ソフトウェアアーキテクチャ
第10回　遠隔利用と電子メール

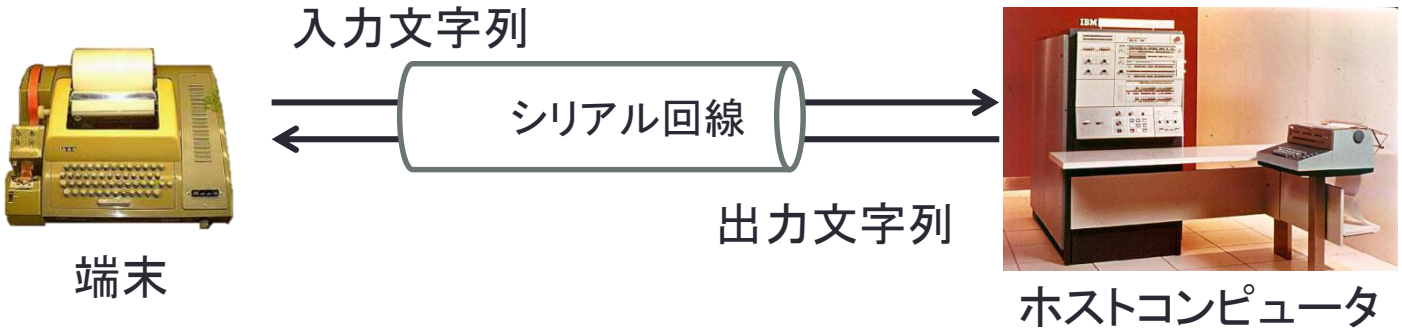
環境情報学部
萩野　達也

lecture URL

https://vu5.sfc.keio.ac.jp/slide/

端末によるコンピュータの利用

- パーソナルコンピュータ普及以前
  - シリアル回線や電話回線を利用して，端末（文字の入出力を行う装置）をホストコンピュータに接続
  - 端末から入力された文字はホストコンピュータに送られ，ホストコンピュータから送られてきた文字を端末で印刷（表示）する．
  - 文字端末，グラフィック端末など

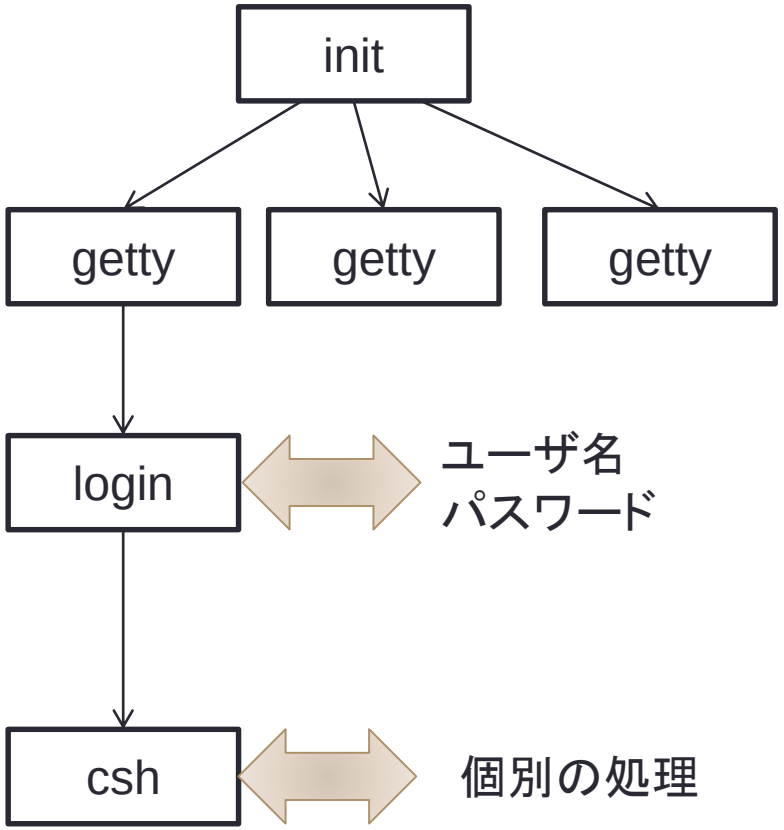


- パーソナルコンピュータ時代
  - PCの端末エミュレーションによる仮想端末
  - 接続はシリアル回線やモデムによる電話回線

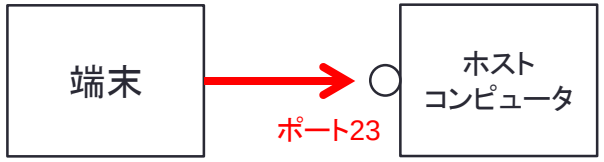
- インターネット時代
  - 仮想端末をインターネットを利用して遠隔コンピュータに接続

UNIXにおける端末接続

- ホストコンピュータはそれぞれの接続回線ごとにgettyを起動して，端末からの接続を待つ
  - getty = get tty
- loginにより，ユーザ名とパスワードによる認証を行う
- ユーザが指定したshellを起動し，端末からの入出力を渡す
  - 以後は端末はshellとの通信して，コマンドやプロセスを起動し処理を行う.



Telnetプロトコル

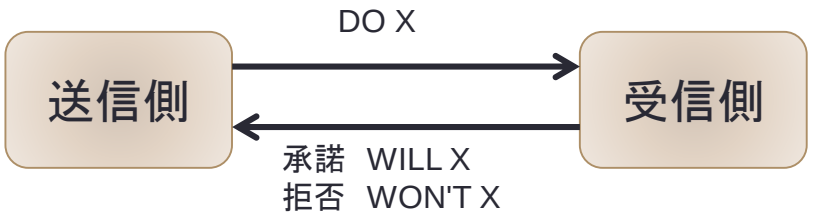


- インターネットを使った仮想端末の実現
  - シリアル回線などの代わりにインターネットを通信に使う
  - TCPコネクション（セッション，信頼性あり）
    - ポート23
- 非常に単純なプロトコル
  - 端末から入力された文字を遠隔のコンピュータに送る
  - 遠隔のホストコンピュータが出力した文字を端末に表示する
  - シリアル回線の代わりを行うだけで，認証などについてこれまでの仕組みを用いる
- 端末とホストの通信におけるいくつかのオプションの交渉も可能
  - DO と DON'T
  - WILL と WON'T

オプションの交渉

- 送信側でXを有効にしたい
  - 送信側 WILL X
  - 受信側 DO X または DON'T X

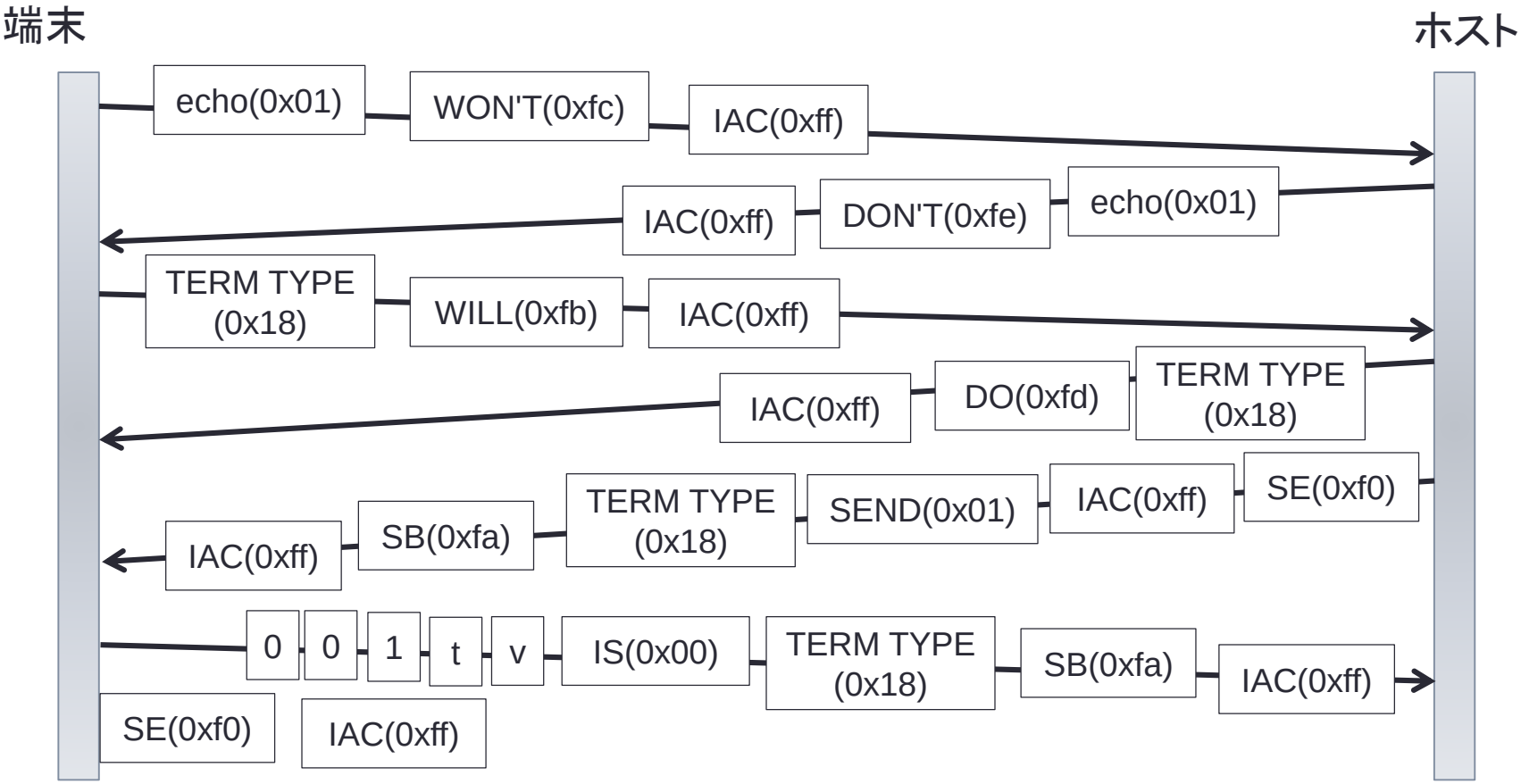
- 受信側でXを有効にてほしい
  - 送信側 DO X
  - 受信側 WILL X または WON'T X



- 送信側でXを無効にしたい
  - 送信側 WON'T X
  - 受信側 DON'T X

- 受信側でXを無効にしてほしい
  - 送信側 DON'T X
  - 受信側 WON'T X

オプション交渉例



IAC: Interpret As Command
SB～SE: Sub negotiation

Telnetの制御機能

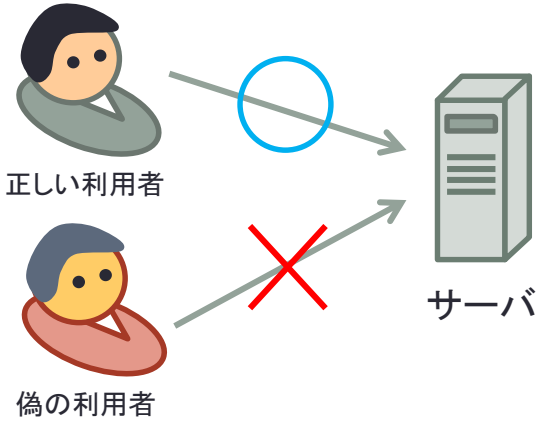
- Are You There
  - IAC, 0xf6
- 文字消去
  - IAC, 0xf7
- ライン消去
  - IAC, 0xf8
- プロセス中断 (ctrl-C)
  - IAC, 0xf4
- 出力中止
  - IAC, 0xf5
- 同期 （TCP緊急データ）
  - IAC, 0xf2

Telnetの発展系

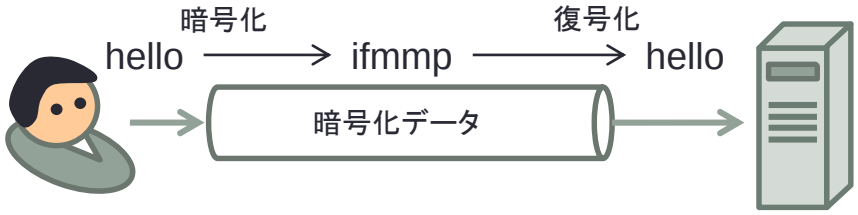
- Telnetの問題点
  - ユーザ名・パスワードが平文で流れるため，セキュリティ上問題がある
- Rlogin (remote login)
  - UNIXにおいて使いやすいように設計
  - BSD 4において導入
  - 信頼するコンピュータを `~/.rhosts` に書いておけば，パスワードなしに利用可能
  - セキュリティの面で使われなくなりつつある
- SSH (Secure Shell)
  - protocol 1, 2
  - 認証方法: password, challenge-response, RSA, DSA
- リモートデスクトップ
  - Windowsにおけるリモートターミナル

ユーザ認証と暗号化とハッシュ

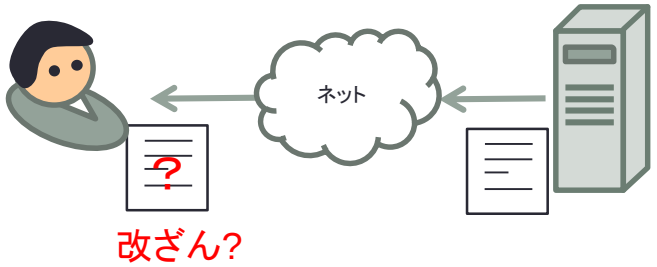
- ユーザ認証
  - 利用者が正しい利用者であることを確認する
  - パスワード認証
  - ワンタイムパスワード
  - チャレンジ／レスポンス認証
  - RSA（Rivest Shamir Adleman）認証
  - DSA（Digital Signature Algorithm）認証



- 暗号化
  - 通信の中身が他人に分からないようにする
  - 共通鍵暗号
    - 鍵を他人に知られてはいけない
  - 公開鍵暗号
    - 公開鍵と秘密鍵のペアを用いる



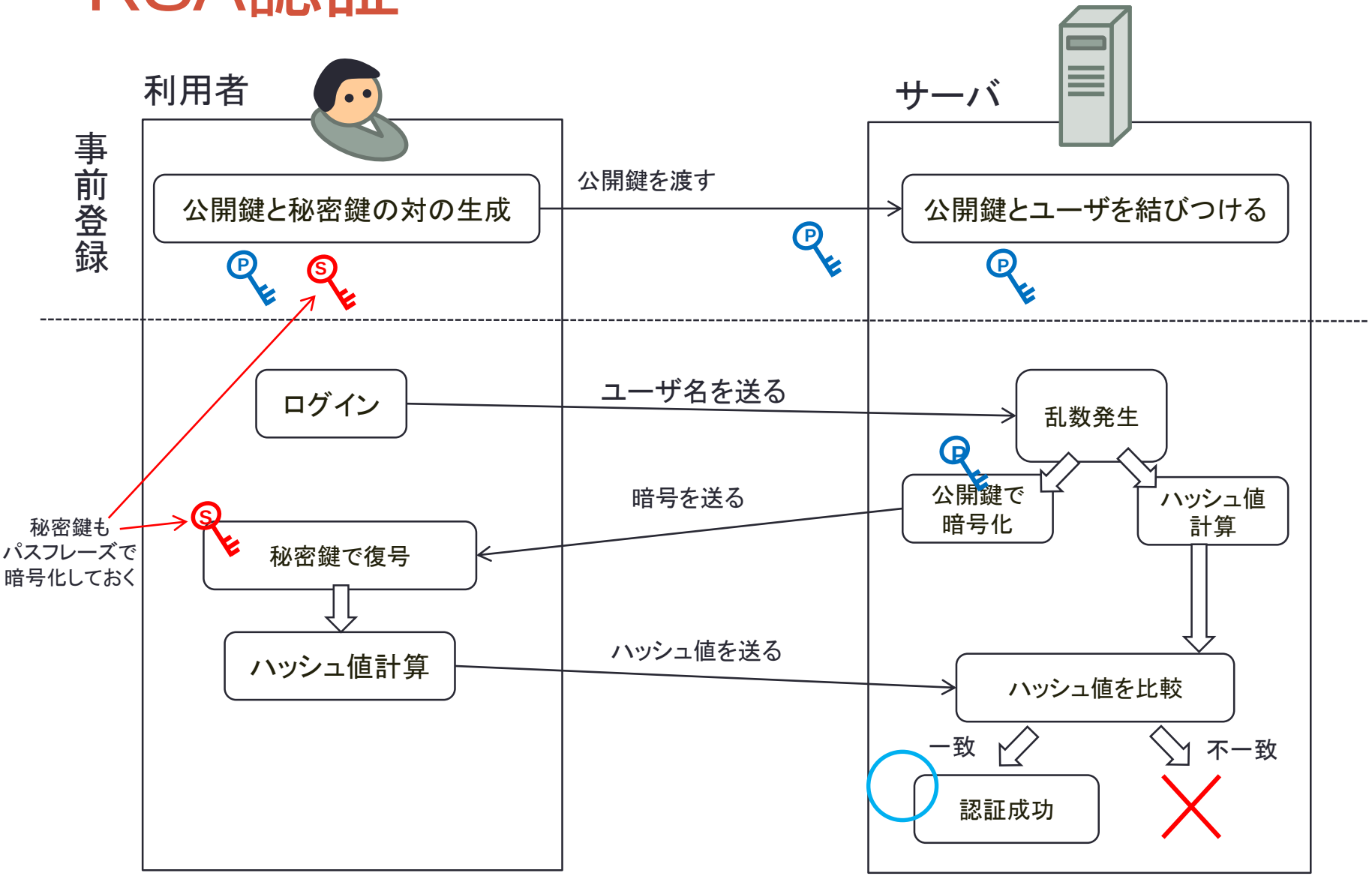
- ハッシュ
  - データが改ざんされていないことを確認する
  - チェックサム
  - 暗号学的ハッシュ関数(MD5，SHA-1, SHA-2)



ユーザ認証

- パスワード認証
  - 前もってサーバに登録されたパスワードを送る
- ワンタイムパスワード
  - ワンタイムパスワード発生器を利用し一度だけのパスワードを生成する
  - 例：現在の時刻から計算した数字
- チャレンジ／レスポンス認証
  - サーバから送られてきた乱数（チャレンジ）を受け取る
  - クライエントは受け取ったチャレンジとパスワードから数字を計算する
  - 計算した数字（レスポンス）をサーバに送り返す
  - サーバでも同じ計算を行いレスポンスと一致するか調べる
- RSA認証
  - RSA（Rivest Shamir Adleman）公開鍵暗号を利用する
  - 公開鍵と秘密鍵のペアを生成する
  - 公開鍵をサーバに登録する
  - サーバは乱数を公開鍵で暗号化しクライアントに送る
  - クライアントは秘密鍵を使って復号化し，ハッシュを計算する
  - ハッシュをサーバに送り返す
  - サーバはハッシュが正しいかどうか検査する

RSA認証

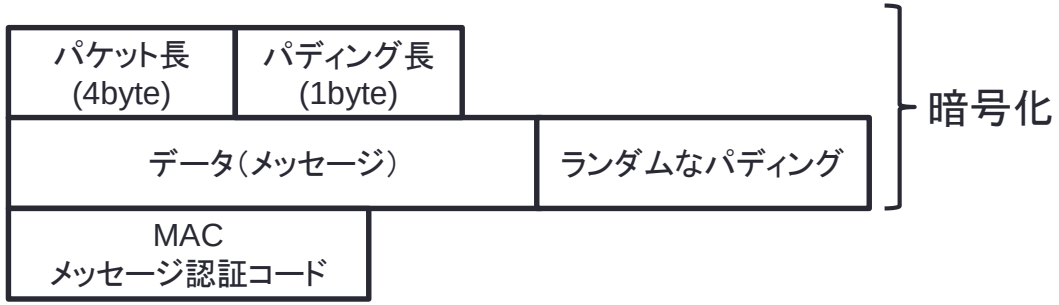

利用者
サーバ
事前登録
公開鍵と秘密鍵の対の生成
公開鍵を渡す
公開鍵とユーザを結びつける
ログイン
ユーザ名を送る
乱数発生
暗号を送る
公開鍵で暗号化
ハッシュ値計算
秘密鍵も
パスフレーズで
暗号化しておく
秘密鍵で復号
ハッシュ値計算
ハッシュ値を送る
ハッシュ値を比較
一致
不一致
認証成功

SSH概要

- 暗号や認証を利用して，端末とホストが安全に通信を行うプロトコル
  - ポート22
- 通信データ
  - クライアントとサーバ間はメッセージをやり取りする
  - メッセージはパケットに入れられて暗号化される
  - 複数のチャネルを作りコマンドを実行

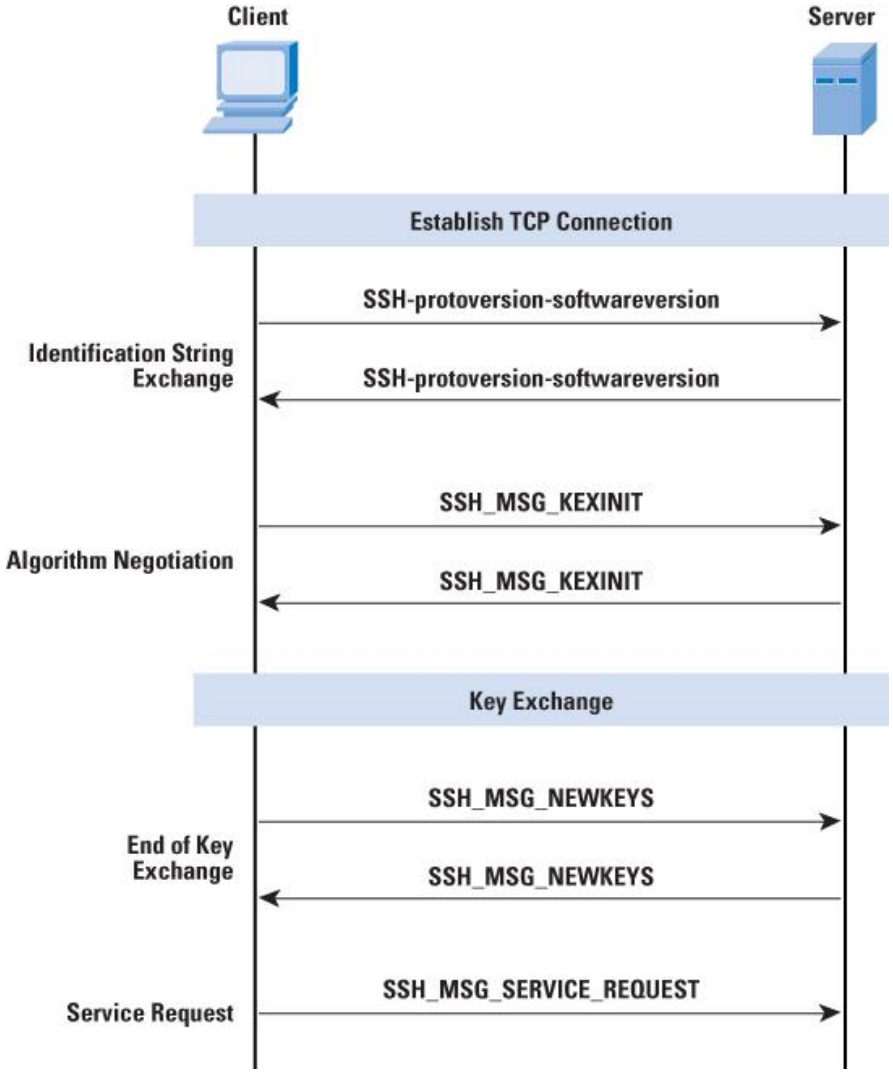
メッセージ

パケット



Message ID	Value
SSH_MSG_DISCONNECT	1
SSH_MSG_IGNORE	2
SSH_MSG_UNIMPLEMENTED	3
SSH_MSG_DEBUG	4
SSH_MSG_SERVICE_REQUEST	5
SSH_MSG_SERVICE_ACCEPT	6
SSH_MSG_KEXINIT	20
SSH_MSG_NEWKEYS	21
SSH_MSG_USERAUTH_REQUEST	50
SSH_MSG_USERAUTH_FAILURE	51
SSH_MSG_USERAUTH_SUCCESS	52
SSH_MSG_USERAUTH_BANNER	53
SSH_MSG_GLOBAL_REQUEST	80
SSH_MSG_REQUEST_SUCCESS	81
SSH_MSG_REQUEST_FAILURE	82
SSH_MSG_CHANNEL_OPEN	90
SSH_MSG_CHANNEL_OPEN_CONFIRMATION	91
SSH_MSG_CHANNEL_OPEN_FAILURE	92
SSH_MSG_CHANNEL_WINDOW_ADJUST	93
SSH_MSG_CHANNEL_DATA	94
SSH_MSG_CHANNEL_EXTENDED_DATA	95
SSH_MSG_CHANNEL_EOF	96
SSH_MSG_CHANNEL_CLOSE	97
SSH_MSG_CHANNEL_REQUEST	98
SSH_MSG_CHANNEL_SUCCESS	99
SSH_MSG_CHANNEL_FAILURE	100

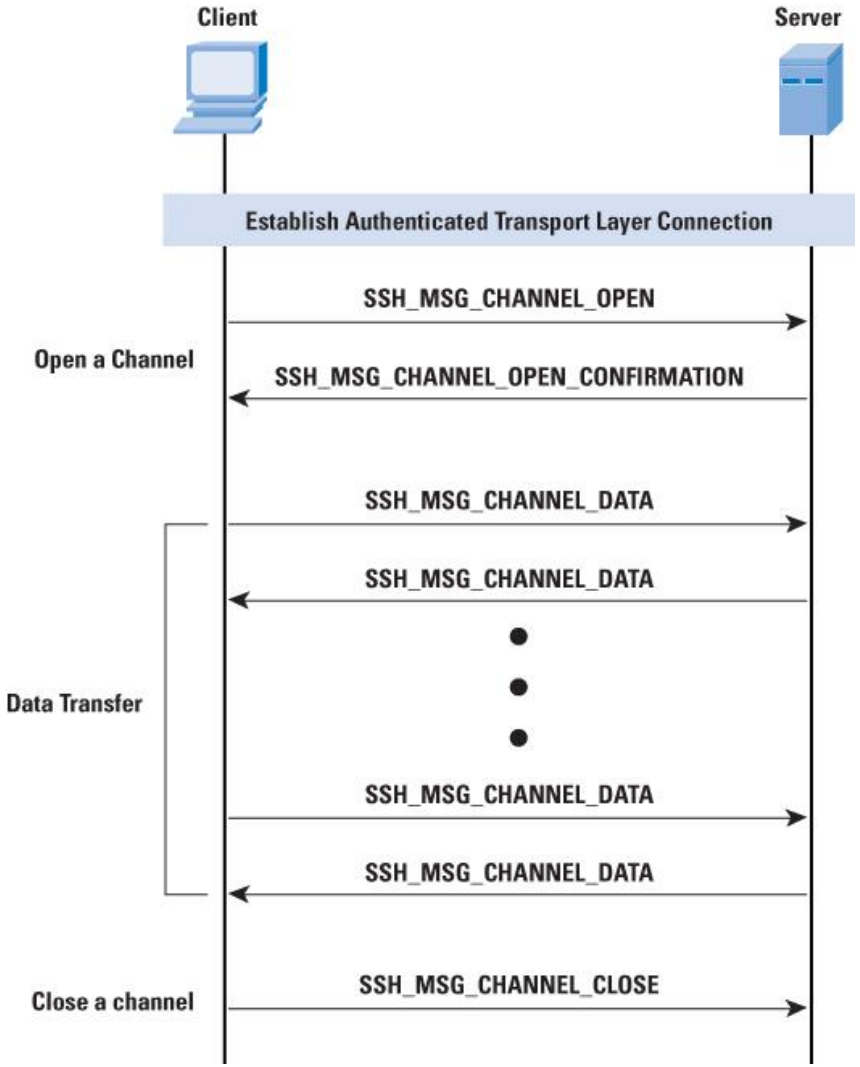
SSH動作シーケンス図（1）



- 接続すると安全な接続を設定するために鍵を交換する

http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-46/124-ssh.html

SSH SSH動作シーケンス図（2）



- チャネルごとに別々のデータのやり取りが可能
  - シェルの実行
  - コマンドの実行
  - ポート転送
  - X11転送

http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-46/124-ssh.html

遠隔コンピュータ利用のまとめ

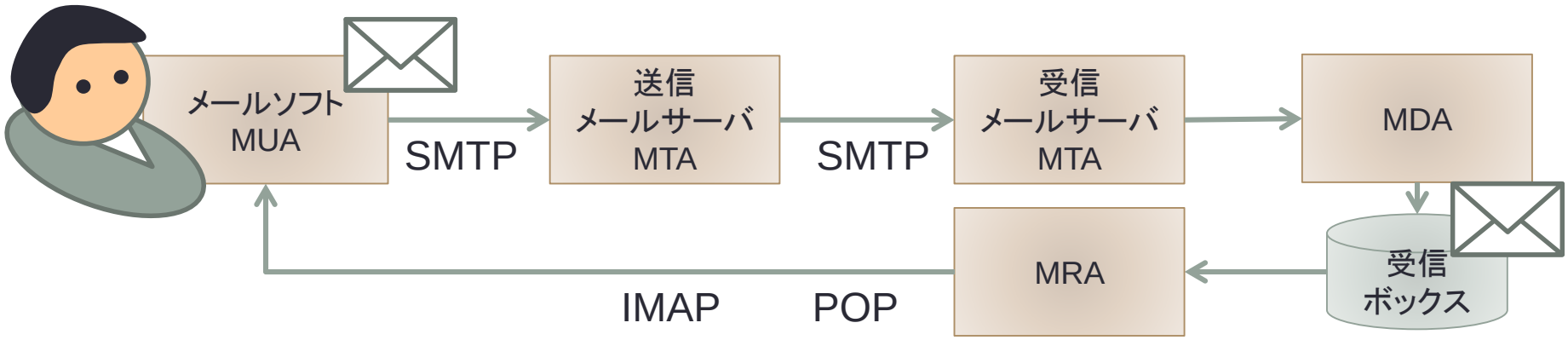
- Telnet
  - もっとも古いTCPプロトコルの一つ
  - 単純に仮想端末をインターネット上で実装したもの
  - セキュリティの問題あり

- SSH
  - 暗号化および認証をもつ端末プロトコル
  - 複数のユーザ認証をサポート
  - １つの接続で複数のチャネルをサポート

電子メール

電子メールの構成要素

- MUA (Mail User Agent)
  - メールクライアント
  - メールの送受信を行う
- MTA (Mail Transfer Agent)
  - メールを宛先に送るサーバ
- MDA (Mail Deliver Agent)
  - MTAがメールボックスに書き込むソフトウェア
- MRA (Mail Retrieval Agent)
  - リモートのメールボックスからメールを取り込むサービス



SMTP

- Simple Mail Transfer Protocol
  - MUAがMTAにメールを送るときに利用
  - MTAが他のMTAにメールを転送するときに利用

- 仕様
  - RFC821（1982年）が最初
  - 各種拡張機能が追加
  - ESMTP (Extended SMTP)

- TCPコネクション
  - ポート25

SMTPサーバ必須コマンド

- HELO
  - 送信ホスト名を渡す
- MAIL
  - メールメッセージの送信者を知らせる
- RCPT
  - 受信者を知らせる
- DATA
  - メールメッセージの内容を送る
- RSET
  - サーバの状態をリセットする
- NOOP
  - 何もしない
- QUIT
  - 終了する

メールの送信例

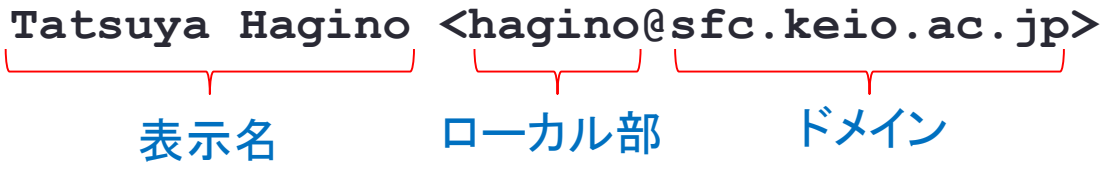
- サーバに接続
  - → 220 smtp.sfc.keio.ac.jp SMPT
- HELO ninna.tom.sfc.keio.ac.jp
  - → 250 ninna.tom.sfc.keio.ac.jp Hello
- MAIL FROM: hagino@sfc.keio.ac.jp
  - → 250 hagino@sfc.keio.ac.jp Sender ok
- RCPT TO: ns@gms.komazawa-u.ac.jp
  - → 250 ns@gsm.komazawa-u.ac.jp Recipient ok
- RCPT TO: timbl@www.org
  - → 220 timbl@www.org No such user
- RCPT TO: timbl@w3.org
  - → 250 timbl@w3.org Recipient ok
- DATA
  - → 354 Enter mail, end with "." on a line by itself

From: hagino@sfc.keio.ac.jp
To: ns@gms.komazawa-u.ac.jp
Cc: timbl@w3.org
Subject: Hello

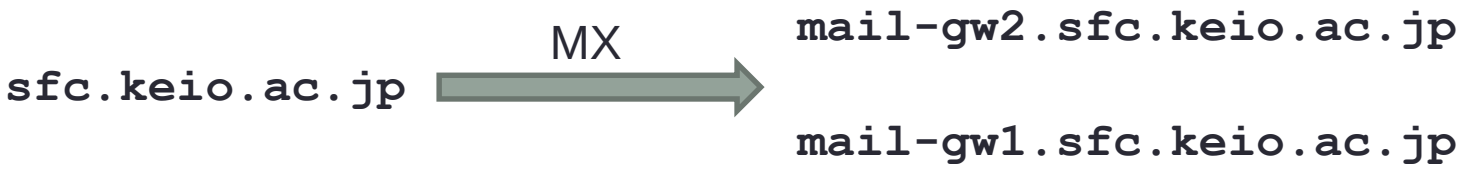
Dear Nobuo and Tim,
(メール本文)
.

- → 250 0AA06460 Message accepted for delivery
- QUIT
  - → 221 smtp.sfc.keio.ac.jp closing connection

メールアドレスとメールサーバ

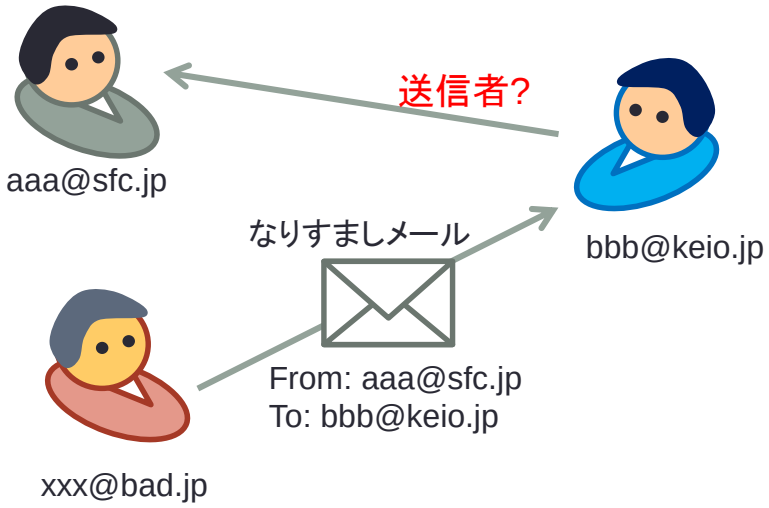


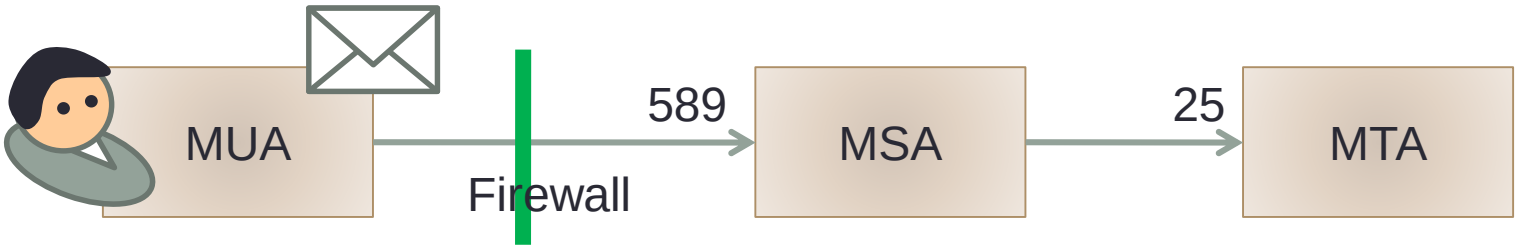
- メールアドレスはローカル部とドメインからなる
  - ドメインはDNSのドメイン，大文字と小文字の区別はない
  - ローカル部は本来は大文字小文字の区別があるが，無視するシステムが多い
- メールサーバ
  - ドメイン名から対応するメールサーバを探す必要がある
  - DNSのMX（Mail Exchange）でドメインのメールサーバを指定する



SMTPのセキュリティ

- 誰でもがメールを送ることができる
  - なりすましメールが可能
  - Spamができる
- MUAを限定する
  - POP before SMTP
  - SMTP AUTH
- 外部MTAの利用を制限
  - 外部の25番ポートへの通信を制限
  - MSA (Message Submission Agent)
- メール内容を暗号化
  - SMTP over SSL





メールサーバソフトウェア

- sendmail
  - 1980年代にBerkeleyで作成
  - UUCPなどSMTP以外の電子メールプロトコルにも対応
  - sendmail.cfに変換規則を書くことにより設定
- qmail
  - sendmailとは異なり，高速でシンプルかつ堅牢な構造
  - 設定が簡素
  - メールボックスはMaildir形式がデフォルト
- postfix
  - sendmailとの操作上の互換性を確保
- courier-MTA
  - オープンソース
  - qmailの後継
- exim

メールメッセージのフォーマット

- ヘッダ
  - RFC822
    - **`To:`** あて先
    - **`From:`** 発信者
    - **`Date:`** 日付
    - **`Subject:`** サブジェクト

- 本文
  - MIME （Multipurpose Internet Mail Extensions）
    - **`MIME-Version: 1.0`**
    - **`Content-Type: type/subtype; parameter`**
    - **`Content-Transfer-Encoding: mechanism`**
    - **`Content-ID: message-id`**
    - **`Content-Description: text`**

Content-Type

- **text/plain;charset=us-ascii**
  - 通常のテキスト
- **text/enriched**
  - He is a <bold>Japanese</bold>
- **multipart/mixed;boundary="-3D3C5DF17D08"**
  - 複数のデータからなる
- **message/rfc822**
  - 電子メール
- **application/octec-stream;name=text.lzh;type=lzh**
- **application/postscript**
- **image/jpeg**
- **audio/basic**
- **video/mpeg**

電子メールでの日本語の取り扱い

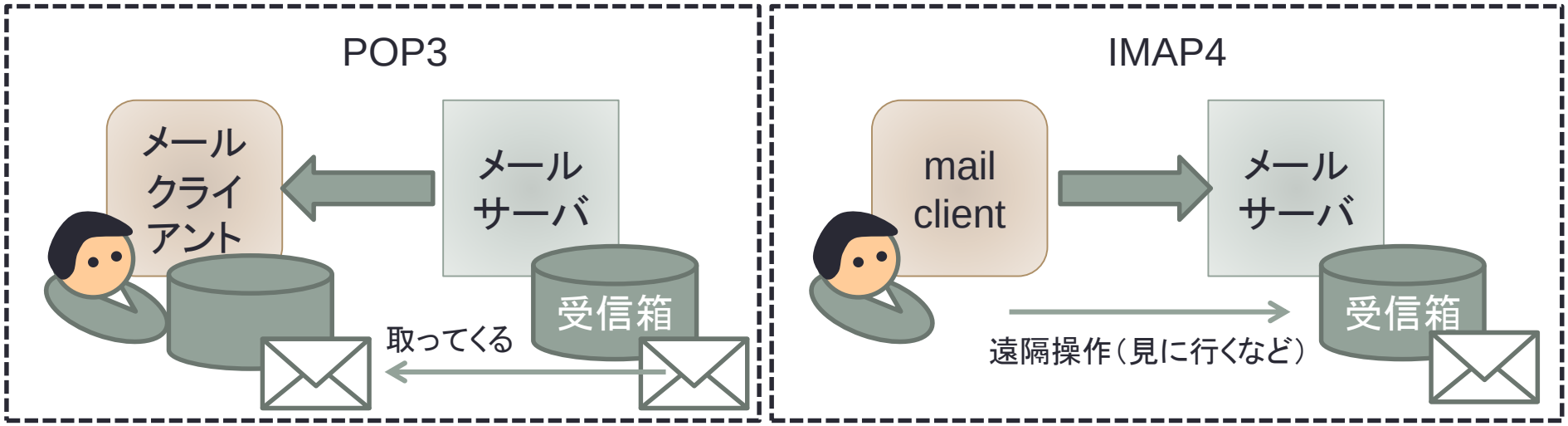
- 本文
  - デフォールトはiso-2022-jp
  - MIMEによりエンコーディングを指定可能

- メッセージヘッダ
  - ASCIIのみ使用可能
  - Subjectに日本語を入れたいときにはASCIIにエンコードする
    - base64
      - `Subject: ?ISO-2022-JP?B?GyRCMGY4fU1NGyhC?=`
    - quoted-printable
      - `Subject: ?ISO-88591?Q?Keld_J=F8rn_Simonsen?=`

電子メールの受信

- メールボックスを直接見る
- リモートなメールボックスからメールを取ってきて見る

- POP3
  - Post Office Protocol
  - メールをMUAに取り込むプロトコル
  - クライアントがメールを持つ
  - メールをサーバに残しておき他のMUAと共有することも可能
- IMAP4
  - Internet Message Access Protocol
  - メールボックスを操作するプロトコル
  - MUAはメールのキャッシュを持つだけ
  - 複数のMUAと共有可能
  - クライアントはキャッシュを持つだけ



POPの接続例

```
        +OK Qpopper at mail.sfc.keio.ac.jp starting.
USER hagino
        +OK Password required for hagino.
PASS password
        +OK Welcome hagino!
STAT
        +OK 3 1230
RETR 1
        (the first mail)
DELE 1
        +OK Message 1 marked for deletion
QUIT
        +OK 1 message expunged. Bye!
```

IMAPの接続例

```
      +OK IMAP4rev1 server ready
A121 CAPABILITY
      * CAPABILITY IMAP4rev1 AUTH=X509
      A121 OK CAPABILITY completed
A123 LOGIN hagino password
      A123 OK LOGIN completed
A125 LIST ~/Mail/%
      * LIST (¥Marked) "/" ~/Mail/Inbox
      * LIST () "/" ~/Mail/Stuff
      A125 OK LIST completed
A127 DELETE ~/Mail/Stuff
      A127 OK DELETE Completed
A129 SELECT ~/Mail/Inbox
      * 23 EXISTS
      * 12 RECENT
      * OK [UNSEEN 3] Messages 3 is first unseen
      * OK [UIDVALIDITY 5732875] UISs valid
      * FLAGS (¥Answered ¥Flagged ¥Deleted ¥Seen ¥Draft)
      A129 OK [READ-WRITE] SELECT completed
A131 FETCH 3 BODY[TEXT]
      * FETCH (Body[TEXT]{62}
      (メール本文)
      )
      A131 OK FETCH completed
A133 LOGOUT
      * BYE IMAP4rev1 Server logging out
      A133 OK LOGOUT completed
```

電子メールその他話題

- メールの転送
  - 届いたメールを別のアドレスに転送する
  - 通常のUNIXでは `~/.forward` に記述（MDAが処理）
  - システム全体のものは `/etc/aliases` に記述（MTAが処理）
- メーリングリスト（ML）
  - 複数人に同時にメールを送る
  - 宛先のリストを登録したメールアドレスを作っておく
- メールの振り分け
  - 届いたメールに対してプログラムを起動し，中身に応じて処理を行う
  - 自動フォルダ振り分け
  - 部分的な転送
  - SPAM振り分け
  - 不在の自動返信
  - MDAとしてprocmailなどが有名
- メールの暗号化
  - 受信者しか読むことができないようにメール本文を暗号化する
  - ヘッダ部分は暗号化されない
  - 例：受信者の公開鍵で暗号化する
- メールの電子署名
  - 送信者が正しいことを保障する
  - メールの中身が改ざんされていないことを調べる
  - 例：送信者の秘密鍵でメッセージダイジェストを暗号化する

電子メールのまとめ

- もっともよく使われているインターネットプロトコルの一つ
  - 複数のRFCで規定
- メールクライアント（MUA）の設定
  - 電子メールの送信と受信ではプロトコルが異なる
  - SMTP
  - POP/IMAP
- セキュリティ上の問題も多い
  - SPAM