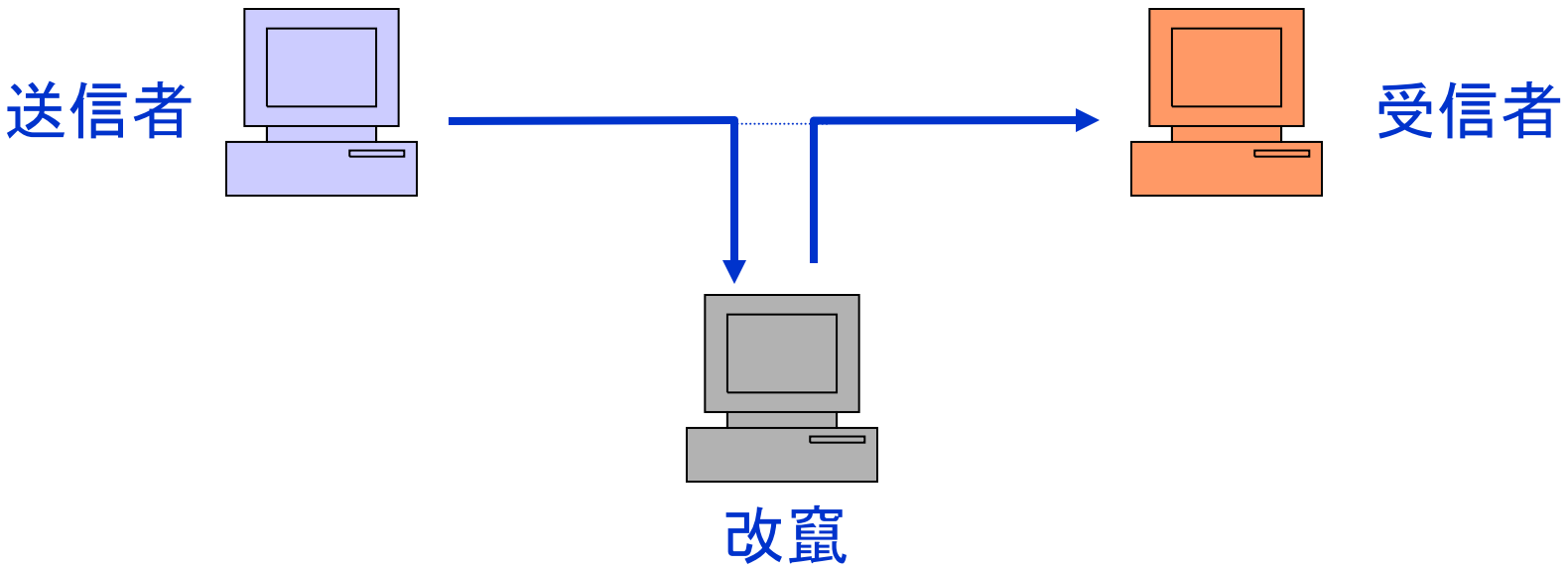
データの改竄を防ぐ仕組み

2002/9/12

データ改竄

ファイルや通信データの中身を第三者が不当に書き換える事

正当な受信者以外の第三者が不正にネットワークにアクセスし、送信者のメッセージの内容を改変して受信者に送付



改竄への対策

1.通信路のセキュリティの強化

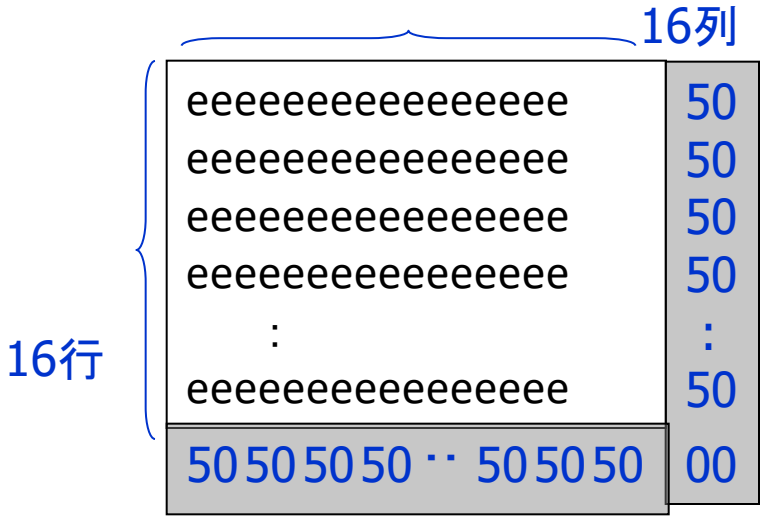
2.データ改竄の検知

- チェックサム
- ハッシュ関数

チェックサム

- 誤り発見用の数値(もしくはその集まり)
  - データを適当なブロックに分割し、それぞれの文字コードを合計

例：データを16バイトずつ16行のマトリックス状に並べ、各行文字コードの和の最下位1バイトを右端に、各列文字コードの和の最下位1バイトを下端に記入してチェックサムを作成

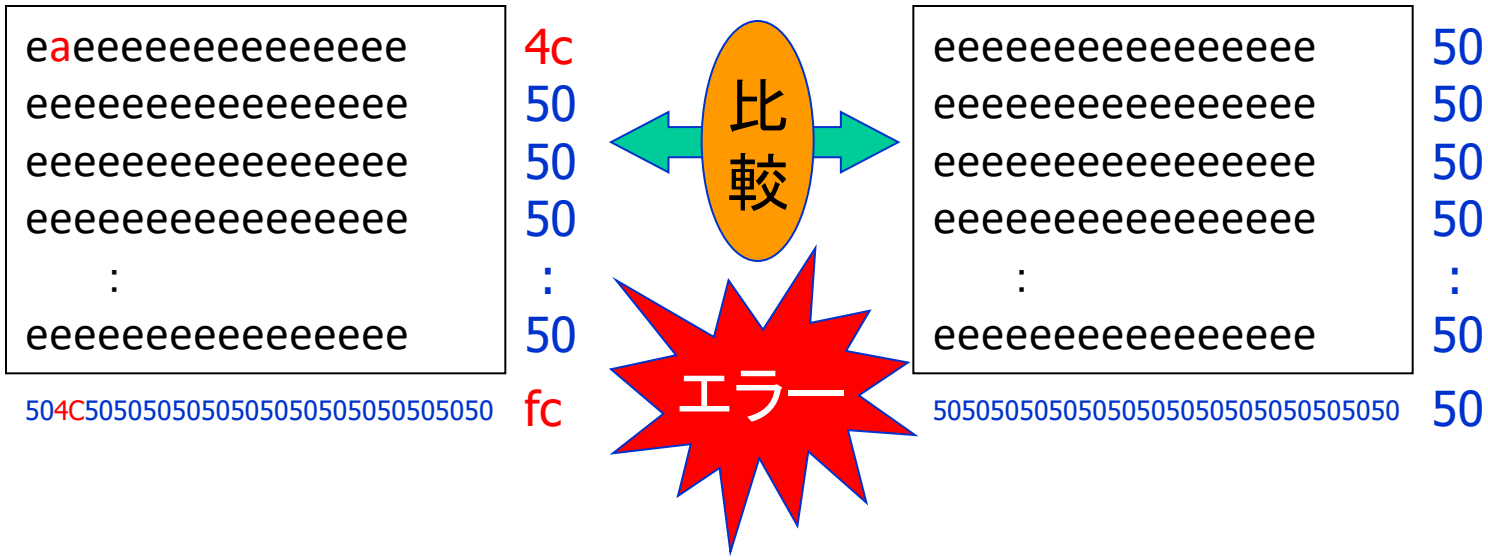


256個の“e”（ASCII文字コードで0x65）の文字により構成されるデータのチェックサム

チェックサムによるデータ改竄の検知

- あらかじめ送信データのチェックサムの値を送っておく

1行2列目のデータが“a”(ASCII文字コードで0x61)に文字化け



チェックサムの穴

- 巧妙なデータ操作により、チェックサムをすり抜ける事が可能

ハッシュ関数

- 任意長のメッセージを一定長のデータ(ハッシュ値)に変換する関数
- 衝突回避性：MとM′が異なるならh(M)≠h(M′)
- 一方向性：h(M)からMの復元が不可能

→メッセージ改竄の検出が可能
MD5(Message Digest 5)
SHA(Secure Hash Algorithm)

改竄検知(MD5)

MD5(Message Digest 5)とは

- 1991年に提案された
  - RFC1321で公開
    - RFC(Request For Comment)
      - TCP/IPコミュニティにおける各種標準仕様を規定するドキュメント類の総称
- 一方向ハッシュ関数
  - メッセージから固定長(128bit)のハッシュ値を返す
  - ハッシュ値から元のメッセージを求めることは不可能
  - 同じハッシュ値を返すメッセージを生成することは困難
    - １２８ビットハッシュ関数の出力は2の１２８乗、１０進数だと３９桁となりとても大きい数となるため

```
% md5 -s AA
MD5 (“AA”) = 3b98e2dffc6cb06a89dcb0d5c60a0206
```

MD5の応用(1)

- ファイルの改竄を検知
  - ファイルのMD5を保存しておけば
    MD5の値が変わっていれば、ファイルが
    変更されたことがわかる
- FreeBSDのportsでも使用
  - 例えば、
    % cat /usr/ports/shells/tcsh/distinfo

    MD5 (tcsh-6.10.tar.gz) = f459c423074d85dfaa55439eb908a053

  - ファイルが改竄されていないかを確認
  - ホームページにファイルのMD5の出力値を
    記述してある場合もある

MD5の応用(2)

- PEM(Privacy Enhanced Mail)
  - インターネットセキュリティーを施した電子メールの規格
  - RFC1421-1424で公開
- PGP (Pretty Good Privacy)
  - 暗号化電子メールプログラムの一つ
  - 電子署名、メッセージの暗号化／復号の機能
- APOP
  - 時間(に関する値)、サーバ名、パスワードからなる文字列にMD5を適用
  - MD5の出力値を比較して認証

今日の実習(1)

- 文字列のMD5の出力を調べる
  - md5 –s “aaaa”
  - md5 –s “aaab”
  - md5 –s “aaa”
    など
- 全てのMD5の出力値がまったく違うことを確認

今日の実習(2)

1. /usr/ports/xxxx/yyyy/…（例. /usr/ports/x11-wm/fvwm2-i18n）で make fetch を行い、ファイルをダウンロード

2. このときダウンロードされたファイルは、/usr/ports/distfile に保存されています

3. ダウンロードしたファイルのMD5の出力値と、usr/ports/xxxx/yyyy/…/distinfo に記載されている値が一致するかどうかを調べる