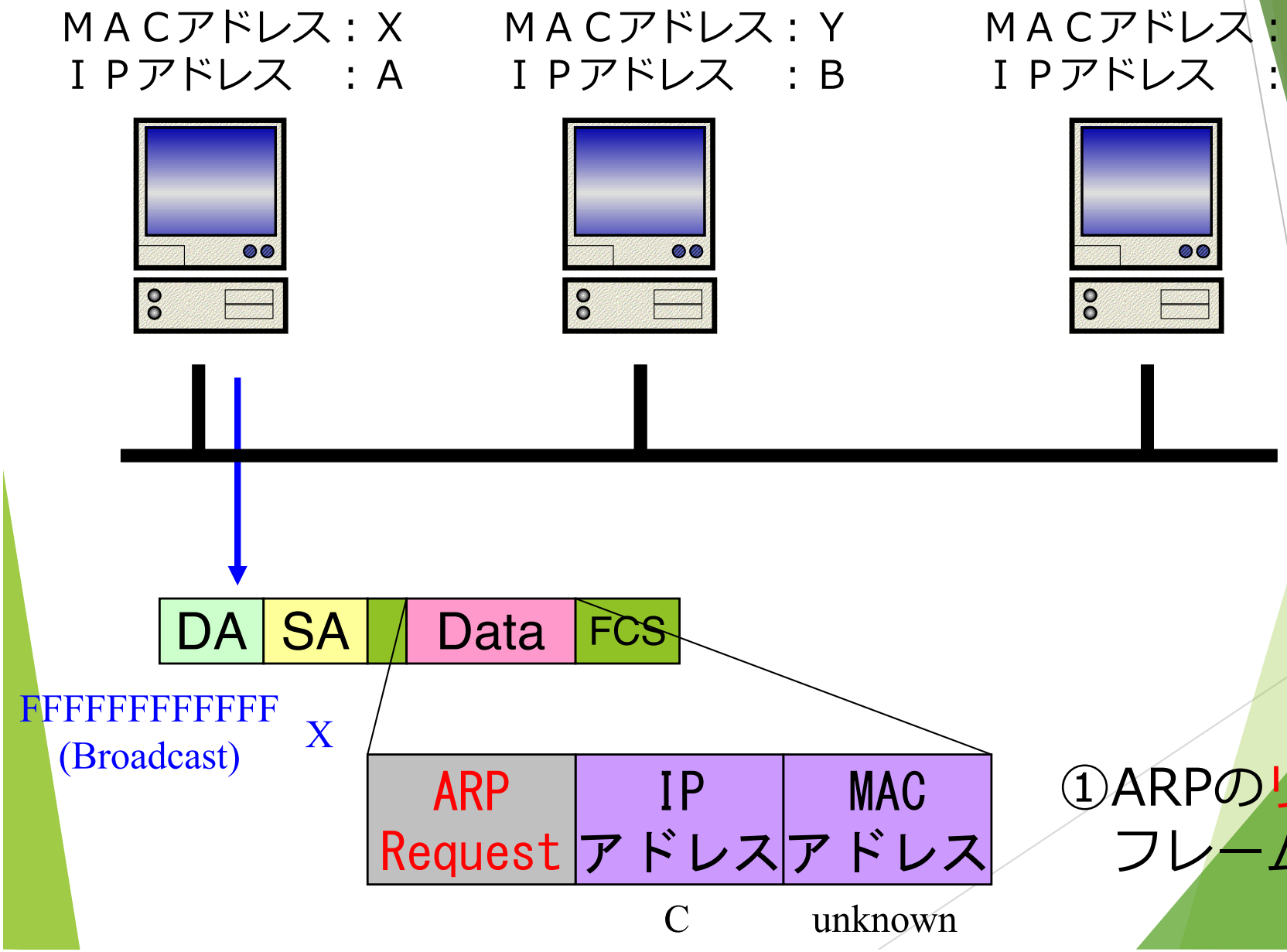
ICMP(Internet Control Message Protocol)

- ICMPの役割
  - IPは宛先ホストにデータグラムを届けるためのサービスを提供するだけ ➡ ICMPが動作の補佐を行う
  - 例えば・・・
    - データグラムに対する異常を知らせる
    - 中継網が混雑してきたとき、発信側の送信速度を緩める指示を通知
    - データグラムの宛先アドレスのホストが存在しない事を通知
    - ネットワークの状態を調べたり、相手ホストやルータの状態を調査ICMPの位置付け
- ICMPの位置づけ
  - ICMPはIPの上位に位置しており、
    IPを実装した場合にはICMPを必ず実装

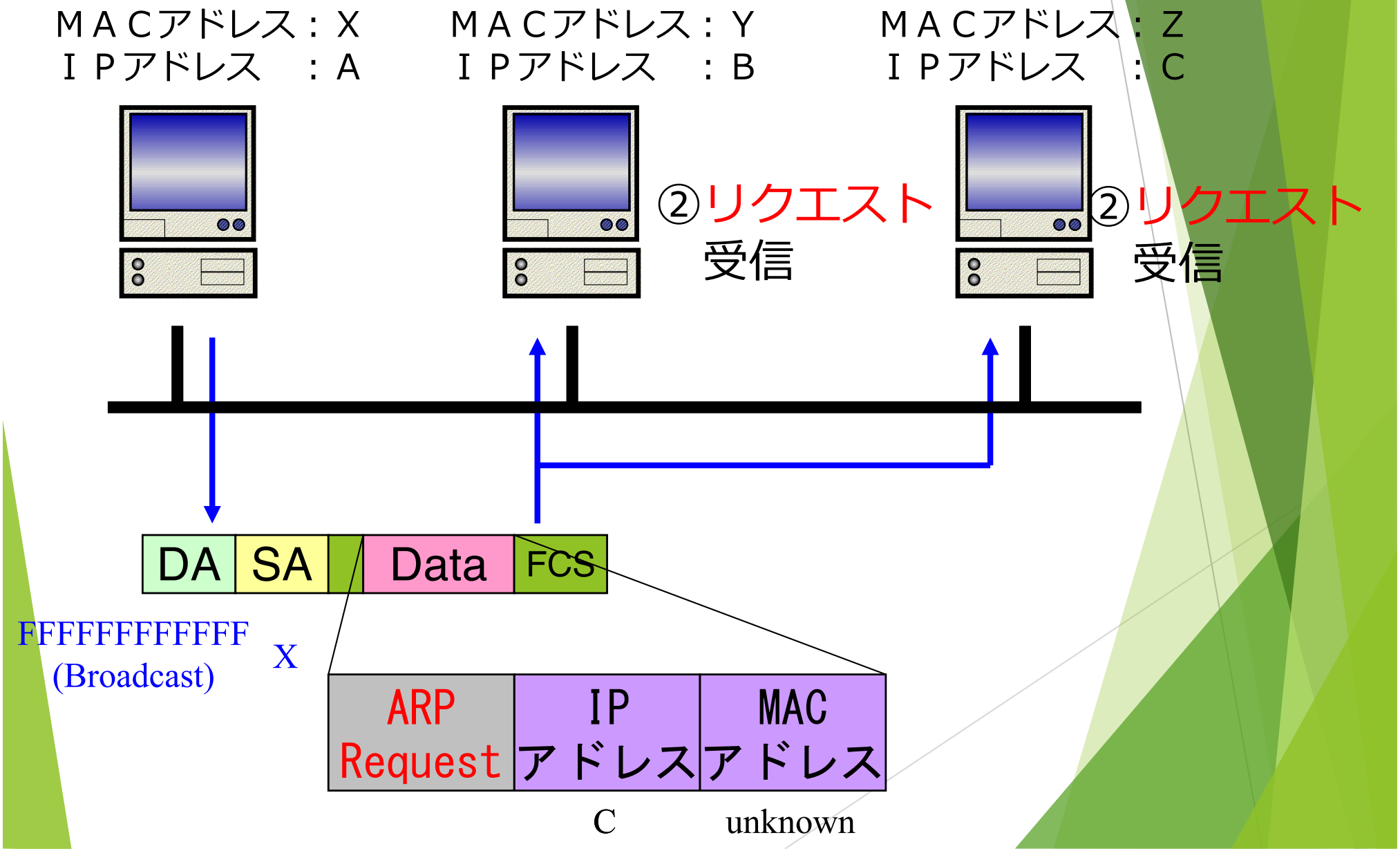
アドレス変換プロトコル

- **ARP**: Address Resolution Protocol
  アドレス解決プロトコル
  - 相手のIPアドレスから相手のMACアドレス（Ethernetアドレス等）を求める
- **RARP**: Reverse ARP
  逆アドレス解決プロトコル
  - EthernetアドレスからIPアドレスを求める
  - 計算機立ち上げ時に自分のIPアドレスを取得

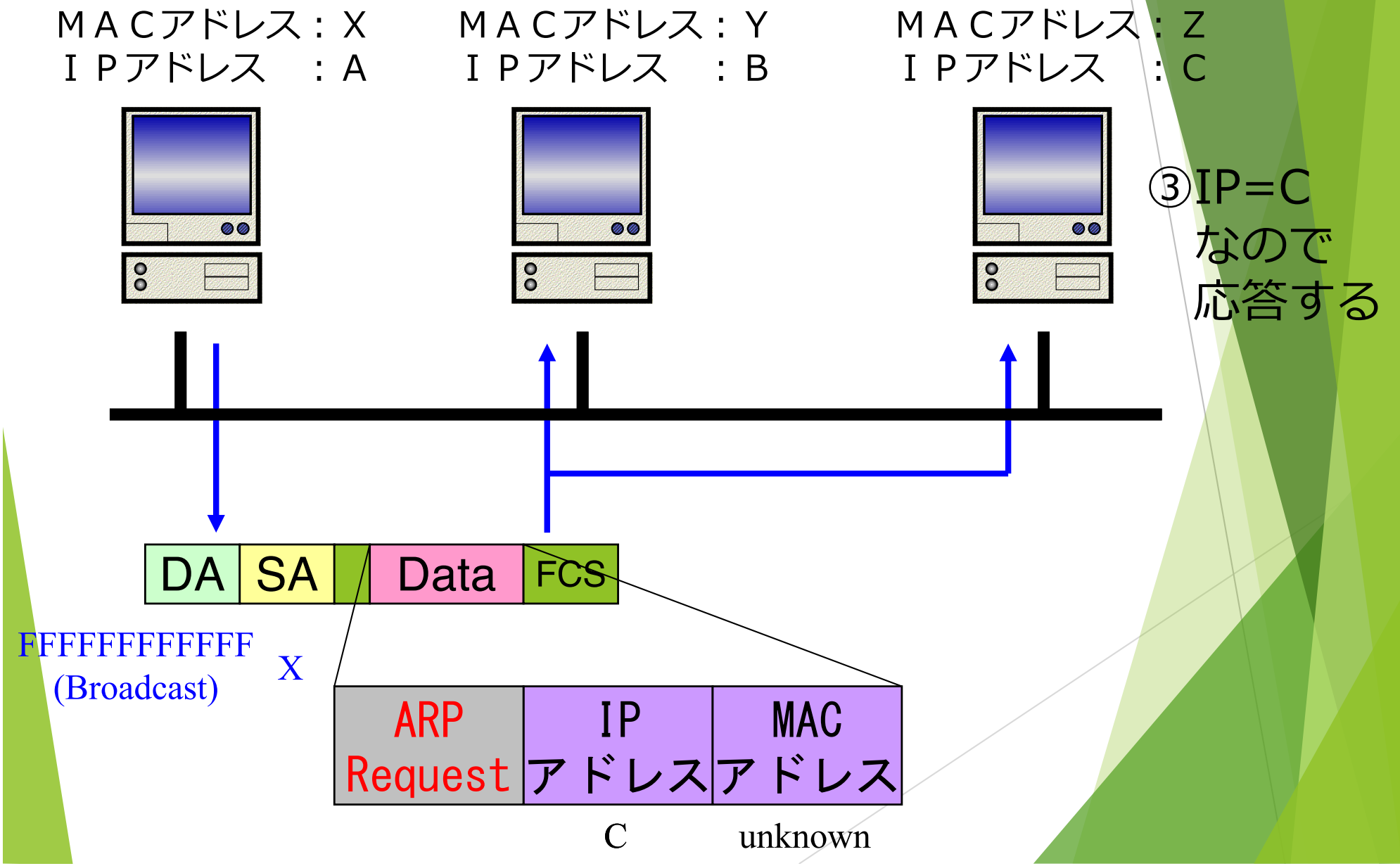
アドレス解決プロトコル(ARP)



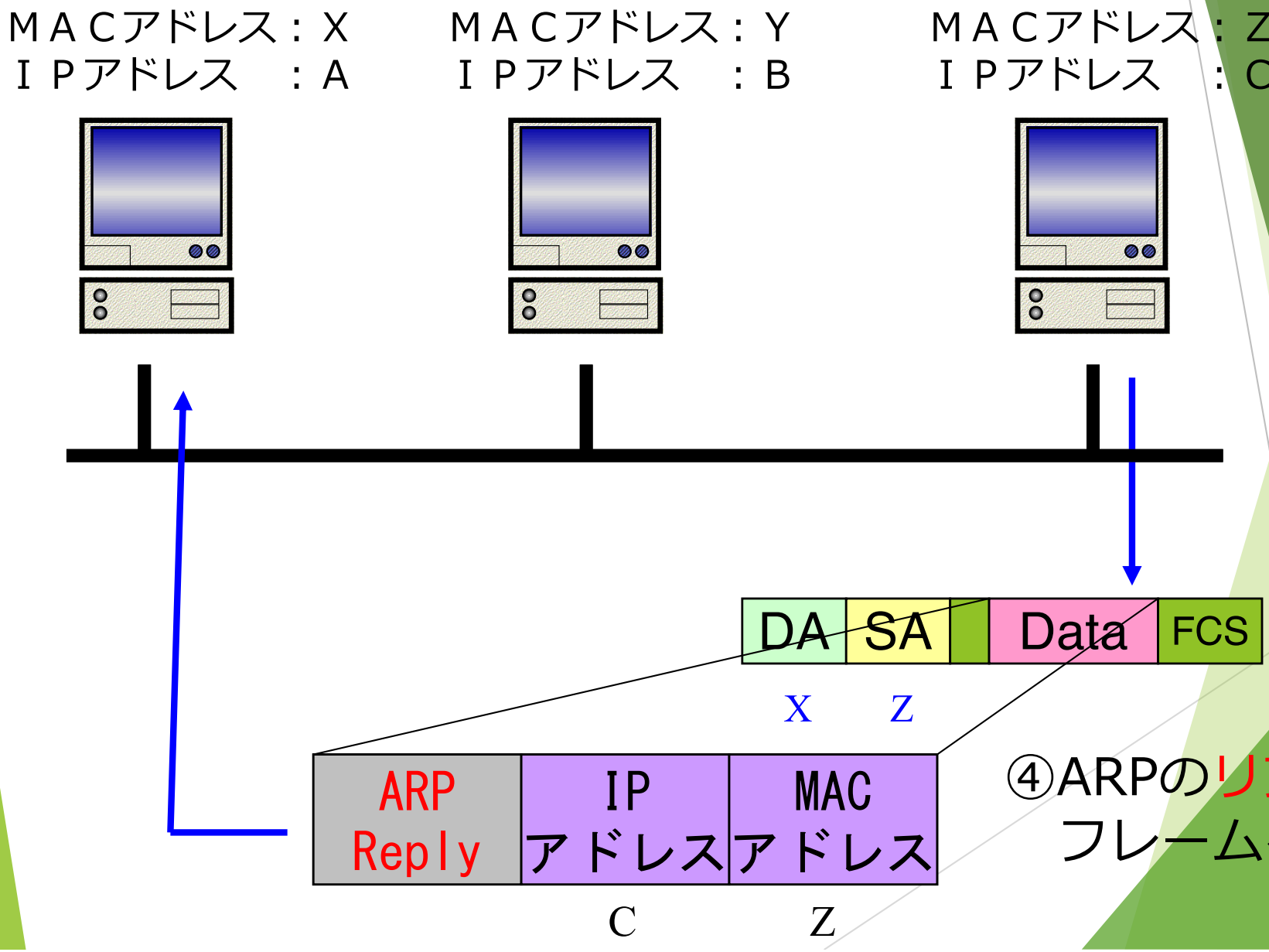
アドレス解決プロトコル(ARP)



アドレス解決プロトコル(ARP)



アドレス解決プロトコル(ARP)



TCP/IP(2)

IPv6

IPv6の背景

- 1990年代初頭
  - インターネットの急成長
    - IPv4アドレス空間枯渇問題
    - IPv4への追加機能に対する要求
- 1993年
  - IPng (Internet Protocol-Next Generation) 部会の発足
- 1994年
  - RFC 1752 “The Recommendation for the IP Next Generation Protocol”　発行

IPv6アドレス

- **128ビットのアドレス空間**
  （IPv4は32ビット＝約43億個）
  - 理論上最大で2^{128}個（v4の約8×10^{28}倍）
  - 地球上では，6.7個 / m^2
- 3つのアドレス形式
  - ユニキャスト
  - マルチキャスト
  - エニーキャスト
    - 複数のインタフェースに付与され，パケットはその内の１つにのみとどけられる（RFC 1546）
  - （IPv4ではユニキャスト，マルチキャスト，ブロードキャスト）

IPv6 Address Type Identification

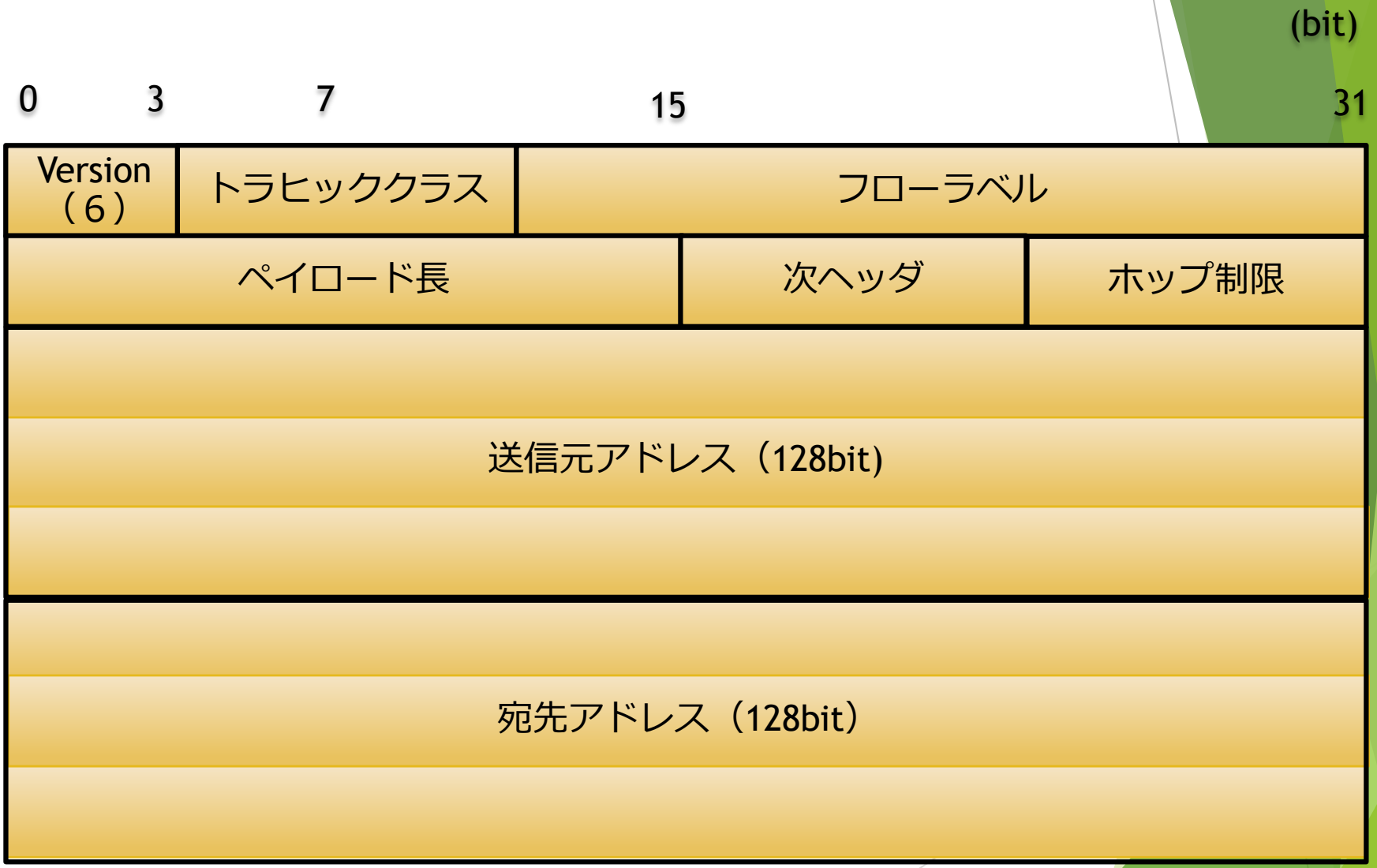
アドレスタイプ	プリフィックス（２進表記）	IPv6での記述
未定義	00…0 (128bit)	::/128
ループバック	00…1 (128bit)	::1/128
マルチキャスト	11111111	FF00::/8
リンクローカルユニキャスト	1111111010	FE80::/10
サイトローカルユニキャスト	1111111011	FEC0::/10
グローバルユニキャスト	その他全部	

注１）エニーキャストはグローバルユニキャストに含まれる
注２）現在IANAが割り当てているグローバルユニキャストは
001からはじまるもののみ

IPv6の特徴

- IPアドレスの拡大と経路制御表の集約
  - IPアドレスの構造をインターネットに適した階層構造に
  - IPアドレスを計画的に配布し、経路制御表を集約
- 性能の向上
  - ヘッダの構造を簡素化しルータの負荷を軽減
  - ルータの分割処理を禁止
- プラグ＆プレイ機能の必須化
  - IPアドレスの自動的な割り当て
- 認証機能や暗号化機能の採用
  - IPアドレスの偽造に対するセキュリティ機能
  - 盗聴防止機能

IPv6ヘッダフォーマット

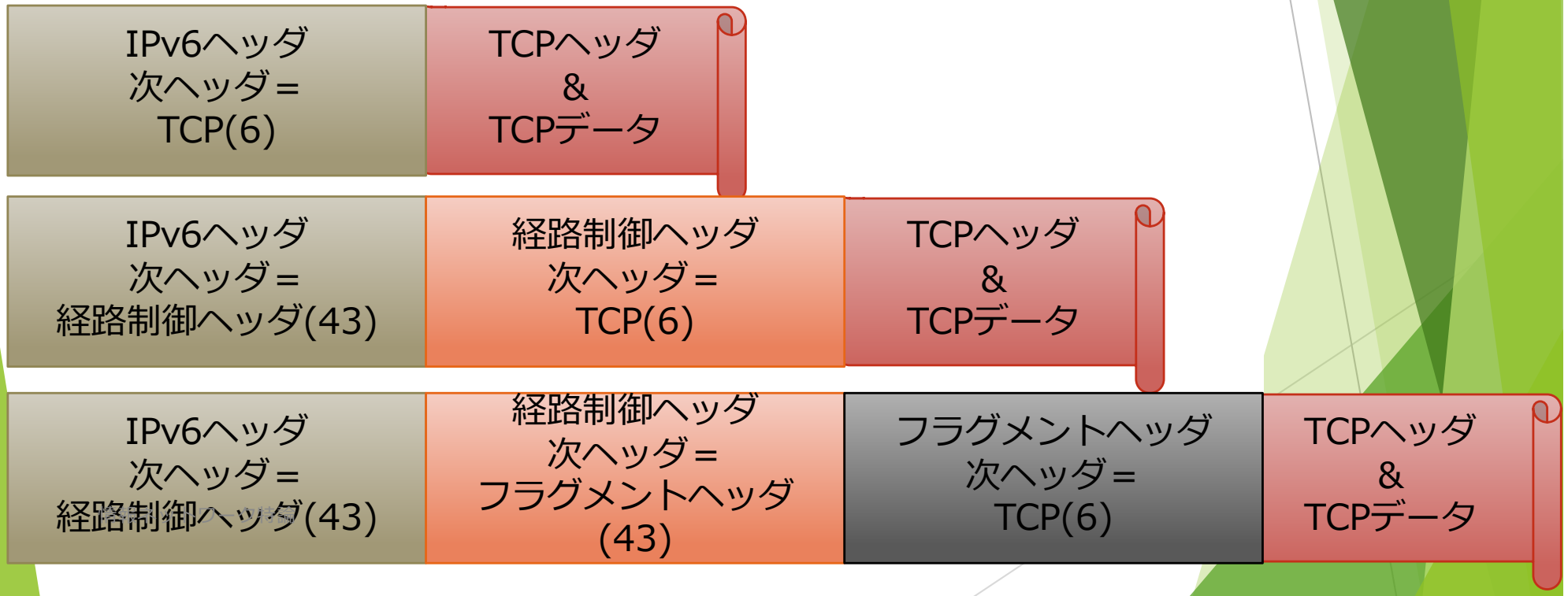


拡張ヘッダ

- オプションのための追加のヘッダ
  - ヘッダ長の固定（IPv4ヘッダは20～60バイト）
  - ヘッダの単純化，処理の高速化
  - ヘッダの拡張をする場合は，拡張ヘッダのみを定義すれば良い
- RFC 2460における拡張ヘッダ
  - ホップバイホップオプションヘッダ(RFC 2460)
  - 経路制御ヘッダ(RFC 2460)
  - フラグメントヘッダ(RFC 2460)
  - 宛先オプションヘッダ(RFC 2460)
  - 認証ヘッダ(RFC 4302)
  - カプセル化ペイロードセキュリティヘッダ(RFC 4835)

拡張ヘッダの例

- RFC 2460の例を参照



IPv6アドレス表記

- 128ビットを16ビットずつ8つに“：”で区切り16進数表記
  - 例（RFC 4291)
    - 2001:DB8:0:0:8:800:200C:
    - FF01:0:0:0:0:0:0:101
    - 0:0:0:0:0:0:0:1
    - 0:0:0:0:0:0:0:0
- 圧縮表記
  - 区切りが連続して０，先頭が０もしくは末尾が０は連続した“：”で省略可能
  - 連続“：”はアドレス内で１ヶ所のみ
  - 例（RFC 4291）
    - 2001:DB8::8:800:200C:
    - FF01::101
    - ::1
    - ::

※IPv4
2進数による表現
10000101.01000100.01100111.00000001
10進数による表現
133.68.103.2

TCP/IP(2)

TCP

TCP概要(1)

TCP（Transmission　Control　Protocol）

- ストリーム指向
  - 転送データ：**オクテット（バイト）に分割**されているビットストリーム（バイトストリーム）
  - 送信者が始点マシン上のストリーム配信サービスに渡したのと全く同じオクテット列を受信者に渡す
  - 非構造化ストリーム
- コネクション型
  - データ転送の前にコネクションを確立
  - （UDPは、コネクションレス型）

IP:　　ホスト間の通信，　IPアドレスによる識別
TCP:　**プロセス間**の通信，**ポート番号**による識別

TCP概要(2)

- **バッファ付き転送**
  - 十分なデータをストリームから集めて、適当な大きさのデータグラムにする（逆に大きなデータブロックは分割する）
  - データがバッファに埋まっていなくても転送されるために、プッシュ機構を提供

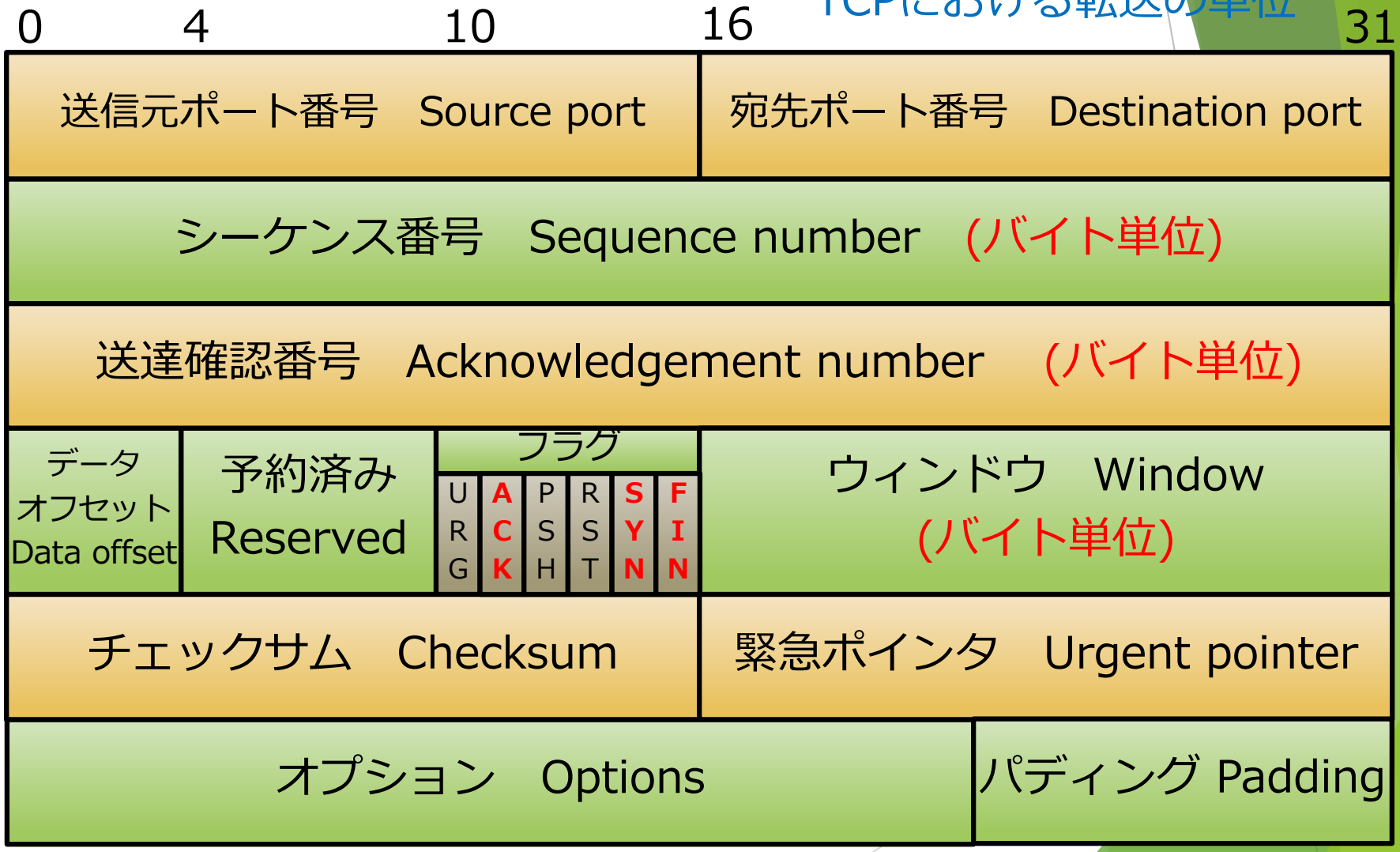
転送効率を良くし、ネットワークのトラヒックを最小化

- **全二重コネクション**
  - 全二重
    - 双方向の同時転送が可能
  - ピギィバック
    - 制御情報等を反対方向のデータを運ぶデータグラムに相乗せして返送する方式

TCPセグメントフォーマット

TCPヘッダフォーマット

セグメント：
TCPにおける転送の単位



信頼性の提供

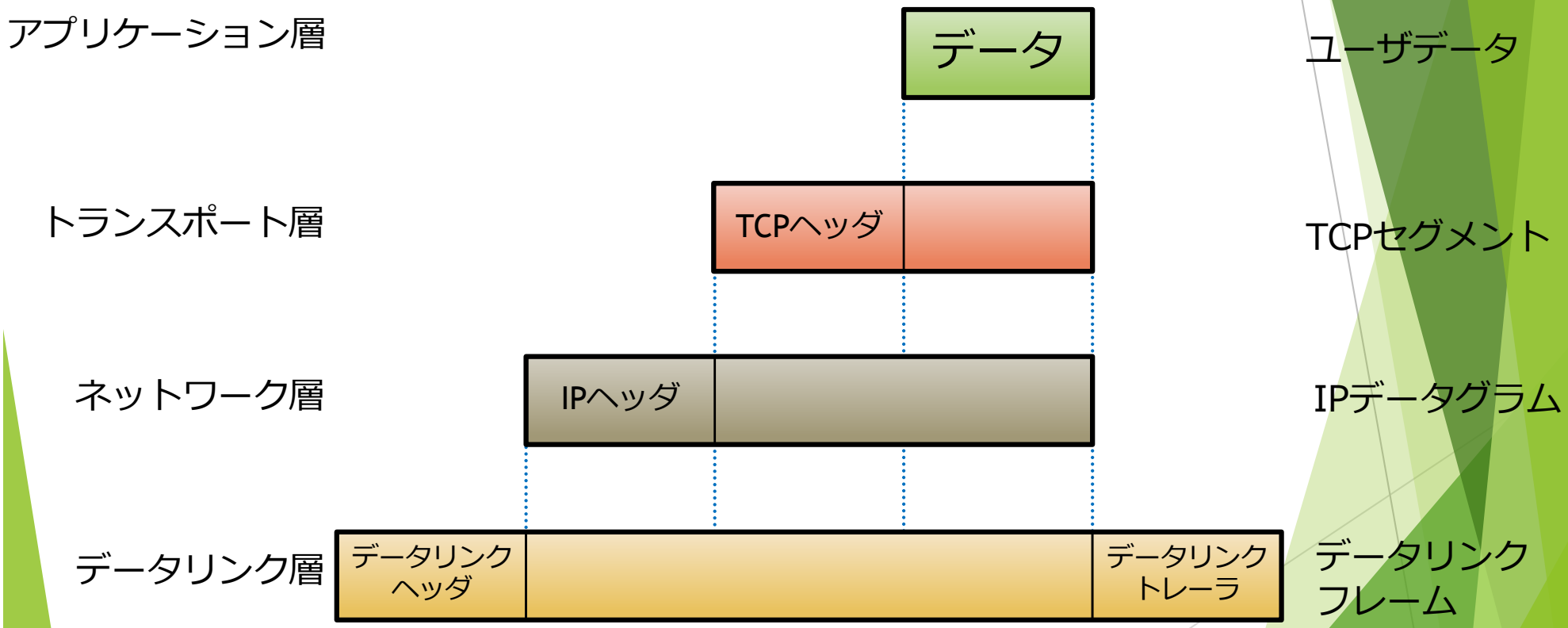
- **信頼性のあるストリーム転送サービス**
  - 転送されるデータのストリームを重複・喪失せずに配送することを保証

再送（go-back-N）を行う

Go-back-N でない再送もある

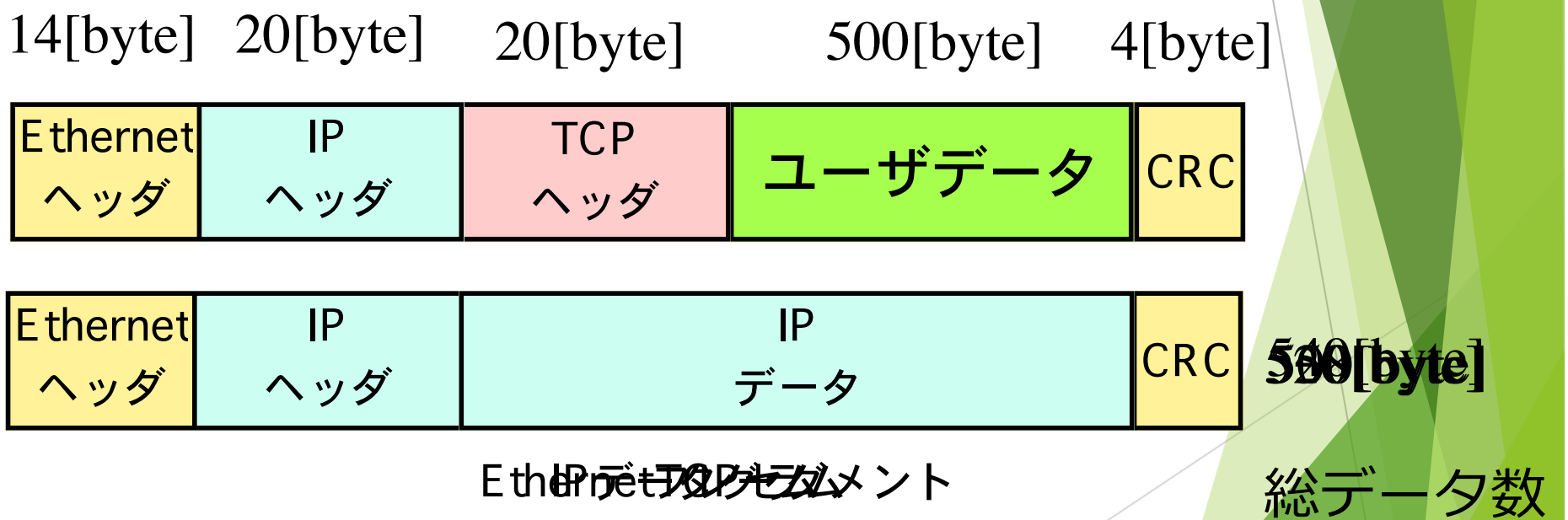
- **セグメント**
  - データストリームを伝送するためにセグメントに分割
  - 通常、各セグメントは１つのIPデータグラムとして転送
- **チェックサム**
  - 受信セグメントの誤りを検出するために付与される情報
  - 受信誤りを検出すると、そのセグメントを破棄
  - 16ビット単位で区切り1の補数で足し算、その結果の1の補数をとる
- **再送付き送達確認**
  - セグメントを受信するたびに送達確認（ACK）を返信
  - セグメント送信時にタイマーをスタートさせ、送達確認が到着する前にタイムアウトすればセグメントを再送

TCP/IPにおける転送単位



データのカプセル化

ユーザが500byteのデータをTCPで送信したい



カプセル化の例

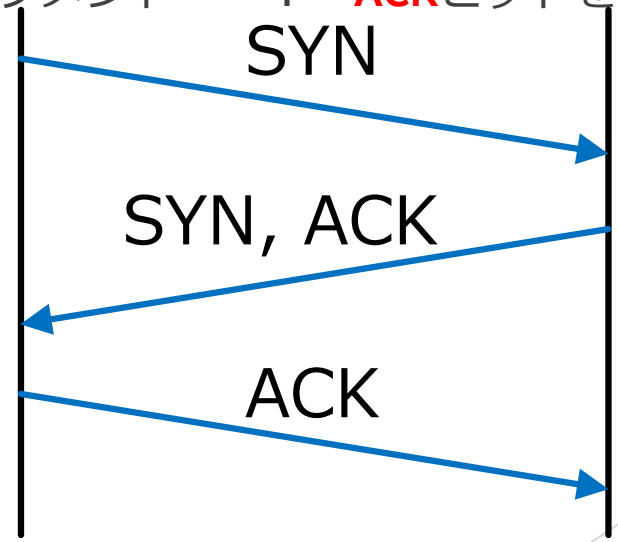
```
DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 290 arrived at  17:54:47.64758 ; frame size is 66 (0042 hex)
bytes.
DLC:  Destination = Station DECnet008CC7
DLC:  Source      = Station DECnet00FABA
DLC:  Ethertype  = 0800 (IP)
DLC:
IP:   ----- IP Header -----
IP:
IP:   Version = 4, header length = 20 bytes
IP:   Type of service = 00
IP:         000. .... = routine
IP:         ...0 .... = normal delay
IP:         .... 0... = normal throughput
IP:         .... .0.. = normal reliability
IP:   Total length    = 51 bytes
IP:   Identification  = 25097
IP:   Flags           = 0X
IP:         .0.. .... = may fragment
IP:         ..0. .... = last fragment
IP:   Fragment offset = 0 bytes
IP:   Time to live    = 14 seconds/hops
```

```
IP:   Protocol       = 6 (TCP)
IP:   Header checksum = 124A (correct)
IP:   Source address     = [128.52.46.32]
IP:   Destination address = [137.28.1.2]
IP:   No options
IP:
TCP:  ----- TCP header -----
TCP:
TCP:  Source port          = 23 (Telnet)
TCP:  Destination port     = 28264
TCP:  Sequence number      = 3654443534
TCP:  Acknowledgment number  = 45416891
TCP:  Data offset          = 20 bytes
TCP:  Flags                = 18
TCP:              ..0. .... = (No urgent pointer)
TCP:              ...1 .... = Acknowledgment
TCP:              .... 1... = Push
TCP:              .... .0.. = (No reset)
TCP:              .... ..0. = (No SYN)
TCP:              .... ...0 = (No FIN)
TCP:  Window               = 4096
TCP:  Checksum             = 411C (correct)
TCP:  No TCP options
TCP:  [11 byte(s) of data]
```

```
ADDR  HEX                                                  ASCII
0000  AA 00 04 00 8C C7 AA 00  04 00 FA BA 08 00 45 00  ..............E.
0010  00 33 62 09 00 00 0E 06  12 4A 80 34 2E 20 89 1C  .3b......J.4. ..
0020  01 02 00 17 6E 68 D9 D2  62 0E 02 B5 01 BB 50 18  ....nh..b.....P.
0030  10 00 41 1C 00 00 1B 5B  33 36 6D 3E 1B 5B 33 37  ..A....[36m>.[37
0040  6D 37                                            m7
```

コネクションの確立

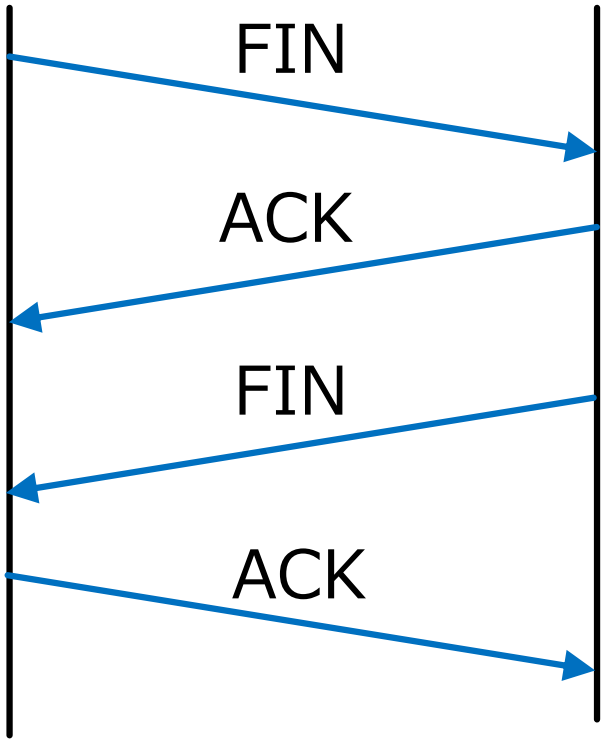
- **三方向ハンドシェイク(three-way handshake;** cf. two-army problem)
  - 信頼性の低い通信路で、コネクションの確立をより確実にする
  - 最初のシーケンス番号を一致させることができる
    - 最初のセグメント　：　**SYN**ビットをセット
    - ２番目のセグメント　：　**SYN**ビットと**ACK**ビットをセット
    - 最後のセグメント　：　**ACK**ビットをセット



コネクションの終了

- **緩やかな切断**(four-way handshake)
  - 両端の合意の下にコネクションを終了する

（徐々にコネクションを切断するという意味）



TCPコネクション確立（三方向ハンドシェイク）の観測例

```
1   0.00000 [192.9.200.85] [192.9.200.99] TCP D=139 S=257
    SYN SEQ=172249 LEN=0 WIN=1024
2   0.00281 [192.9.200.99] [192.9.200.85] TCP D=257 S=139
    SYN ACK=172250 SEQ=257461 LEN=0 WIN=2152
3   0.01602 [192.9.200.85] [192.9.200.99] TCP D=139 S=257
    ACK=257462 WIN=1024
4   0.00464 [192.9.200.85] [192.9.200.99] TCP D=139 S=257
    ACK=257462 SEQ=172250 LEN=78 WIN=1024
5   0.00417 [192.9.200.99] [192.9.200.85] TCP D=257 S=139
    ACK=172328 SEQ=257462 LEN=4 WIN=2074
6   0.02045 [192.9.200.85] [192.9.200.99] TCP D=139 S=257
    ACK=257466 WIN=1020
```

TCPコネクション終了（緩やかな切断）の観測例

```
41 0.00591 [192.9.200.99] [192.9.200.85] TCP D=257 S=139
           ACK=172759 WIN=2113
42 0.00570 [192.9.200.99] [192.9.200.85] TCP D=257 S=139
           ACK=172759 SEQ=258243 LEN=39 WIN=2113
43 0.01734 [192.9.200.85] [192.9.200.99] TCP D=139 S=257
           ACK=258282 WIN=892
44 0.01466 [192.9.200.85] [192.9.200.99] TCP D=139 S=257
           FIN ACK=258282 SEQ=172759 LEN=0 WIN=892
45 0.00294 [192.9.200.99] [192.9.200.85] TCP D=257 S=139
           FIN ACK=172760 SEQ=258282 LEN=0 WIN=2113
46 0.01166 [192.9.200.85] [192.9.200.99] TCP D=139 S=257
           ACK=258283 WIN=892
```