第9回　保護とセキュリティ（2）

ネットワークセキュリティと暗号，認証

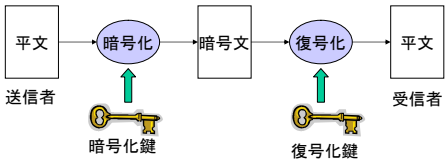
ネットワークセキュリティの必要性

- 不正アクセス
  - 利用権限を越えてﾈｯﾄﾜｰｸ経由でｼｽﾃﾑの利用を図る
  - 情報の窃盗、改ざん、ｼｽﾃﾑの破壊
    - ﾊﾟｽﾜｰﾄﾞの管理を厳重に行う必要がある
    - ssh（ｾｷｭｱｼｪﾙ）：暗号を用いた安全な本人認証
- （意図的に作られた悪質な）不正プログラム
  - ｳｲﾙｽ：感染、潜伏、発症機能をもつ不正ﾌﾟﾛｸﾞﾗﾑ
  - ﾌｧｲﾙなどに自身を付着させ（感染）、条件が揃うまで動かず（潜伏）、その後悪質な振る舞いをする（発症）
    - OSの保護機能、バグによる脆弱性の対処
    - 怪しいソフトは受け取らない、開かない。
- 注：悪意を持った不正行為を行う者をｸﾗｯｶｰという。ﾊｯｶｰは、ｺﾝﾋﾟｭｰﾀ技術に長け、その技術を生産（善意）的なことに利用する者を指す。但し、日本のﾏｽｺﾐでは、ｸﾗｯｶｰのことを「ﾊｯｶｰ」と言っている。

暗号（cryptography）

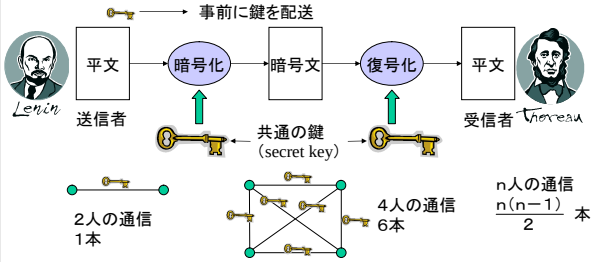
- 当事者以外にわからないﾃﾞｰﾀに変換。暗号化により以下が実現
  - 秘匿通信：盗聴されても（ﾃﾞｰﾀが盗まれる）意味がわからない
  - 認証：本人であること、ﾃﾞｰﾀが本人のものであることを確認
  - 完全性の保障：ﾃﾞｰﾀが改ざんされていないことを受信者が確認

平文（ひらぶん）（元のﾒｯｾｰｼﾞ）を暗号化鍵をﾊﾟﾗﾒｰﾀとする関数により、暗号文に変換
上記の変換を暗号化、変換手順を暗号化アルゴリズムという。
受信者は、復号化鍵を使って、暗号文を平文に戻す（復号化）。



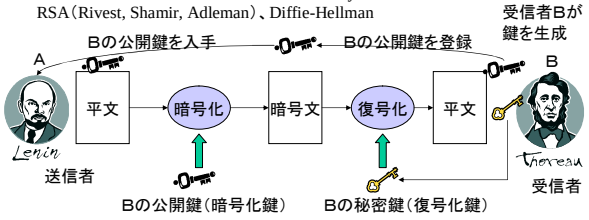
共通鍵暗号

共通鍵暗号：暗号化と復号化に同じ鍵（secret keyと呼ぶ）を使用
処理が高速なので、大量データの暗号化に使用
鍵を秘密にして配送する必要がある。
複数の相手と通信する場合、鍵の本数が増えて、管理が大変
DES（Data Encryption Standard）、IDEA、RC5



公開鍵暗号

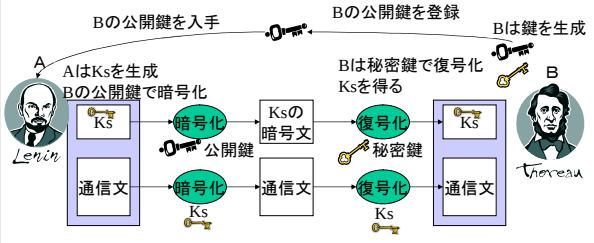
暗号化と復号化に異なる鍵を使用．Aの鍵：Aさんが生成した鍵という意味
暗号化鍵を公開（公開鍵）．復号化鍵は秘密にする（秘密鍵）
送信時：受信者の公開鍵で暗号化．受信時：受信者の秘密鍵（private key）で復号化
　安全性は高い（秘密にした鍵配送が不要）が、処理量が大きい
　ディジタル署名や共通鍵暗号の鍵（secret key）の配送に使用
　RSA（Rivest, Shamir, Adleman）、Diffie-Hellman



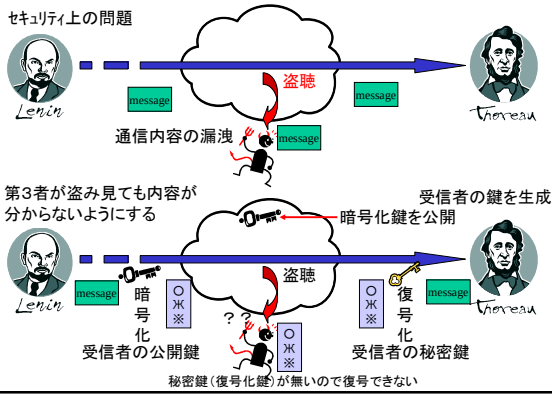
公開鍵は、全員に公開しており、通信相手が何人いても、1つでよい。
秘密鍵を持った受信者だけが、暗号文を送信者が送った平文に戻せる
共通鍵暗号とは異なり、公開鍵（暗号化鍵）では、復号化できない

組み合わせ方式

- 共通鍵暗号と公開鍵暗号の組み合わせ
- ①Aは共通鍵暗号のsecret key（Ks）を公開鍵暗号により、Bに配送
  - secret keyを第三者に見られないようにBに送ることができる
- ②以後の通信は、Aのsecret keyを用いた共通鍵暗号で行う
- ③Ksは使い捨てにする（次に通信するときは、別の鍵を使う）



公開鍵暗号（秘匿通信）



鍵の種類と暗号化の方式

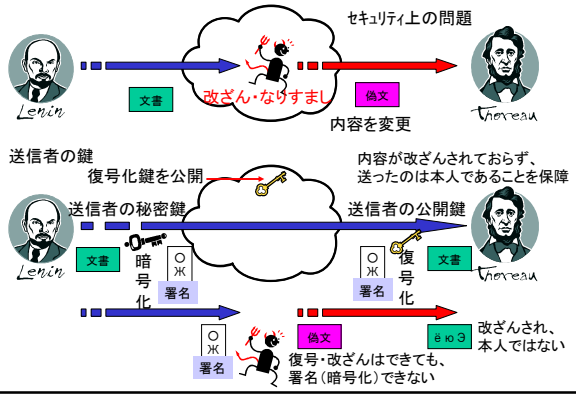
- 暗号化鍵：平文を暗号文に変換する（暗号化）ときに使用
- 復号化鍵：暗号文を平文に戻す（復号化）ときに使用

- 共通鍵暗号：暗号化鍵＝復号化鍵である暗号化方式
  - この共通鍵は、絶対に秘密にしておく必要がある
  - 正式名称は、「秘密鍵」（Secret key）という
    - Secret：秘密の、隠れた
- 公開鍵暗号：暗号化鍵≠復号化鍵である暗号化方式
  - 公開鍵（Public key）：公開する方の鍵
  - 秘密鍵（Private key）：公開しない（秘密にしておく）方の鍵
    - Private：私的な、非公開の、内密の
  - 暗号：受信者の暗号化鍵を公開鍵、復号化鍵を秘密鍵にする
    - 盗み見ても秘密鍵を持たないと中身が分からない。
  - ﾃﾞｨｼﾞﾀﾙ署名：送信者の暗号化鍵を秘密鍵、復号化鍵を公開鍵
    - 秘密鍵を持った者だけが署名を作成できる。

利用者IDと認証

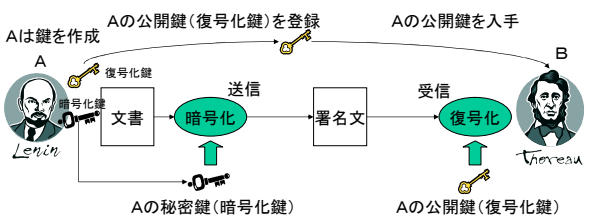
- 利用者ID（アカウント）：登録された利用者に付けられた識別子
  - ｼｽﾃﾑを利用する際に、OSに利用者IDを提示する
- 認証：利用者が、利用者IDで表される本人かどうかを確認すること
- パスワード：合言葉。認証の方式として最も一般的なもの。
- パスワードを守るためのOSの仕組み
  - 裸のﾊﾟｽﾜｰﾄﾞを持たない。Unixのﾊﾟｽﾜｰﾄﾞﾌｧｲﾙは、暗号化情報。
    - 入力ﾊﾟｽﾜｰﾄﾞ→暗号化関数で変換→ﾊﾟｽﾜｰﾄﾞﾌｧｲﾙと照合
    - ﾊﾟｽﾜｰﾄﾞﾌｧｲﾙが盗まれてもﾊﾟｽﾜｰﾄﾞは漏れない。
  - 誤ったﾊﾟｽﾜｰﾄﾞを何回か投入したら認証失敗として打ち切る
    - 記録をとる（不正侵入攻撃かも知れない）
- 辞書に載っている単語をﾊﾟｽﾜｰﾄﾞにすると危険
  - 全単語を暗号化関数で変換してﾊﾟｽﾜｰﾄﾞﾌｧｲﾙと比較
    - たかが数万の単語ならあっという間に総当りできる

ディジタル署名の目的



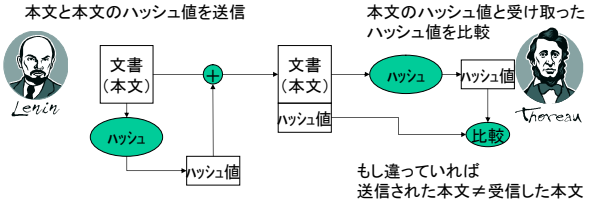
ディジタル署名（基本方式）

ﾃﾞｨｼﾞﾀﾙ署名：ﾒｯｾｰｼﾞの作成者が本人であることを確認する
（受け取ったﾒｯｾｰｼﾞは送信者本人が作成したものである）
①ﾒｯｾｰｼﾞが改ざんされていない、②署名したのは本人である
③「自分は署名しなかった」と主張することができない否認防止にもなる



ﾒｯｾｰｼﾞを送信者の秘密鍵で暗号化（署名）。署名文を送信者の公開鍵で復号化。
秘密鍵を持った人だけが署名文を作成できる
但し、メッセージ全体に署名するため、処理時間がかかる（計算量が多い）

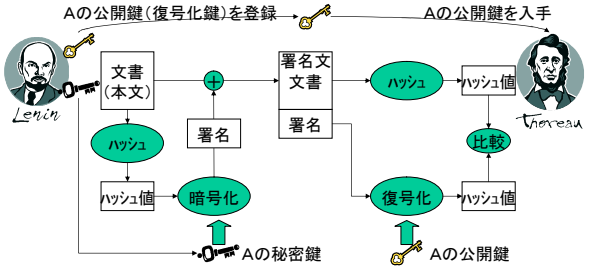
ハッシュ値の利用



ハッシュ：任意長のﾒｯｾｰｼﾞを固定長の短い数値に変換する関数
・MD5：128ビットのハッシュ値を生成（RFC1321）
・SHA1：160ビットのハッシュ値を生成（米国政府の標準ハッシュとして採用）

ハッシュ値の計算は，処理量が少ない．
ただし，ハッシュ値は，だれにでも計算できる．
このままでは，なりすましや改ざんを防ぐことはできない．
→ハッシュ値を用いた比較とディジタル署名の基本方式を組み合わせる．

ディジタル署名（ハッシュ値に署名）

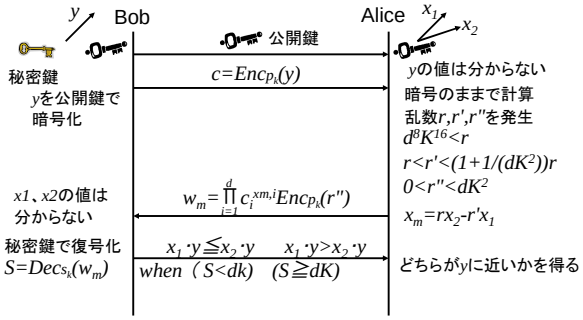


文書のハッシュ値に署名（暗号化）　　署名を復号化し、ハッシュ値を比較

なりすましの防止
ハッシュ値を正しく暗号化できるのは秘密鍵を持った本人のみ．
改ざんの有無を検出
ハッシュ値が一致するのは，改ざんされていない場合のみ．

参考：より高度な暗号の応用例

PPDM（Privacy Preserving Data Mining：プライバシー保護データマイニング）
Aliceのx1とx2のどちらが、Bobのyに近いか？　情報を秘密にしたままで判定



加法的準同型性という性質を持った暗号を用いる

参考：加法的準同型性暗号

s_k：秘密鍵
p_k：公開鍵
Enc_{pk}：暗号化
Dec_{sk}：復号化
$m_1, m_2 \in Z^*_n$：平文

$Dec_{sk}(Enc_{pk}(m_1) \cdot Enc_{pk}(m_2)) = m_1 + m_2$
$Dec_{sk}(Enc_{pk}(m_1)^{m_2}) = m_1 m_2$

例：Pailler暗号系

$p_k = n = pq$　　$p,q \in P$
$sk = \lambda = lcm(p-1, q-1)$
$c = Enc_n(m) = g^m r^n \bmod n^2$　　$g \in z^*_{n^2}$
$m = Dec_\lambda(c) = L(c^\lambda \bmod n^2) L(g^\lambda \bmod n^2)^{-1}$　　$L(u) = (u-1)/n$