2012 年度秋期

数の世界

(担当: 角皆)

本授業の最後に、

今まで触れてきたような整数論が応用されている

数理技術　の中から、

公開鍵暗号（RSA暗号）

について紹介しよう。

物理技術 (17 世紀以来) :

基礎数理 $\Longrightarrow$ **物理現象** $\Longrightarrow$ **実用技術**

理論的裏付け　　**仕組みの構成**

情報技術 (20 世紀以来) :

数理現象 $\Longrightarrow$ 数理技術 $\Longrightarrow$ 実用技術

仕組みの構成　　物理的実現

数理の解明が直接に技術発展に繋がる

物理技術 (17 世紀以来) :

基礎数理 $\Longrightarrow$ **物理現象** $\Longrightarrow$ **実用技術**

理論的裏付け　　**仕組みの構成**

情報技術 (20 世紀以来) :

数理現象 $\Longrightarrow$ **数理技術** $\Longrightarrow$ **実用技術**

仕組みの構成　　**物理的実現**

数理の解明が直接に技術発展に繋がる

物理技術 (17 世紀以来) :

基礎数理 $\Longrightarrow$ **物理現象** $\Longrightarrow$ **実用技術**

理論的裏付け　　**仕組みの構成**

情報技術 (20 世紀以来) :

数理現象 $\Longrightarrow$ **数理技術** $\Longrightarrow$ **実用技術**

仕組みの構成　　**物理的実現**

数理の解明が直接に技術発展に繋がる

計算機で扱えるもの

計算機では本質的に

有限・離散

のものしか扱えない

- 無限・連続のものの近似
- 有限・離散であることの積極的活用

計算機で扱えるもの

計算機では本質的に

有限・離散

のものしか扱えない

- 無限・連続のものの近似
- 有限・離散であることの積極的活用

計算機で扱えるもの

計算機では本質的に

有限・離散

のものしか扱えない

- 無限・連続のものの近似
- 有限・離散であることの積極的活用

有限の算術 (剰余系)

m : 1 以上の整数を一つ取って固定

m で割った余りのみに注目して計算する

a と b とが m を法として合同
(congruent modulo m)

$$a \equiv b \pmod{m}$$

$\overset{\leftarrow}{\Longleftrightarrow}$ a と b とを m で割った余りが等しい

$\Longleftrightarrow$ $m \mid (a-b)$ ($a-b$ が m で割切れる)

$\Longleftrightarrow$ $\exists t \in \mathbb{Z} : a - b = mt$

剰余系の演算

剰余のみに着目して
(well-defined に) 足し算・掛け算が出来る

$$a \equiv a', b \equiv b' \pmod{m}$$

$$\Longrightarrow a + b \equiv a' + b', ab \equiv a'b' \pmod{m}$$

剰余系の演算

$\mathrm{mod}\ m$ で考えたとき、a, b に対して、

$a + x \equiv b \pmod{m}$ となる x は
$\mathrm{mod}\ m$ で丁度 1 つ存在する

$\longrightarrow$ $\mathrm{mod}\ m$ で引き算が出来る
$\mathrm{mod}\ m$ の世界で “$x = b - a$”

しかし、

$ax \equiv b \pmod{m}$ となる x は
($a \not\equiv 0 \pmod{m}$ でも) 存在するとは限らない

$\longrightarrow$ $\mathrm{mod}\ m$ で割り算が出来るとは限らない

剰余系の演算

$\mathrm{mod}\ m$ で考えたとき、a, b に対して、

$a + x \equiv b \pmod{m}$ となる x は
$\mathrm{mod}\ m$ で丁度 1 つ存在する

$\longrightarrow$ $\mathrm{mod}\ m$ で引き算が出来る
$\mathrm{mod}\ m$ の世界で “$x = b - a$”

しかし、

$ax \equiv b \pmod{m}$ となる x は
$(a \not\equiv 0 \pmod{m}$ でも) 存在するとは限らない

$\longrightarrow$ $\mathrm{mod}\ m$ で割り算が出来るとは限らない

素数を法とする剰余系の演算

実際には $\gcd(a, m) = 1$ **のときは、**

$ax \equiv b \pmod{m}$ **となる** x **が**

$\bmod m$ **で丁度** 1 **つ存在する**

特に、法 m **が素数** p **なら、**

$a \not\equiv 0 \pmod{p}$ **であれば、**

$ax \equiv b \pmod{p}$ **となる** x **は**

必ず丁度 1 **つ見付かる**

$\longrightarrow$ $\bmod p$ **で (0 以外での) 割り算も出来る**

$\bmod p$ **の世界で** “$x = b/a$”

素数を法とする剰余系の演算

実際には $\gcd(a, m) = 1$ **のときは、**

$ax \equiv b \pmod{m}$ **となる** x **が**

$\bmod m$ **で丁度** 1 **つ存在する**

特に、法 m **が素数** p **なら、**

$a \not\equiv 0 \pmod{p}$ **であれば、**

$ax \equiv b \pmod{p}$ **となる** x **は**

必ず丁度 1 **つ見付かる**

$\longrightarrow$ $\bmod p$ **で (**0 **以外での) 割り算も出来る**

$\bmod p$ **の世界で** “$x = b/a$”

有限体

体 (field) : 四則演算 (加減乗除) が出来る集合

例 : 有理数体 $\mathbb{Q}$ ・実数体 $\mathbb{R}$ ・複素数体 $\mathbb{C}$

素数 p に対して、

p で割った余りの集合 $\overset{1:1}{\longleftrightarrow} \{0, 1, \ldots, p-1\}$

この中で (0 で割る以外の) 四則演算が出来る

$\cdots$ 有限体 (finite field) ・p 元体　　$\mathbb{F}_p$, $\mathbb{Z}/p\mathbb{Z}$

$\longrightarrow$ 四則演算しか使わない計算なら、

有限体 $\mathbb{F}_p$ 上でも $\mathbb{R}$ や $\mathbb{C}$ と同様に行なえる

有限体

体 (field) ： 四則演算 (加減乗除) が出来る集合

例 ： 有理数体 $\mathbb{Q}$ ・実数体 $\mathbb{R}$ ・複素数体 $\mathbb{C}$

素数 p に対して、

p で割った余りの集合 $\overset{1:1}{\longleftrightarrow} \{0, 1, \ldots, p-1\}$

この中で (0 で割る以外の) 四則演算が出来る

$\cdots$ 有限体 (finite field) ・p 元体　　$\mathbb{F}_p$, $\mathbb{Z}/p\mathbb{Z}$

$\longrightarrow$ 四則演算しか使わない計算なら、
有限体 $\mathbb{F}_p$ 上でも $\mathbb{R}$ や $\mathbb{C}$ と同様に行なえる

有限体

体 (field) : 四則演算 (加減乗除) が出来る集合

例 : 有理数体 $\mathbb{Q}$ • 実数体 $\mathbb{R}$ • 複素数体 $\mathbb{C}$

素数 p に対して、

p で割った余りの集合 $\overset{1:1}{\longleftrightarrow} \{0, 1, \ldots, p-1\}$

この中で (0 で割る以外の) 四則演算が出来る

$\cdots$ **有限体 (finite field) • p 元体** $\mathbb{F}_p,\ \mathbb{Z}/p\mathbb{Z}$

$\longrightarrow$ **四則演算しか使わない計算なら、**
有限体 $\mathbb{F}_p$ 上でも $\mathbb{R}$ や $\mathbb{C}$ と同様に行なえる

数理技術としての応用例

暗号 (cryptography)

- 秘密通信
- 電子認証・電子署名
- 鍵共有

などに用いられる

暗号の利用

- **古典的： 戦争・謀略など**
- **現代： 情報通信一般**

$\longrightarrow$ **個人の独立を守るための重要な数理技術**

暗号の利用

- **古典的： 戦争・謀略など**
- **現代： 情報通信一般**

$\longrightarrow$ **個人の独立を守るための重要な数理技術**

暗号の利用

- **古典的： 戦争・謀略など**
- **現代： 情報通信一般**

$\longrightarrow$ 個人の独立を守るための重要な数理技術

既に身近な暗号の利用

メディアセンターのコンピュータを使う際の
パスワードによる本人認証にも
暗号 (暗号化) が使われている

入力したパスワードを
保管してあるデータと照合しているのだが、

実は、
パスワードそのものを保管しているのではない

定まった方式 (暗号化関数) で
パスワードを変換して保管している

既に身近な暗号の利用

メディアセンターのコンピュータを使う際の
パスワードによる本人認証にも
暗号 (暗号化) が使われている

入力したパスワードを
保管してあるデータと照合しているのだが、

実は、
パスワードそのものを保管しているのでは**ない**

定まった方式 (暗号化関数) で
パスワードを変換して保管している

既に身近な暗号の利用

メディアセンターのコンピュータを使う際の
パスワードによる本人認証にも
暗号 (暗号化) が使われている

入力したパスワードを
保管してあるデータと照合しているのだが、

実は、
パスワードそのものを保管しているのでは**ない**

定まった方式 (暗号化関数) で
パスワードを変換して保管している

パスワードによる本人認証

- 暗号化関数でパスワードを変換して保管
- 入力したパスワードを暗号化関数で変換して、
  保管してある文字列と照合

暗号化関数に要請される性質は？

- 間違った文字列では変換結果が一致しない
  $\longrightarrow$ 異なる入力には異なる値を返す (単射)
- 保管してある文字列が露見しても
  元のパスワードが判明しない
  $\longrightarrow$ 一方向性関数 (one-way function)

パスワードによる本人認証

- 暗号化関数でパスワードを変換して保管
- 入力したパスワードを暗号化関数で変換して、
  保管してある文字列と照合

暗号化関数に要請される性質は？

- 間違った文字列では変換結果が一致しない
  $\longrightarrow$ 異なる入力には異なる値を返す (単射)
- 保管してある文字列が露見しても
  元のパスワードが判明しない
  $\longrightarrow$ 一方向性関数 (one-way function)

パスワードによる本人認証

- 暗号化関数でパスワードを変換して保管
- 入力したパスワードを暗号化関数で変換して、
  保管してある文字列と照合

暗号化関数に要請される性質は？

- 間違った文字列では変換結果が一致しない
  $\longrightarrow$ 異なる入力には異なる値を返す (単射)
- 保管してある文字列が露見しても
  元のパスワードが判明しない
  $\longrightarrow$ **一方向性関数** (one-way function)

パスワードによる本人認証

良い暗号化関数で変換して保管していても

通信経路の途中で盗聴されてしまっては

パスワードが露見してしまう

⟶ 暗号による秘密通信

パスワードによる本人認証

良い暗号化関数で変換して保管していても

通信経路の途中で盗聴されてしまっては

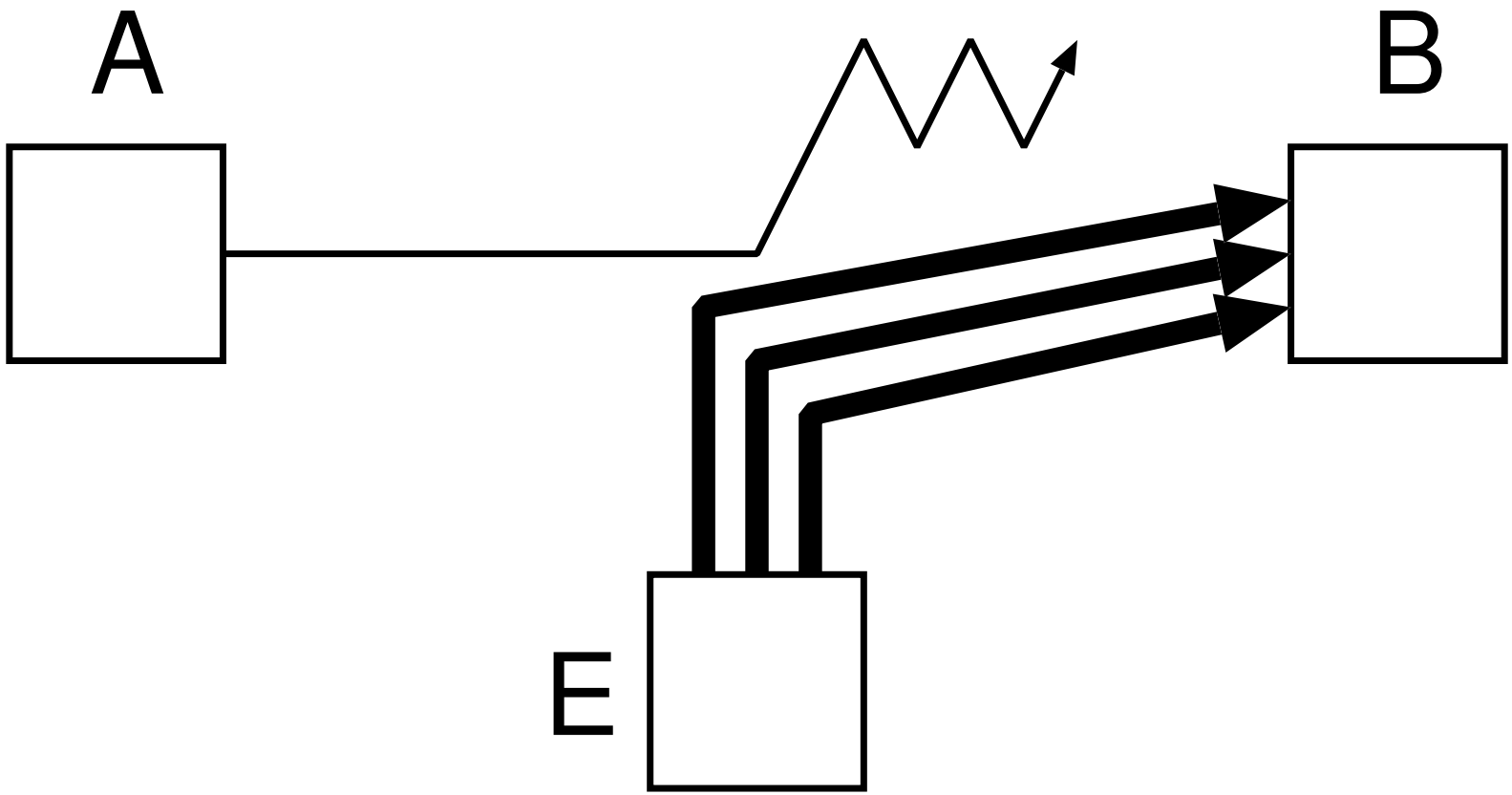
パスワードが露見してしまう

$\longrightarrow$ **暗号による秘密通信**

安全な情報伝達を阻害するもの

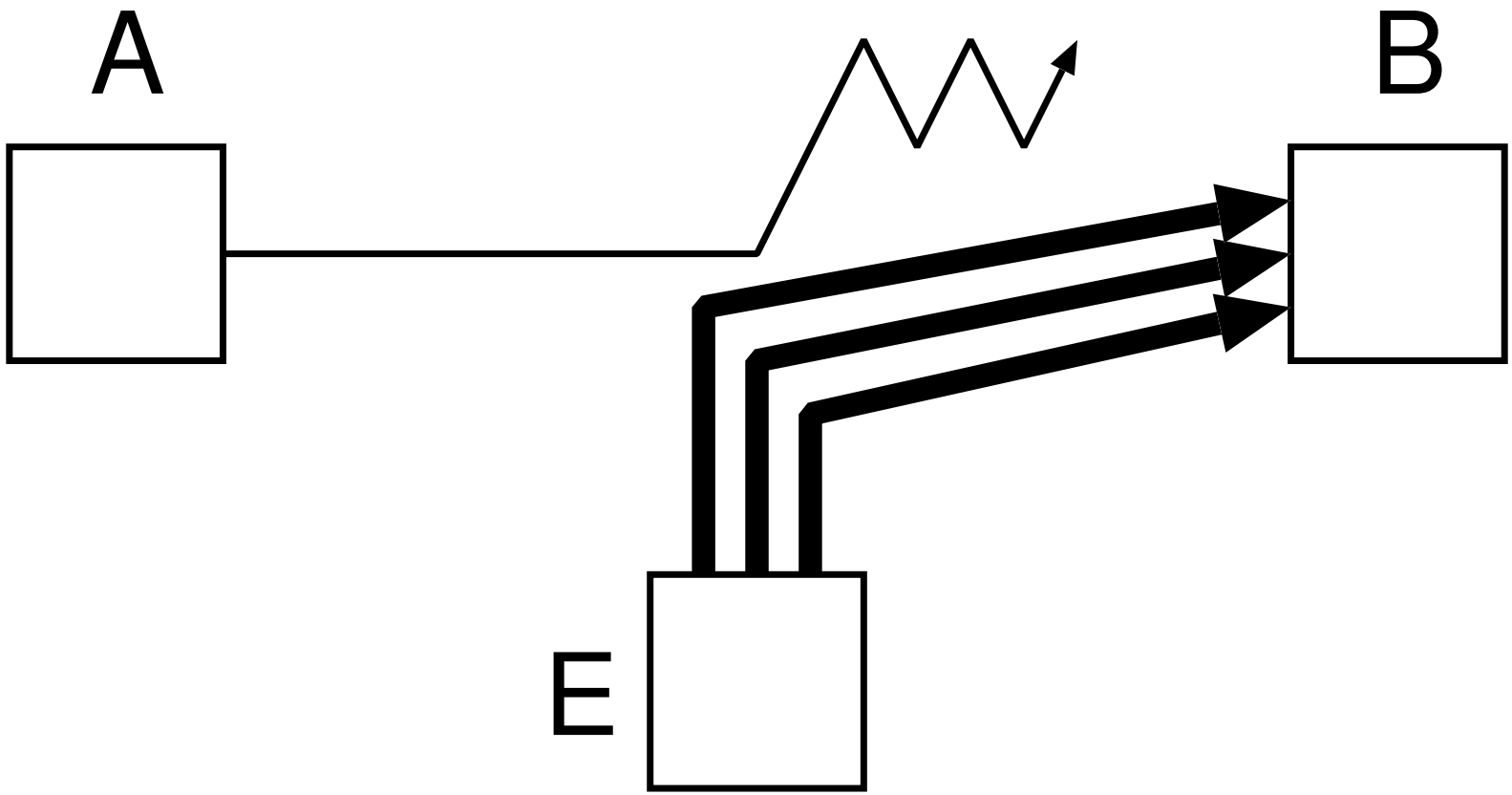
- 妨害 (DoS 攻撃など)
- 盗聴
- 改竄
- なり済まし　　など

DoS (Denial of Service) 攻撃



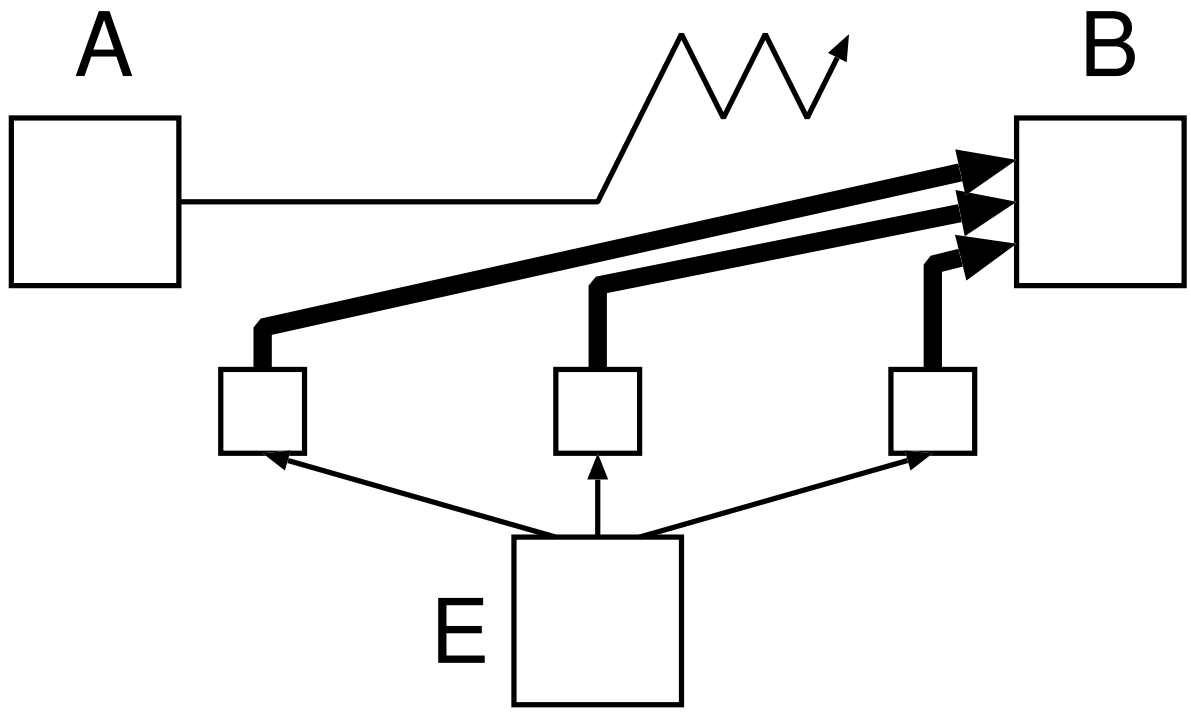
B を機能停止に追い込むには
E に相当のマシンパワーが必要
そこで実際には · · ·

DoS (Denial of Service) 攻撃



B を機能停止に追い込むには
E に相当のマシンパワーが必要
そこで実際には ...

DoS (Denial of Service) 攻撃



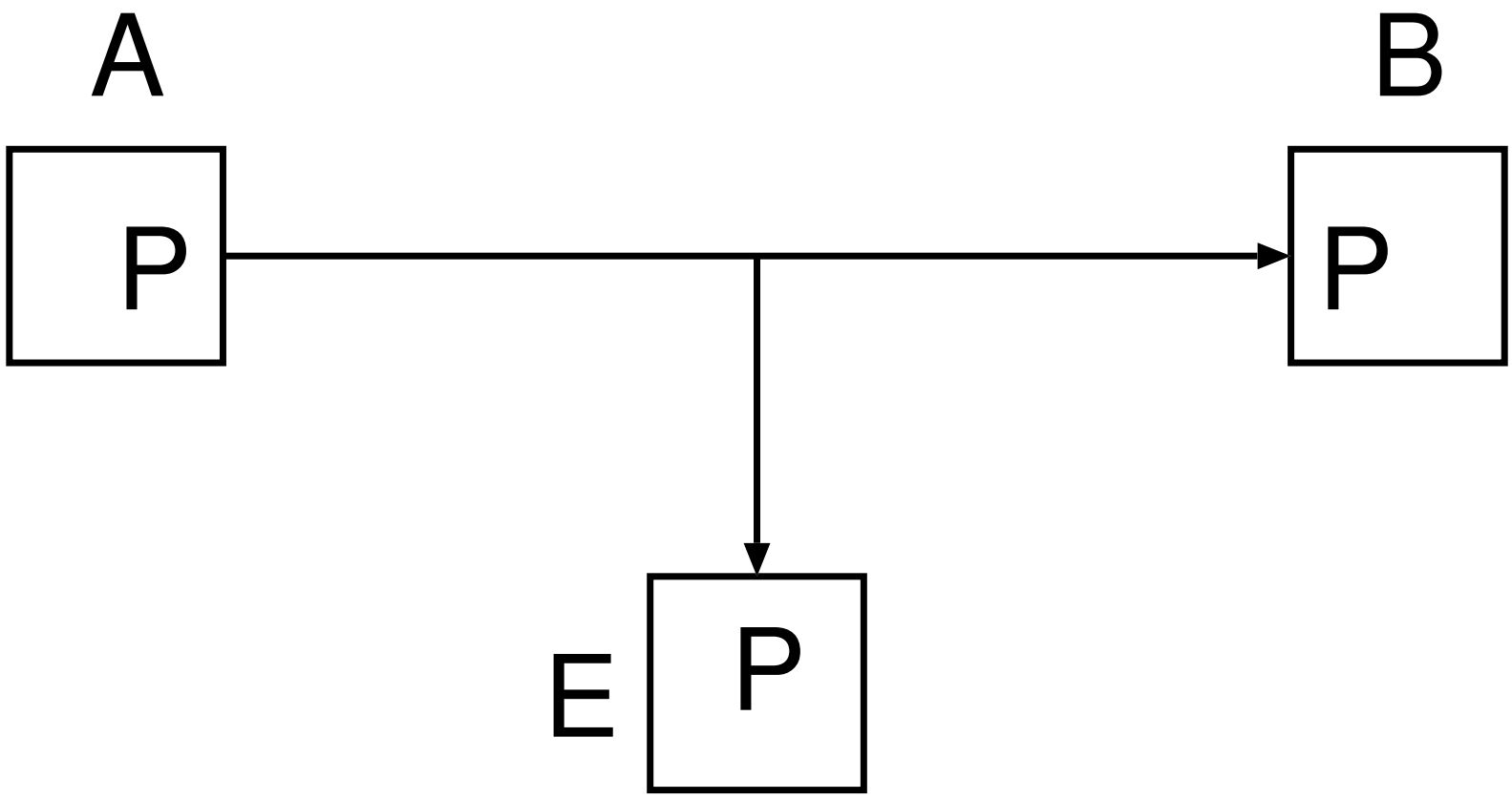
実際には、

コンピュータウイルス・乗っ取りなどで

制御下に置いた多数の機械から一斉に攻撃

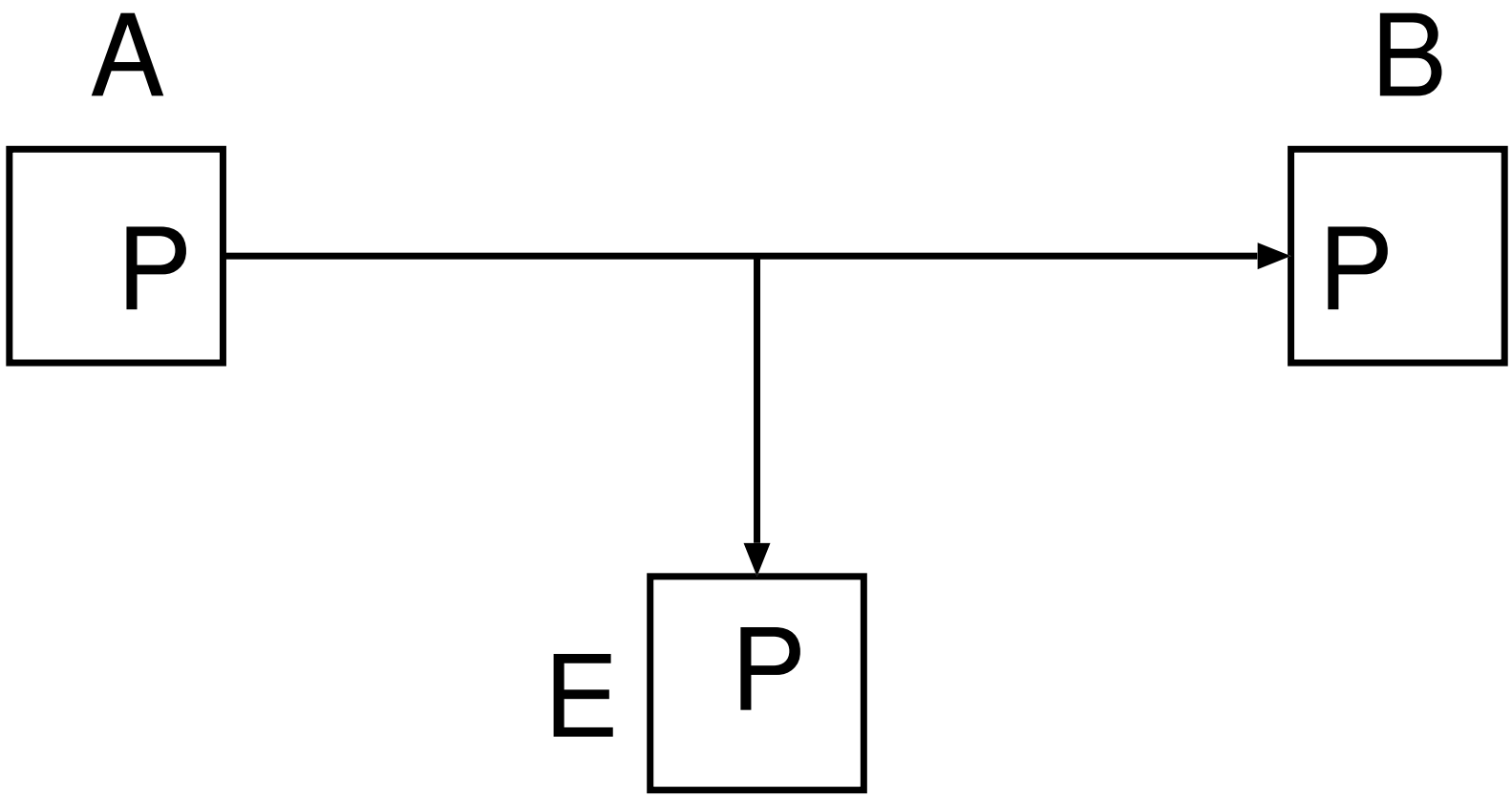
(Distributed DoS, DDoS)

盗聴



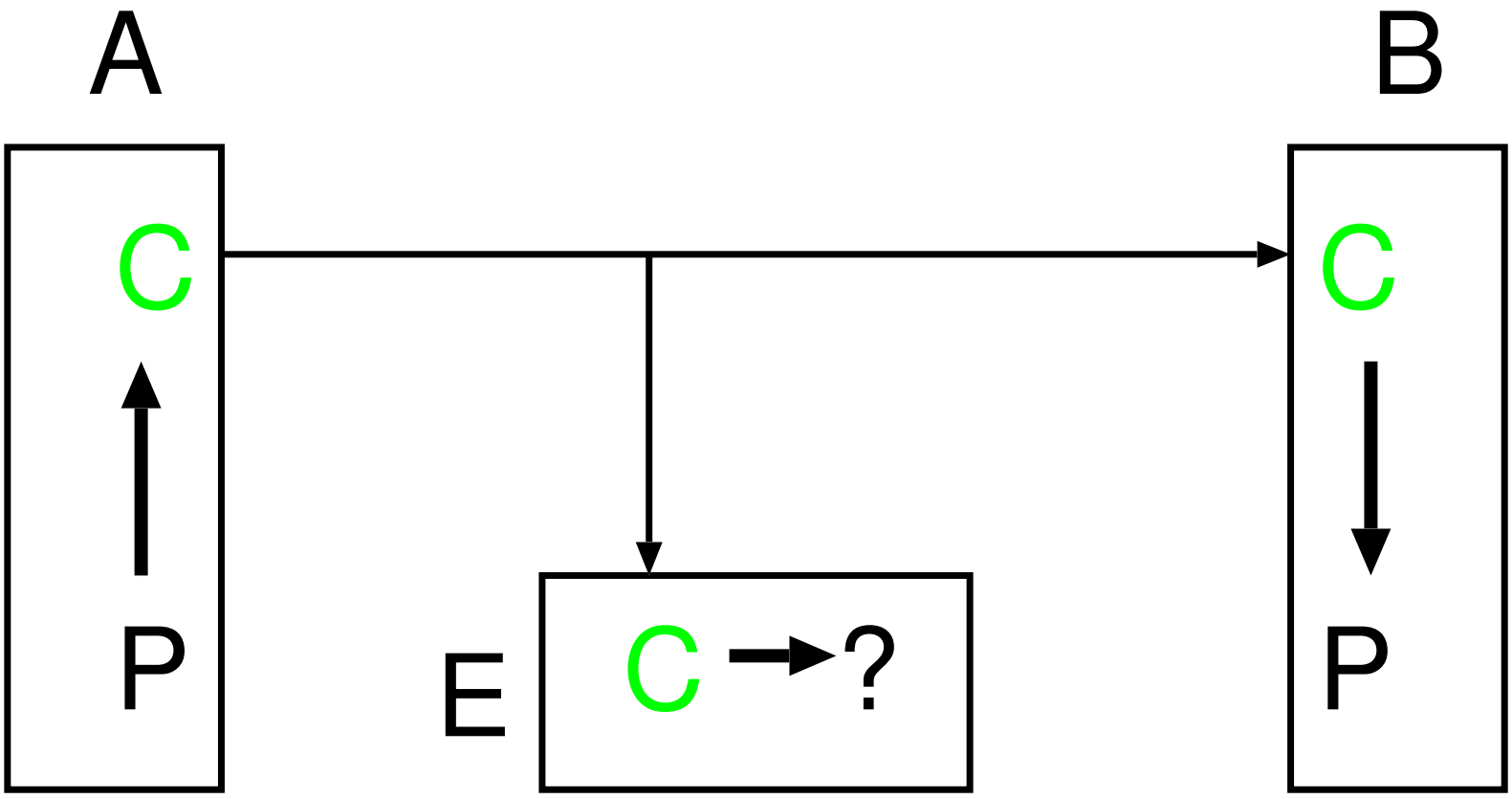
現在の計算機ネットワークの仕組みでは、
事実上、通信経路は誰にでも見られる

盗聴



現在の計算機ネットワークの仕組みでは、
事実上、通信経路は誰にでも見られる

暗号通信で盗聴対策

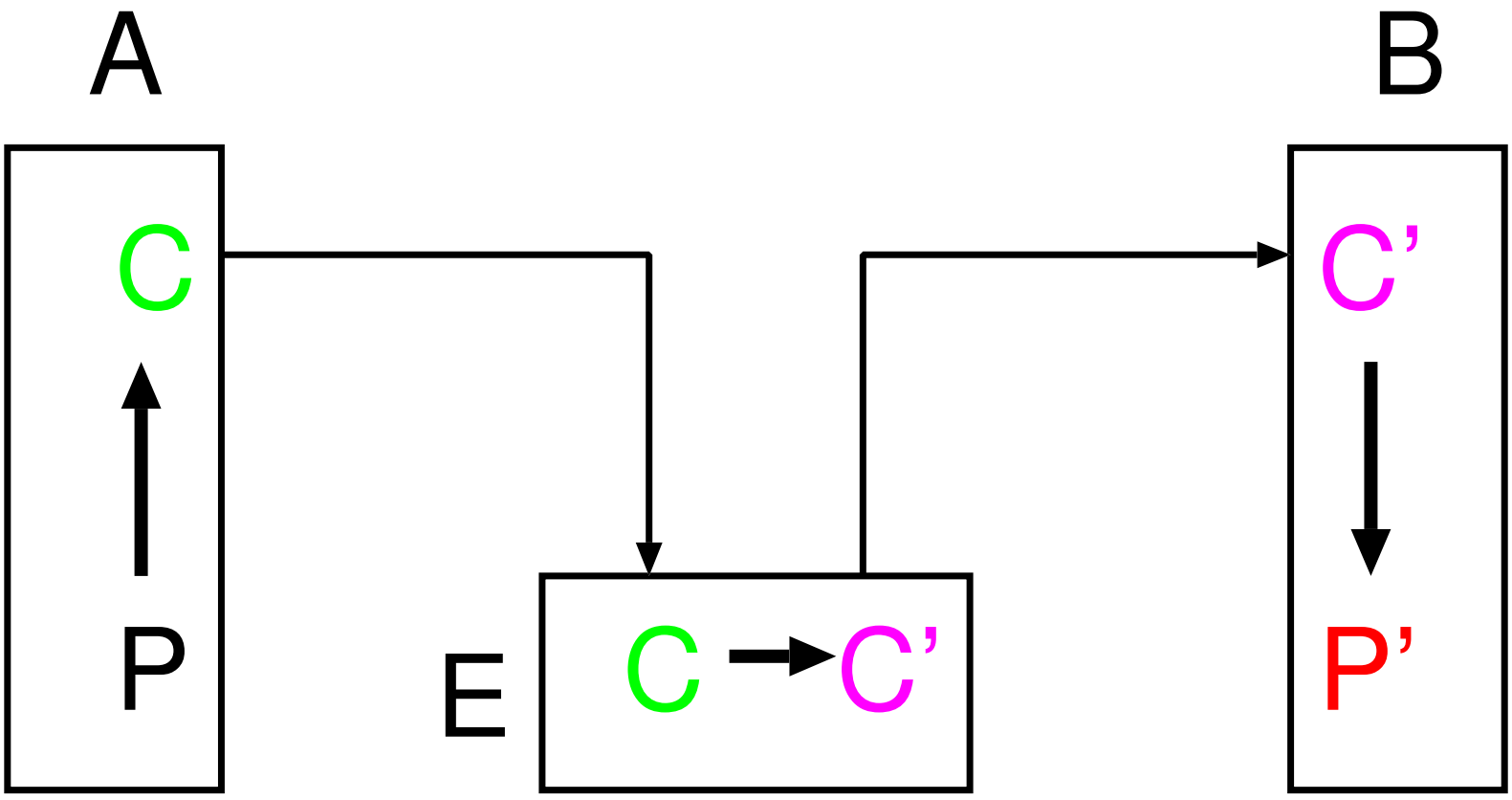


P : 平文 (plain text),　C : 暗号文 (ciphertext)

P → C : 暗号化 (encryption)

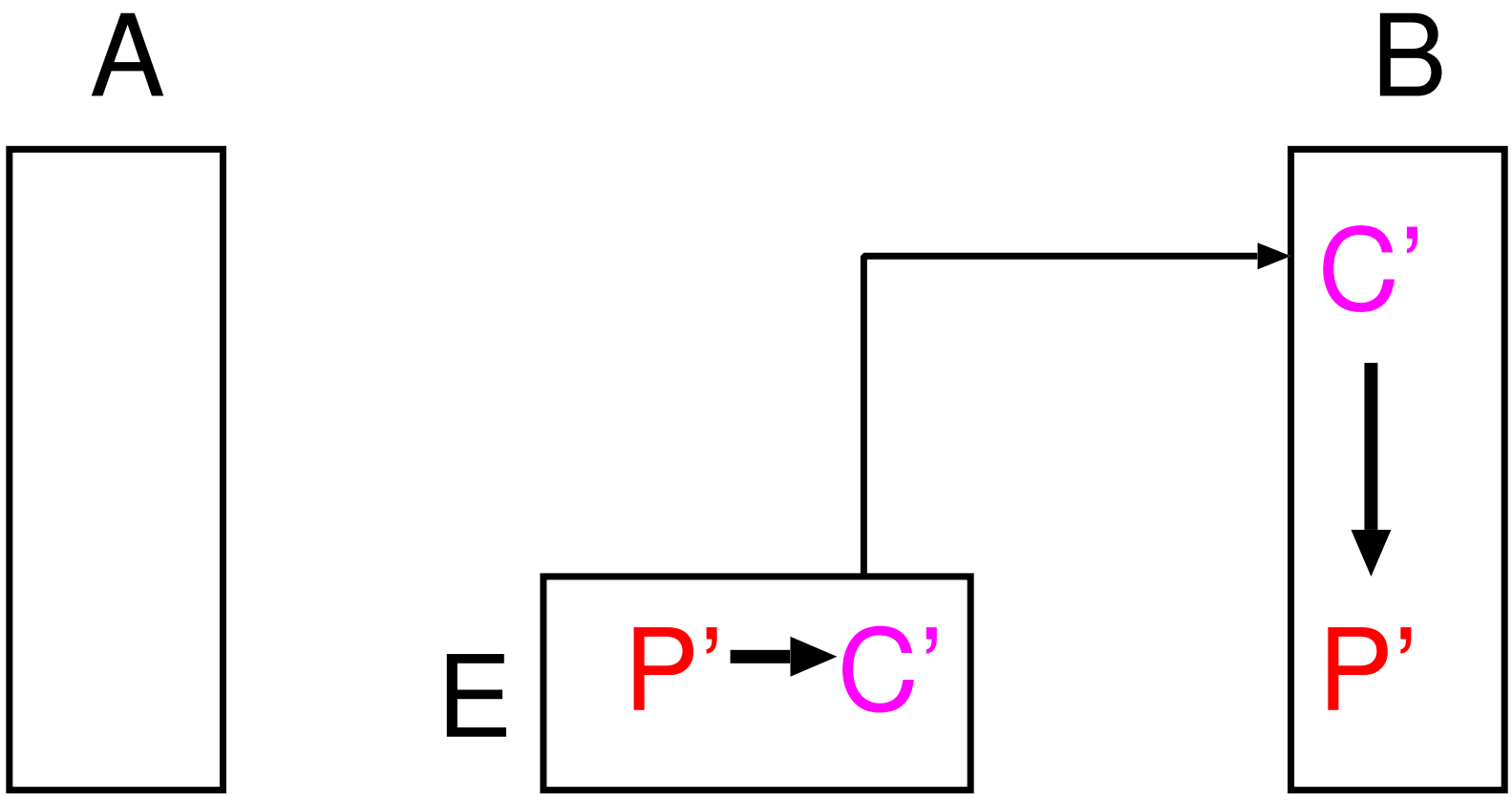
C → P : 復号 (decryption)・解読

改竄



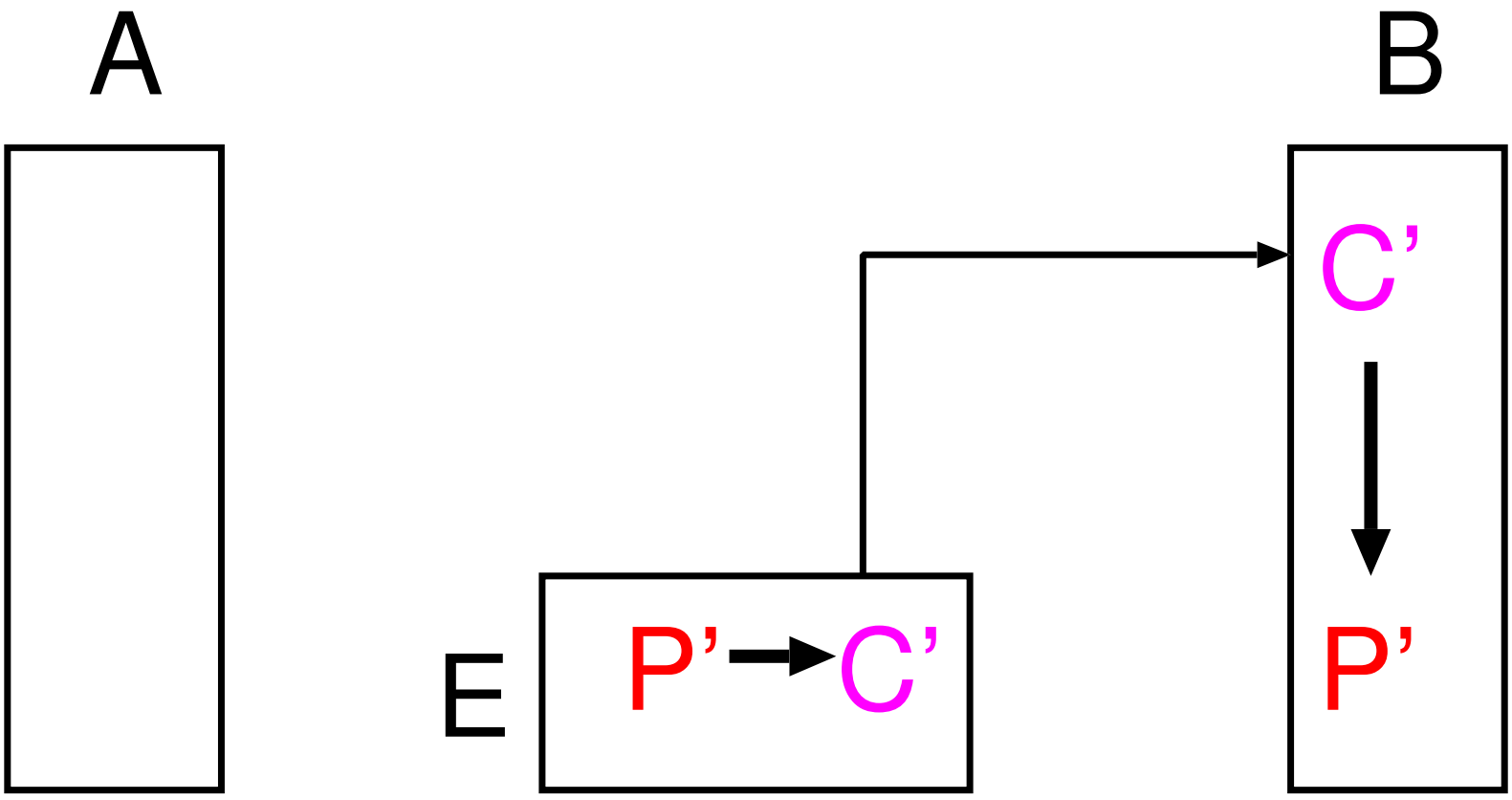
A が送信した情報であることを
確かめられるような仕組みが必要
(電子認証・電子署名)

なり済まし



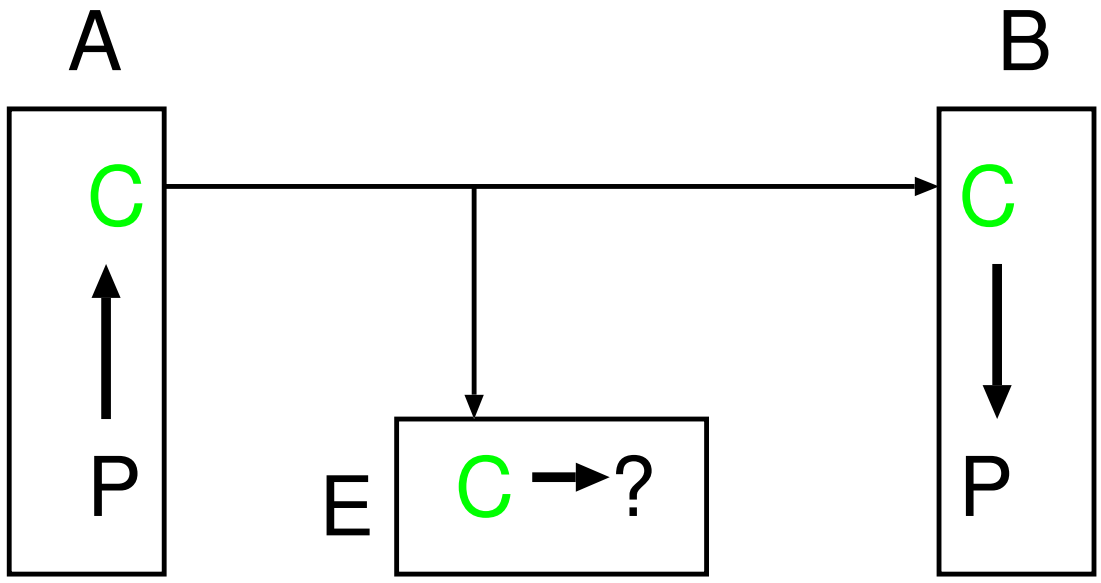
A が送信した情報であることを
確かめられるような仕組みが必要
(電子認証・電子署名)

なり済まし



A が送信した情報であることを
確かめられるような仕組みが必要
(電子認証・電子署名)

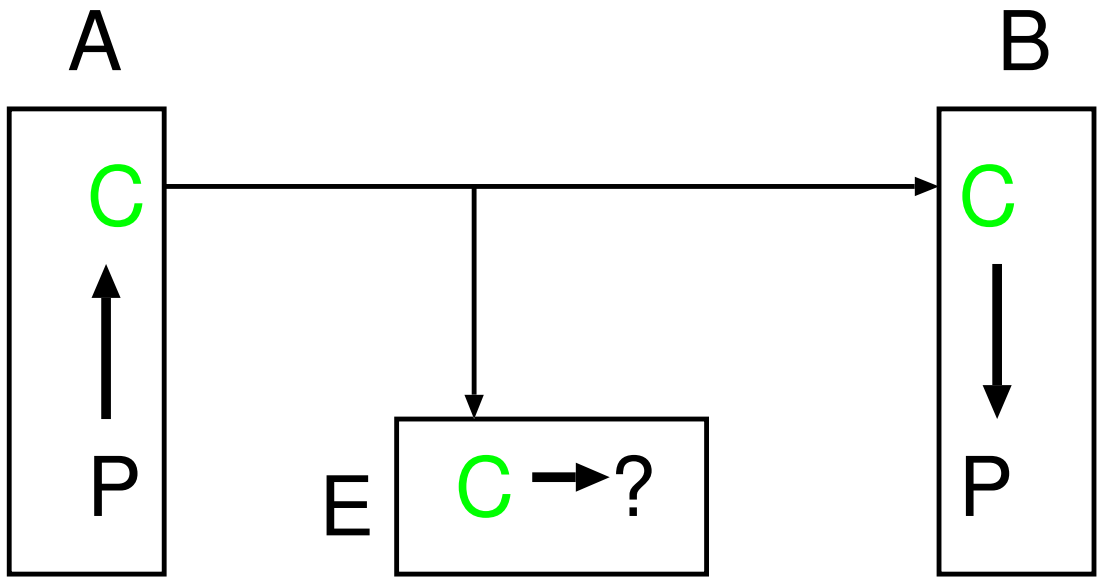
暗号 (cryptography)



- 送信者 A が平文 P を暗号化、暗号文 C を送信
- 受信者 B が暗号文 C を受信、平文 P に復号
- 盗聴者 E は暗号文 C を知っても
  平文 P を復元できない

⟶ B だけが復号鍵を持っていることが必要

暗号 (cryptography)



- 送信者 A が平文 P を暗号化、暗号文 C を送信
- 受信者 B が暗号文 C を受信、平文 P に復号
- 盗聴者 E は暗号文 C を知っても
  平文 P を復元できない

$\longrightarrow$ B だけが**復号鍵**を持っていることが必要

暗号 (cryptography)

仮定：

公開された情報伝達路 (盗聴可能 と仮定) で、

暗号方式を公開 して通信

- **秘密鍵暗号 (共通鍵暗号)**
- **公開鍵暗号**

暗号 (cryptography)

仮定：

公開された情報伝達路 (盗聴可能 と仮定) で、

暗号方式を公開 して通信

- **秘密鍵暗号 (共通鍵暗号)**
- **公開鍵暗号**

暗号 (cryptography)

- **共通鍵暗号 (秘密鍵暗号)**
  - ⋆ 送信者・受信者で同じ鍵を秘密裡に共有
  - ⋆ 共通の鍵で暗号化・復号を行なう

- **公開鍵暗号**
  - ⋆ 暗号化鍵 (公開鍵)・復号鍵 (秘密鍵) が別
  - ⋆ 公開された暗号化鍵を用いて暗号化
  - ⋆ 復号鍵は受信者だけの秘密

暗号 (cryptography)

- **共通鍵暗号** (秘密鍵暗号)
  - ⋆ 送信者・受信者で同じ鍵を秘密裡に共有
  - ⋆ 共通の鍵で暗号化・復号を行なう

- **公開鍵暗号**
  - ⋆ 暗号化鍵 (公開鍵)・復号鍵 (秘密鍵) が別
  - ⋆ 公開された暗号化鍵を用いて暗号化
  - ⋆ 復号鍵は受信者だけの秘密

共通鍵暗号の例： Caesar 暗号

- ここでは、
  a ～ z のアルファベットから成る文字列を
  暗号化

- 共通鍵： $1 \leq n \leq 25$ なる整数 n

- 暗号化： アルファベットを後ろに n 個ずらす

- 復号： アルファベットを前に n 個戻す

  (但し、$\cdots$ xyzabc $\cdots$ と繋がるとする)

共通鍵暗号の例： Caesar 暗号

Caesar 暗号で暗号化された
次の文字列を解読してみよう

phq dqg zrphq iru rwkhuv zlwk rwkhuv

- **共通鍵：** $1 \leq n \leq 25$ **なる整数** n
- **暗号化：アルファベットを後ろに** n **個ずらす**
- **復号：アルファベットを前に** n **個戻す**
  (但し、$\cdots$ **xyzabc** $\cdots$ **と繋がるとする)**

Caesar 暗号の脆弱性

鍵を知らなくても容易に解読されてしまった

何故か？

- 鍵の可能性が少なく、総当たりで倒せる
- 暗号文に平文の特徴が残っている

このような脆弱性を克服した暗号方式が現在では用いられている

- DES (Deta Encryption Standard)
- AES (Advanced Encryption Standard)

Caesar 暗号の脆弱性

鍵を知らなくても容易に解読されてしまった

何故か？

- **鍵の可能性が少なく、総当たりで倒せる**
- **暗号文に平文の特徴が残っている**

このような脆弱性を克服した暗号方式が
現在では用いられている

- **DES (Deta Encryption Standard)**
- **AES (Advanced Encryption Standard)**

Caesar 暗号の脆弱性

鍵を知らなくても容易に解読されてしまった

何故か？

- **鍵の可能性が少なく、総当たりで倒せる**
- **暗号文に平文の特徴が残っている**

このような脆弱性を克服した暗号方式が
現在では用いられている

- **DES (Deta Encryption Standard)**
- **AES (Advanced Encryption Standard)**

現代における暗号への要請

現在の情報化社会では
様々な場面で暗号が使われている

例：インターネット取引 (ネットショッピングなど)

- 不特定多数の人と暗号通信をしたい
- 事前に鍵を共有できない

$\longrightarrow$ 共通鍵暗号では実現が困難

$\longrightarrow$ 公開鍵暗号・鍵共有方式のアイデア
(1976〜77)

現代における暗号への要請

現在の情報化社会では
様々な場面で暗号が使われている

例：インターネット取引 (ネットショッピングなど)

- **不特定多数の人と暗号通信をしたい**
- **事前に鍵を共有できない**

$\longrightarrow$ 共通鍵暗号では実現が困難

$\longrightarrow$ 公開鍵暗号・鍵共有方式のアイデア
(1976〜77)

現代における暗号への要請

現在の情報化社会では
様々な場面で暗号が使われている

例：インターネット取引 (ネットショッピングなど)

- **不特定多数の人と暗号通信をしたい**
- **事前に鍵を共有できない**

$\longrightarrow$ 共通鍵暗号では実現が困難

$\longrightarrow$ 公開鍵暗号・鍵共有方式のアイデア
(1976〜77)

公開鍵暗号

暗号化鍵 (公開鍵)・復号鍵 (秘密鍵) が別

- 事前の鍵共有の必要無し
  $\longrightarrow$ 見ず知らずの人からも送ってもらえる

- 認証・署名機能がある
  $\longrightarrow$ 改竄・なり済ましの対策
  $\longrightarrow$ 否認防止の機能も持つ

公開鍵暗号

暗号化鍵 (公開鍵)・復号鍵 (秘密鍵) が別

- 事前の鍵共有の必要無し
  $\longrightarrow$ 見ず知らずの人からも送ってもらえる

- 認証・署名機能がある
  $\longrightarrow$ 改竄・なり済ましの対策
  $\longrightarrow$ 否認防止の機能も持つ

公開鍵暗号

暗号化鍵 (公開鍵)・復号鍵 (秘密鍵) が別

- 事前の鍵共有の必要無し
  $\longrightarrow$ 見ず知らずの人からも送ってもらえる

- 認証・署名機能がある
  $\longrightarrow$ 改竄・なり済ましの対策
  $\longrightarrow$ 否認防止の機能も持つ

公開鍵暗号

但し、一般には、
暗号化・復号が共通鍵暗号に比べて低速

そこで、

- **始めに公開鍵暗号方式で鍵を送付・共有**
- **その鍵を用いて秘密鍵暗号方式で通信**

というように、組合わせて用いることが多い

公開鍵暗号

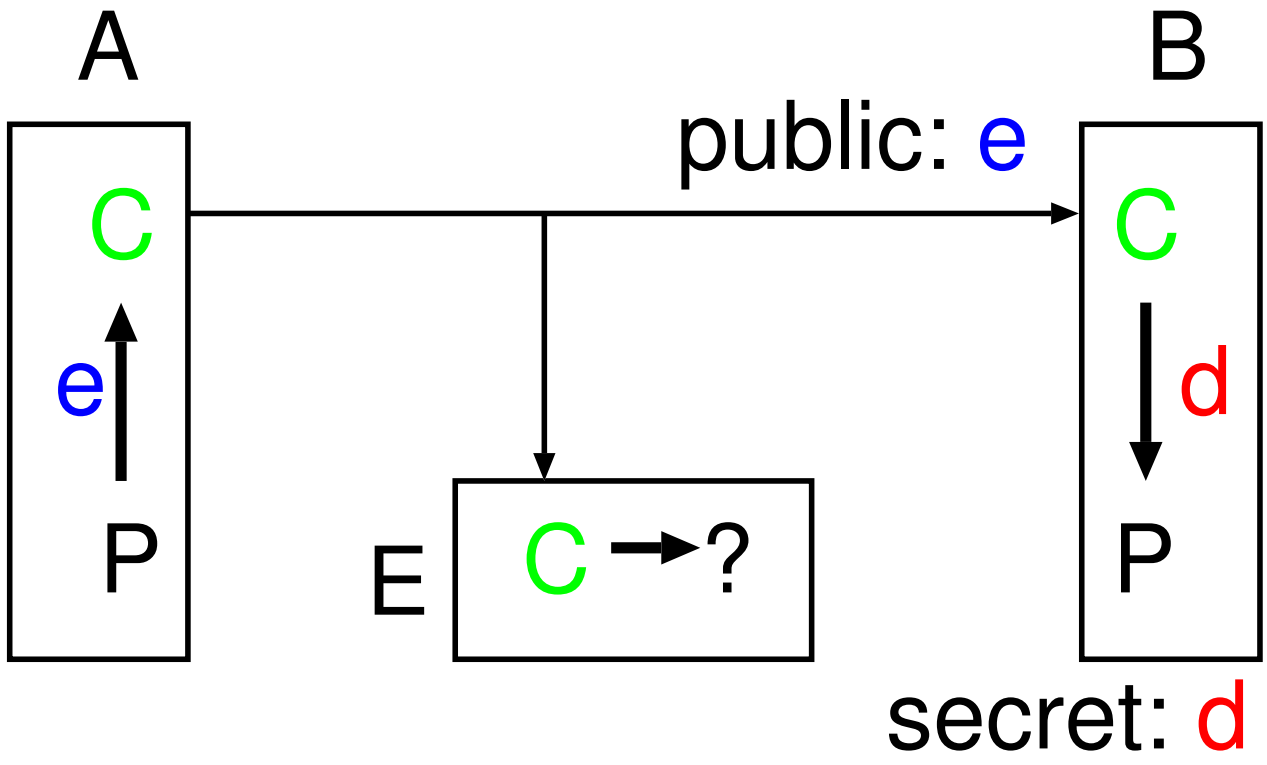
但し、一般には、
暗号化・復号が共通鍵暗号に比べて低速

そこで、

- 始めに公開鍵暗号方式で鍵を送付・共有
- その鍵を用いて秘密鍵暗号方式で通信

というように、組合わせて用いることが多い

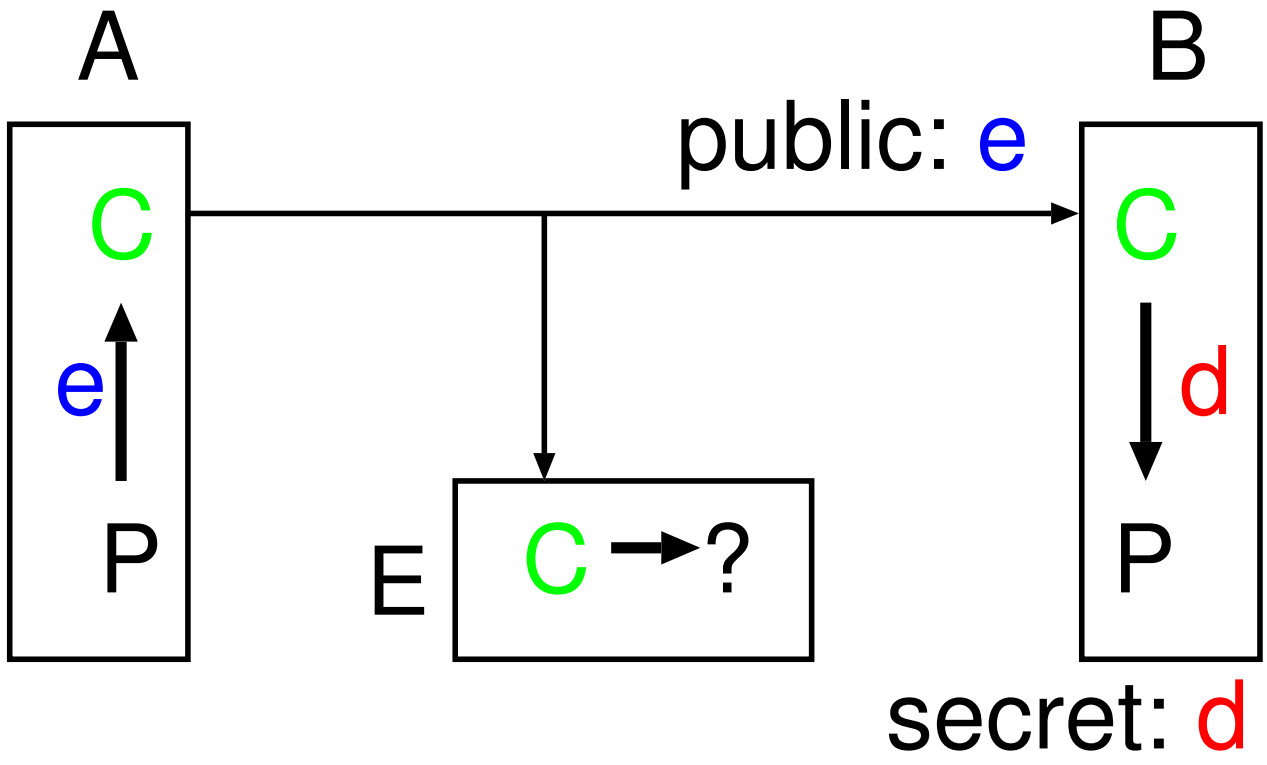
公開鍵暗号による暗号通信



しかし、これだと誰でも暗号化できるので、
A 氏が送った保証がない

⟶ 署名の必要性

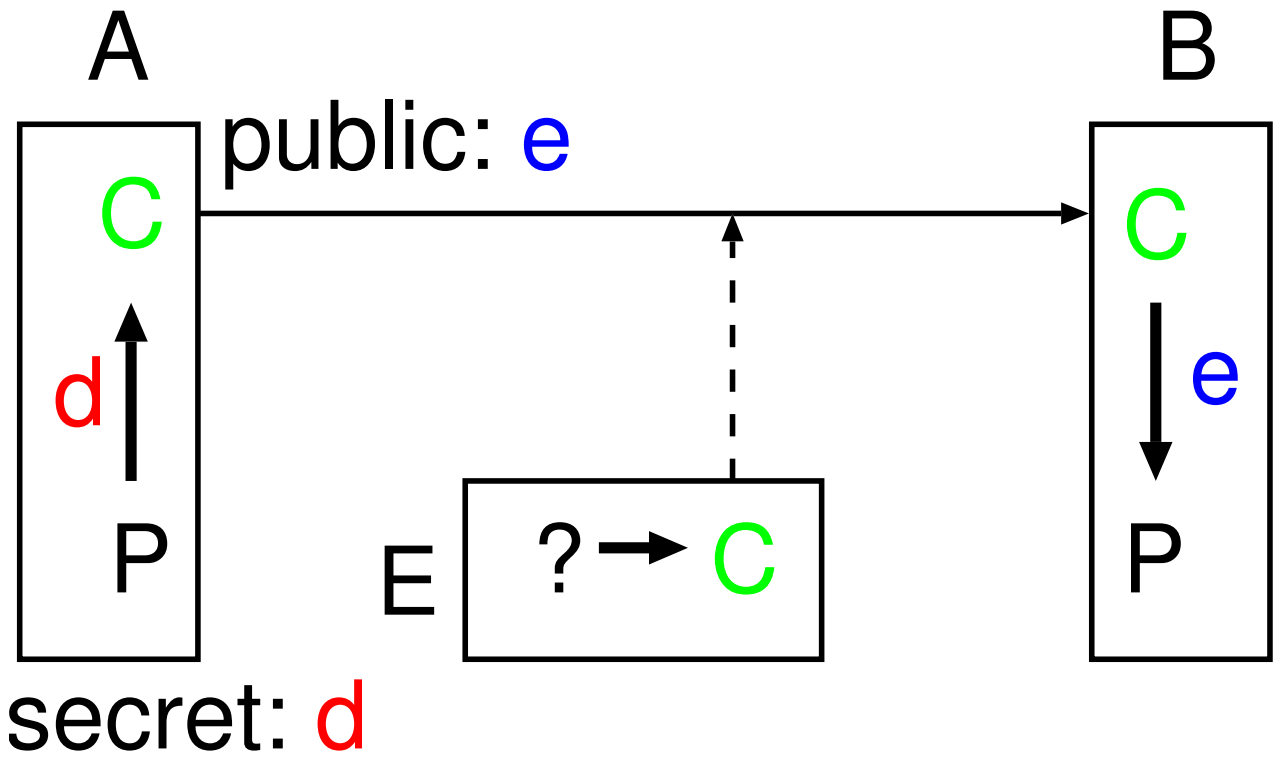
公開鍵暗号による暗号通信



しかし、これだと誰でも暗号化できるので、
A 氏が送った保証がない

⟶ 署名の必要性

公開鍵暗号を用いた認証・署名

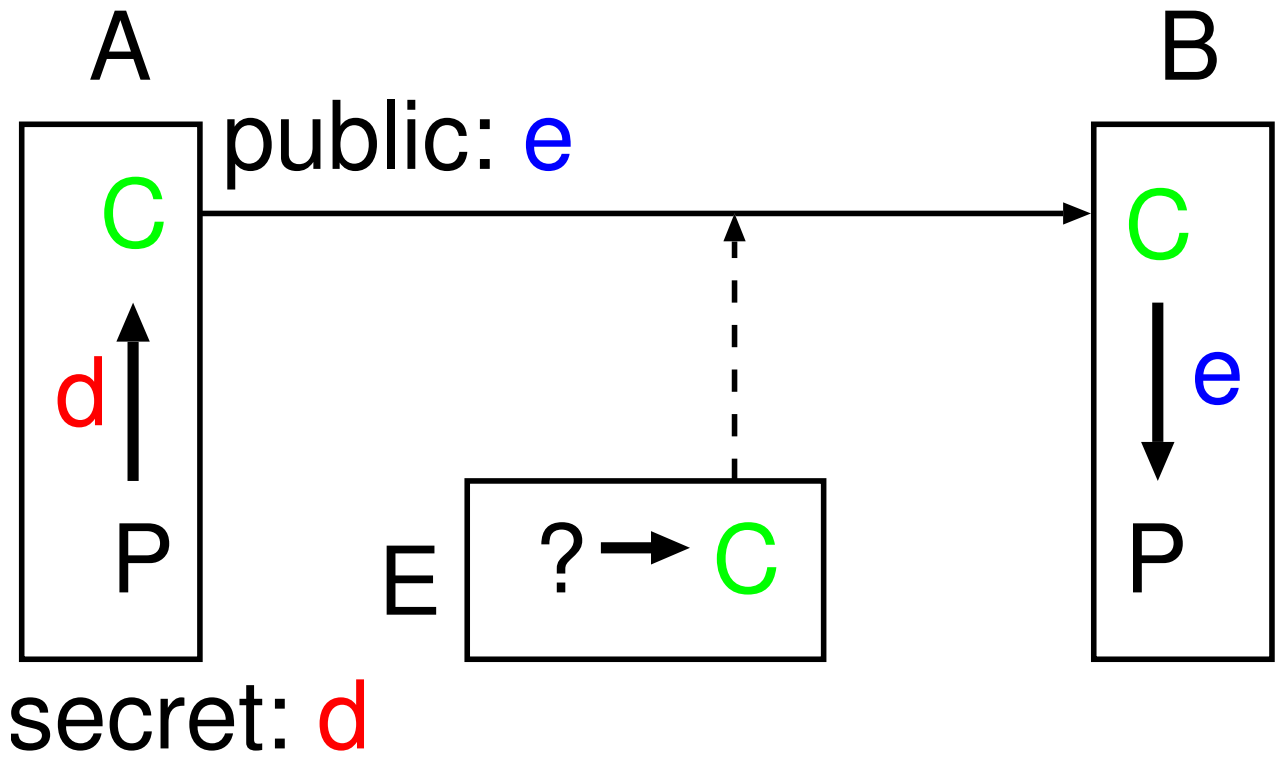


盗聴者 E 氏は

平文 P は判らないが、暗号文 C は盗聴可能

⟶ いつも同じ署名は使えない

公開鍵暗号を用いた認証・署名



盗聴者 E 氏は

平文 P は判らないが、暗号文 C は盗聴可能

⟶ いつも同じ署名は使えない

公開鍵暗号を用いた認証・署名

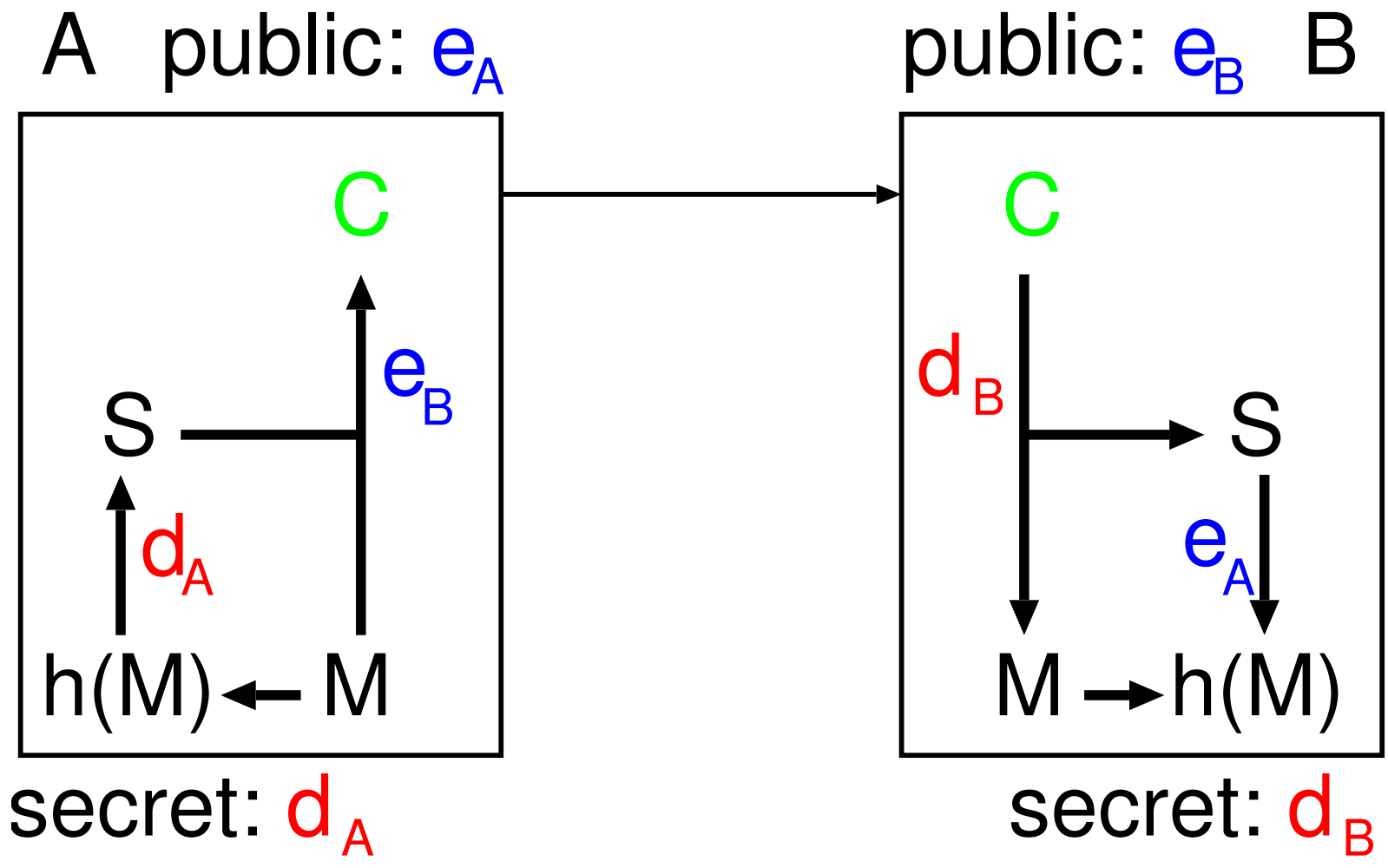
実際には、メッセージ本文 M に対して、

M から決まる短い値 (ハッシュ値) $\mathrm{h}(\mathrm{M})$ を
送信者 A 氏の秘密鍵で暗号化した文字列 S

を本文 M に添付して、

受信者 B 氏の公開鍵で一緒に暗号化して送る

公開鍵暗号を用いた認証・署名



公開鍵暗号の特徴

- 暗号化は誰でも出来る
  (暗号化鍵は公開されている)

- 復号は秘密鍵を知らないと出来ない
  (もの凄く時間が掛かる)

そんな都合の良い仕組みが本当にあるのか？

公開鍵暗号の特徴

- 暗号化は誰でも出来る
  (暗号化鍵は公開されている)

- 復号は秘密鍵を知らないと出来ない
  (もの凄く時間が掛かる)

そんな都合の良い仕組みが本当にあるのか？

公開鍵暗号の例： RSA 暗号

Rivest, Shamir, Adleman (1977)

- **大きな素数 p, q を選び、積 $n = pq$ を作る**
- **n を用いて、公開鍵 e・秘密鍵 d の対を作る**
- **暗号化の計算は n と公開鍵 e とから可能**
- **復号は秘密鍵 d を用いる**
- **n と公開鍵 e とから秘密鍵 d を求めるには、n の素因数分解 $n = pq$ が必要**
- **しかしそれは困難 (膨大な計算時間が掛かる)**

公開鍵暗号の例： RSA 暗号

Rivest, Shamir, Adleman (1977)

- **大きな素数 p, q を選び、積 $n = pq$ を作る**
- **n を用いて、公開鍵 e・秘密鍵 d の対を作る**
- **暗号化の計算は n と公開鍵 e とから可能**
- **復号は秘密鍵 d を用いる**
- **n と公開鍵 e とから秘密鍵 d を求めるには、n の素因数分解 $n = pq$ が必要**
- **しかしそれは困難 (膨大な計算時間が掛かる)**

公開鍵暗号の例： RSA 暗号

Rivest, Shamir, Adleman (1977)

- **大きな素数 p, q を選び、積 $n = pq$ を作る**
- **n を用いて、公開鍵 e・秘密鍵 d の対を作る**
- **暗号化の計算は n と公開鍵 e とから可能**
- **復号は秘密鍵 d を用いる**
- **n と公開鍵 e とから秘密鍵 d を求めるには、n の素因数分解 $n = pq$ が必要**
- **しかしそれは困難 (膨大な計算時間が掛かる)**

公開鍵暗号の例： RSA 暗号

Rivest, Shamir, Adleman (1977)

- 大きな素数 p, q を選び、積 $n = pq$ を作る
- n を用いて、公開鍵 e・秘密鍵 d の対を作る
- 暗号化の計算は n と公開鍵 e とから可能
- 復号は秘密鍵 d を用いる
- n と公開鍵 e とから秘密鍵 d を求めるには、n の**素因数分解** $n = pq$ が必要
- しかしそれは困難 (膨大な計算時間が掛かる)

公開鍵暗号の例： RSA 暗号

Rivest, Shamir, Adleman (1977)

- **大きな素数 p, q を選び、積 $\mathrm{n} = \mathrm{pq}$ を作る**
- **n を用いて、公開鍵 e・秘密鍵 d の対を作る**
- **暗号化の計算は n と公開鍵 e とから可能**
- **復号は秘密鍵 d を用いる**
- **n と公開鍵 e とから秘密鍵 d を求めるには、n の素因数分解 $\mathrm{n} = \mathrm{pq}$ が必要**
- **しかしそれは困難 (膨大な計算時間が掛かる)**

RSA 暗号

- **大きな素数 p, q を選び、積 $n = pq$ を作る**
  - $p - 1$ **と** $q - 1$ **との最小公倍数**
    $l := \mathrm{lcm}(p - 1, q - 1)$ **を求めておく**
- **公開鍵 e・秘密鍵 d の対を作る**
  - l **と互いに素な整数 e を取る**
  - $ed \equiv 1 \pmod{l}$ **なる整数 d を求める**
- **暗号化：** $C \equiv P^e \pmod{n}$
- **復号：** $P \equiv C^d \pmod{n}$

何故これでうまく機能するのか？

RSA 暗号

- **大きな素数 p, q を選び、積 $n = pq$ を作る**
  - $\star$ $p - 1$ **と** $q - 1$ **との最小公倍数** $l := \mathrm{lcm}(p - 1, q - 1)$ **を求めておく**
- **公開鍵 e・秘密鍵 d の対を作る**
  - $\star$ l **と互いに素な整数** e **を取る**
  - $\star$ $ed \equiv 1 \pmod{l}$ **なる整数** d **を求める**
- **暗号化：** $C \equiv P^e \pmod{n}$
- **復号：** $P \equiv C^d \pmod{n}$

何故これでうまく機能するのか？

中国式剰余定理

m, n **が互いに素のとき、**

$$\mathbb{Z}/mn\mathbb{Z} \simeq \mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$$

mn **で割った余りは、**
m **で割った余りと** n **で割った余りとで**
丁度決まる

$$a \equiv b \pmod{mn} \Longleftrightarrow \begin{cases} a \equiv b \pmod{m} \\ a \equiv b \pmod{n} \end{cases}$$

RSA 暗号の検証

- **暗号化：** $C \equiv P^e \pmod{n}$
- **復号：** $P \equiv C^d \pmod{n}$

$$P \equiv (P^e)^d \pmod{n} \text{ **であるか**}$$

---

p, q **は相異なる素数** $\longrightarrow$ **互いに素**

$\longrightarrow n = pq$ **で中国式剰余定理が使える**

$\longrightarrow \bmod p$ **と** $\bmod q$ **とで見れば良い**

Fermat の小定理

p を素数とするとき、

p と互いに素な整数 a に対し、

$$a^{p-1} \equiv 1 \pmod{p}$$

$n = pq, l = \mathrm{lcm}(p-1, q-1), ed \equiv 1 \pmod{l}$ のとき、

$ed \equiv 1 \pmod{(p-1)}$ より $P^{ed} \equiv P^1 \pmod{p}$

$ed \equiv 1 \pmod{(q-1)}$ より $P^{ed} \equiv P^1 \pmod{q}$

併せて、$P^{ed} \equiv P \pmod{n}$

Fermat の小定理

p を素数とするとき、

p と互いに素な整数 a に対し、

$$a^{p-1} \equiv 1 \pmod{p}$$

$n = pq, l = \mathrm{lcm}(p-1, q-1), ed \equiv 1 \pmod{l}$ のとき、

$ed \equiv 1 \pmod{(p-1)}$ より $P^{ed} \equiv P^1 \pmod{p}$

$ed \equiv 1 \pmod{(q-1)}$ より $P^{ed} \equiv P^1 \pmod{q}$

併せて、$P^{ed} \equiv P \pmod{n}$

鍵対の構成

では、l と互いに素な整数 e を与えたとき、

$ed \equiv 1 \pmod{l}$ **なる整数** d
(l を法とした e の “逆数”) は
どうやって求めるのか

逆数の計算を効率良く行なう方法
$\longrightarrow$ Euclid の拡張互除法とその拡張版を用いる

鍵対の構成

では、l と互いに素な整数 e を与えたとき、

$ed \equiv 1 \pmod{l}$ **なる整数** d
(l を法とした e の “逆数”) は
どうやって求めるのか

逆数の計算を効率良く行なう方法
$\longrightarrow$ **Euclid の互除法とその拡張版を用いる**

復習： Euclid の互除法とその拡張版

(1) 互除法により 29 と 17 との最大公約数が 1 であることを求めよ

(2) 互除法の各段階の余り r_i に対し、

$$17x_i + 29y_i = r_i$$

となる整数 x_i, y_i を順次求めよ

(3) $17x + 29y = 1$ となる整数 x, y は何か

(4) 29 を法とした 17 の “逆数”
($17x \equiv 1 \pmod{29}$ となる整数 x) は何か

鍵対の構成

Euclid の互除法を用いることにより、

$ed \equiv 1 \pmod{l}$ **なる整数** d
(l **を法とした** e **の “逆数”) を**
効率良く求めることが出来た

これで RSA 方式が実際に動かせることが判ったが、

では、RSA 暗号は安全な暗号なのか？

鍵対の構成

Euclid の互除法を用いることにより、

$ed \equiv 1 \pmod{l}$ **なる整数** d
(l **を法とした** e **の “逆数”) を**
効率良く求めることが出来た

これで RSA 方式が実際に動かせることが判ったが、

では、RSA 暗号は安全な暗号なのか？

RSA 暗号

- **大きな素数** p, q **を選び、積** $n = pq$ **を作る**
  - $p - 1$ **と** $q - 1$ **との最小公倍数**
    $l := \mathrm{lcm}(p - 1, q - 1)$ **を求めておく**

- **公開鍵** e **・秘密鍵** d **の対を作る**
  - l **と互いに素な整数** e **を取る**
  - $ed \equiv 1 \pmod{l}$ **なる整数** d **を求める**

- **暗号化：** $C \equiv P^e \pmod{n}$

- **復号：** $P \equiv C^d \pmod{n}$

RSA 暗号

- 大きな素数 p, q を選び、積 $n = pq$ を作る
- n を用いて、公開鍵 e・秘密鍵 d の対を作る
  (Euclid の互除法を用いる)
- 暗号化の計算は n と公開鍵 e とから可能
$$C \equiv P^e \pmod{n}$$
- 復号は秘密鍵 d を用いる
$$P \equiv C^d \pmod{n}$$
- n と公開鍵 e とから秘密鍵 d を求めるには、
  n の**素因数分解** $n = pq$ が必要
  (p, q が不明だと l が求まらない)

RSA 暗号の安全性

結局、RSA 暗号の安全性は、

素因数分解問題の (計算量的) 困難さ

に掛かっている

現在の所、
充分速い計算法 (多項式時間アルゴリズム) は
知られていない

⟶ 多くの数学者・計算機科学者が鋭意研究中

RSA 暗号の安全性

結局、RSA 暗号の安全性は、

素因数分解問題の (計算量的) 困難さ

に掛かっている

現在の所、
充分速い計算法 (多項式時間アルゴリズム) は
知られていない

$\longrightarrow$ 多くの数学者・計算機科学者が鋭意研究中

RSA 暗号の安全性

もしも画期的に高速なアルゴリズムを発見したら、

いち早く学会に公表して
人類共有の財産とするであろう

我々科学者の独立によって
情報化社会の安全性が保証されている

RSA 暗号の安全性

もしも画期的に高速なアルゴリズムを発見したら、

いち早く学会に公表して
人類共有の財産とするであろう

我々科学者の独立によって
情報化社会の安全性が保証されている

素因数分解問題の他にも、

離散対数問題 (Discrete Logarithm Problem)

など、

色々な数理現象が、

暗号・符号などの
数理技術・情報技術に応用されていて、

このような基本的な数理技術を組み合わせると、

電子投票方式などの実用システムを
構成することも出来る

まとめ

- 現代の情報化社会を支える基盤技術として
  種々の数理技術が利用されている

- 数理現象の解明が
  直接に技術の進歩に繋がっている

- 人間は弱いもの
  $\longrightarrow$ 不正をしようとしても出来ない
  システムが望まれる
  $\longrightarrow$ “最も確かなもの” としての数理の利用

おしまい