暗号・認証分科会

活動方針

目的:

暗号・認証技術を
一般の人々に
分かりやすく説明する！

活動方針

どういう人に説明する？

- 暗号そのものを良く知らない
- 暗号への認識が古典暗号止まり
- AES・RSAくらいは知ってるが最新動向は良く知らない

などなど…

活動方針

どういう人に説明する？

暗号そのものを
良く知らない

暗号への認識が
古典暗号止まり

AES・RSAくらいは
知ってるが
最新動向は
良く知らない

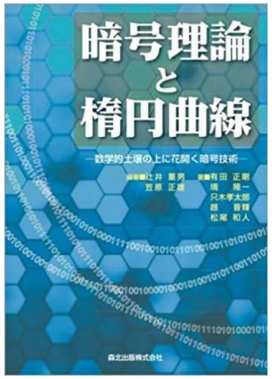
「最新の暗号技術ってどのようなものがあるのか？」について説明

研究キーワード

共通鍵暗号	情報量的安全性	相手認証
公開鍵暗号	計算量的安全性	生体・人工物認証
IDベース暗号	証明可能安全性	ヒューマンクリプト
電子署名	汎用結合可能性	匿名性
ハッシュ関数	暗号実装の安全性	PKI
鍵共有	CRYPTREC	PGP
マルチパーティプロトコル	量子暗号	PAKE

活動メンバー

研究リーダー
趙 晋輝 先生



指導教授
花岡 悟一郎 先生

活動メンバー

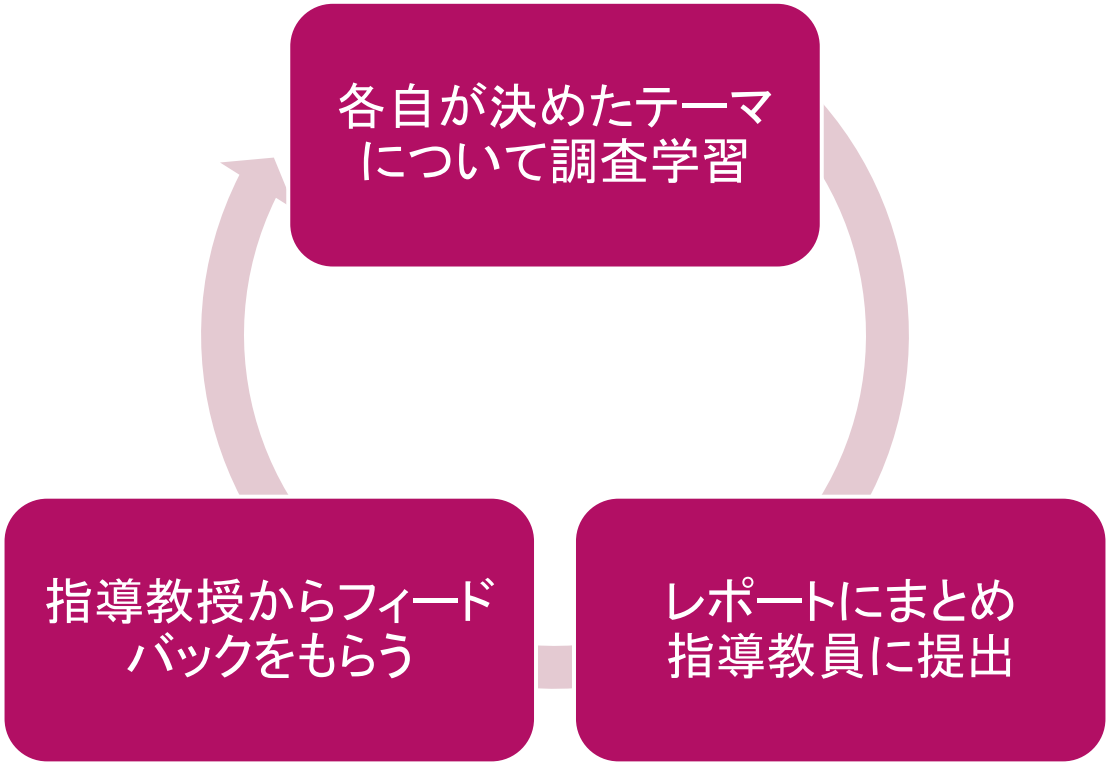
M2	M1	
登丸 尚哉	髙橋 モハマドシャールク	奥村 祐太
西垣 佳亮	小倉 朱門	藤田 真緒
三平 大悟	上里 優介	山本 恭敬

活動内容

暗号技術を分かりやすく説明

まずは我々がその技術を
理解する必要がある！

活動内容



活動テーマ

これらはすべて「先端的」暗号技術！

髙橋 モハマドシャールク	量子暗号
小倉 朱門	格子暗号
上里 優介	プロキシ再暗号化
奥村 祐太	IDベース暗号
藤田 真緒	属性ベース暗号
山本 恭敬	CKKS方式準同型暗号

「先端的」とは？

本活動で考える
「先端的」の
定義について
紹介する。

「非先端的」≠「古典的」

ここでは、RSA暗号や楕円曲線暗号、AESも「非先端的」と考える。

古典暗号	現代暗号	先端的暗号
シーザー暗号	RSA暗号	格子暗号
ヴィジュネル暗号	AES	IDベース暗号

「先端的」と「非先端的」の違いは？

従来の安全性・機能性

新しいテクノロジー

より高い
安全性が
欲しい！

より高い
機能性が
欲しい！

これらの要望に応えるべく考案されたのが「**先端的**」暗号技術

「新しいテクノロジー」とは？

もちろん沢山あるが、我々が取り上げたのは…

- 量子コンピュータ
- クラウドコンピューティング

量子コンピュータと先端的暗号技術

- 量子コンピュータによってどのような要望が生まれたのか？
- その要望に応じてどのような技術が考案されたのか？

量子コンピュータと耐量子性

- 量子コンピュータによって
  従来の暗号の安全性が脅かされる恐れがある

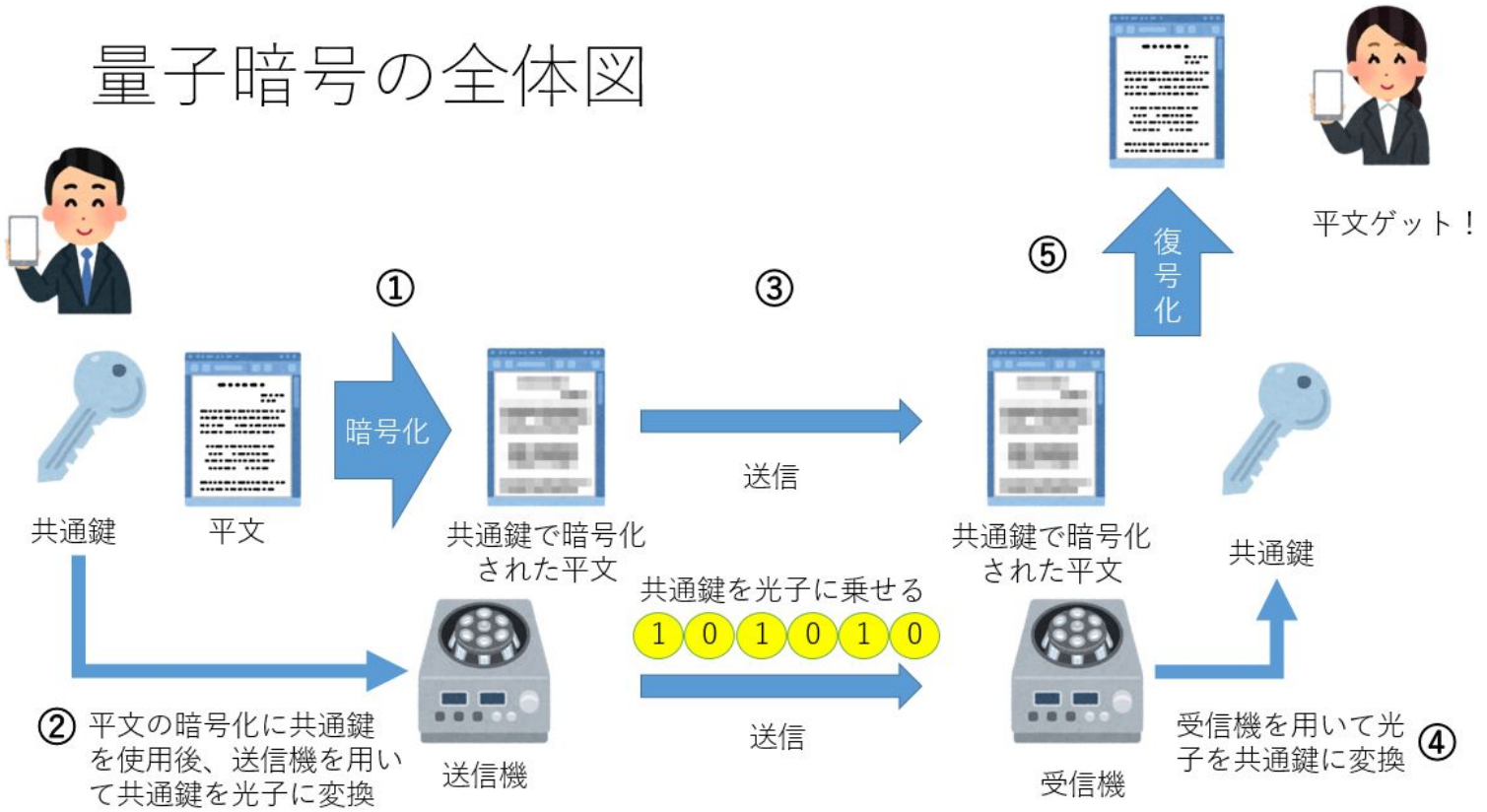
- **量子コンピュータでも解けない暗号**が欲しい！
  - そのような暗号を耐量子暗号という。

髙橋 モハマドシャールク	量子暗号
小倉 朱門	格子暗号

量子暗号

- 量子暗号は，通信を盗聴しようとするどんな盗聴者に対しても盗聴を防止できる安全な暗号方式
- 量子暗号は量子鍵配送、ワンタイムパッドで成り立っている
  - 量子鍵配送：共通鍵を安全に相手に送信するための仕組み
  - ワンタイムパッド：情報理論的に安全であることが証明された暗号技術

量子暗号


量子暗号の全体図
①
暗号化
共通鍵
平文
共通鍵で暗号化
された平文
②
平文の暗号化に共通鍵
を使用後、送信機を用い
て共通鍵を光子に変換
送信機
③
送信
共通鍵を光子に乗せる
1 0 1 0 1 0
送信
⑤
復号化
共通鍵で暗号化
された平文
受信機
受信機を用いて光
子を共通鍵に変換
④
共通鍵
平文ゲット！

格子暗号

- 暗号化に格子問題を利用した暗号

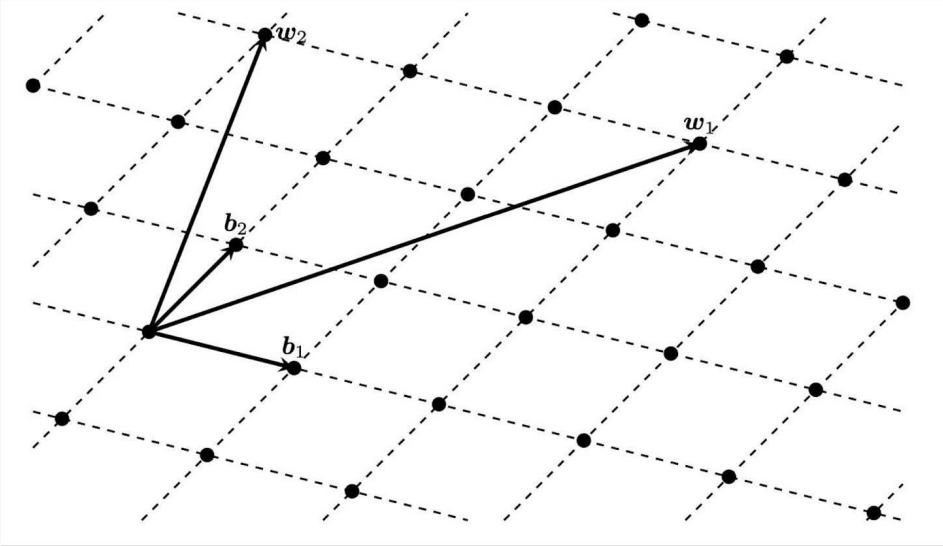
  格子とは…基底を整数倍してたし合わせてできる点の集合

n個の線形独立なベクトル：

$b_i \in \mathbb{R}^n$

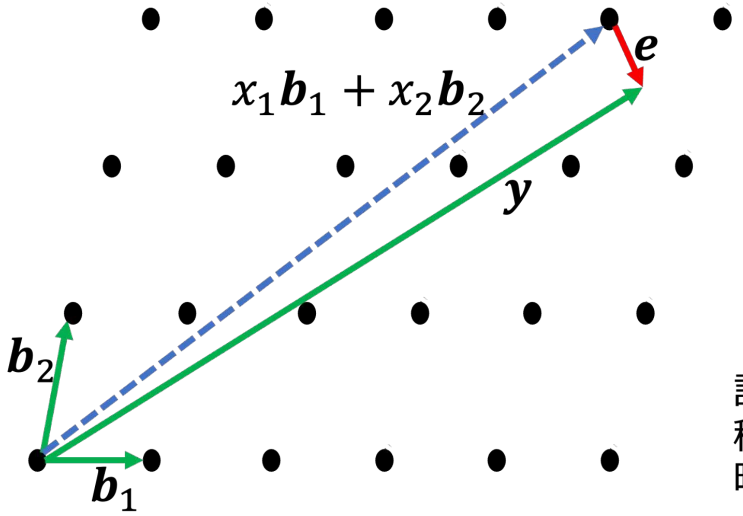
$(i = 1, 2, \cdots, n)$

$\mathcal{L} = \{\sum_{i=1}^{n} a_i b_i \,|a_i \in \mathbb{Z}\}$



格子暗号

- 格子暗号の構成
  公開鍵：基底$\boldsymbol{B}$, $\boldsymbol{y}$，秘密鍵：$\boldsymbol{x}$



$$\boldsymbol{y} = \boldsymbol{B}\boldsymbol{x} + \boldsymbol{e}$$

public key　secret key　error

誤差を含む連立方程式を解いて
秘密鍵：$\boldsymbol{x}$を計算することで
暗号を解読可能

- NP困難に分類され，量子コンピュータでも解くことは難しい
- NIST(National Institute of Standards and Technology)による標準化も進む

クラウドコンピューティングと先端的暗号技術

- クラウドによってどのような要望が生まれたのか？
- その要望に応じてどのような技術が考案されたのか？

クラウドサービスと暗号

- 多くの情報がクラウドサービスに集まってきている
- それらの情報が安全に扱われる保証はあるのか？
  - ベンダーが誤って秘密情報をばらまいたりしないか？
  - 内部処理で秘密情報をそのまま扱うのは潜在的に危険では？

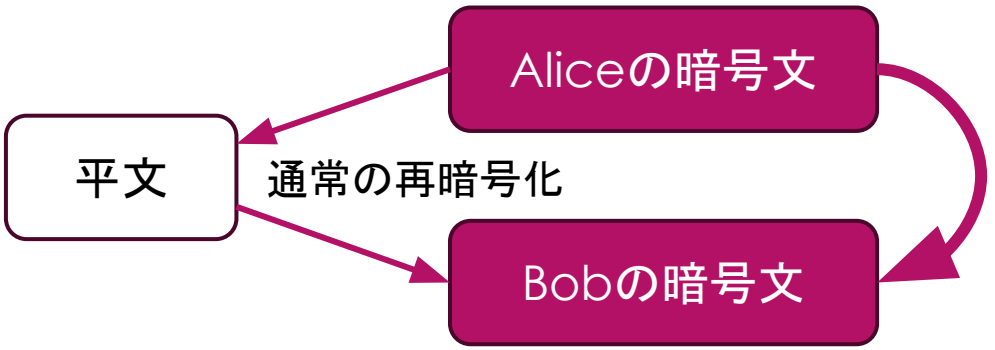
- 安全性(とプライバシー)と機能性を両立させたい！

クラウドサービスと暗号

- 安全性(とプライバシー)と機能性を両立させたい！

上里 優介	プロキシ再暗号化
奥村 祐太	IDベース暗号
藤田 真緒	属性ベース暗号
山本 恭敬	CKKS方式準同型暗号

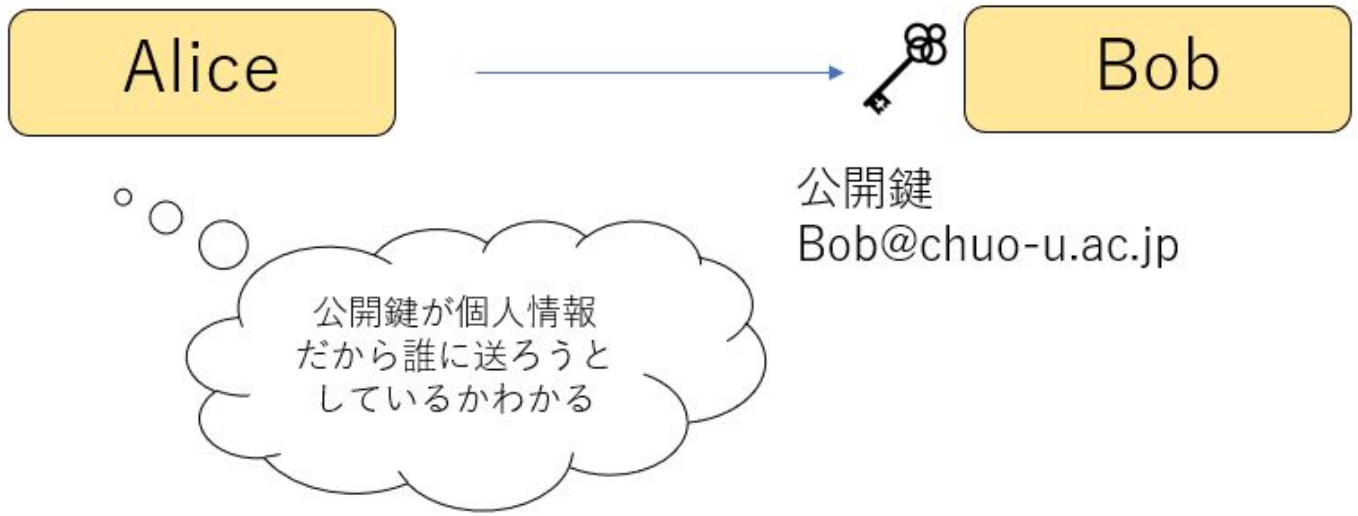
プロキシ再暗号化



- 平文を介さない → 再暗号化の過程で**平文が漏洩しない**
- 再暗号化鍵は公開可能 (公開しても秘密情報は得られない)
  → **第三者が再暗号化可能** → 処理負担の制御が可能
- 再暗号化可能 ⇔ 再暗号化鍵が存在する
  → **非中央集権的なアクセス制御**が実現できる

具体例: BBS暗号、AFGH暗号

IDベース暗号

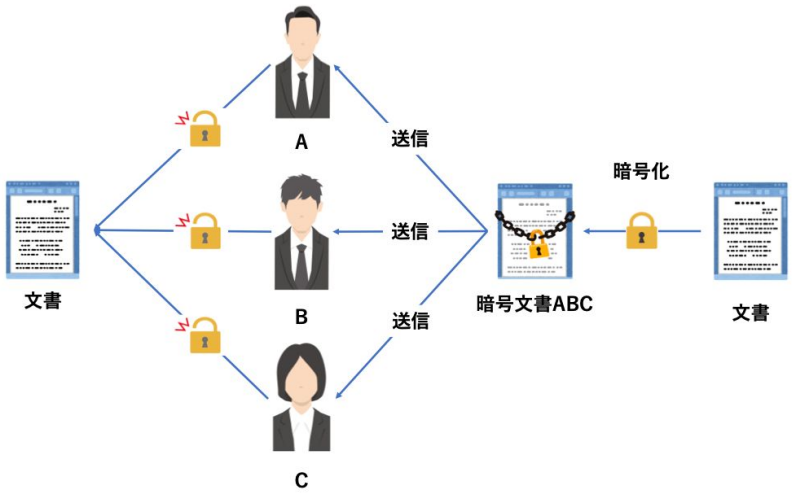


- 従来の公開鍵暗号は公開鍵では相手が誰かわからない
  → **公開鍵証明書認証局(CA)に問い合わせて本人認証**
- IDベース暗号では公開鍵が個人情報
  → **公開鍵の認証が必要ない暗号方式**の実現

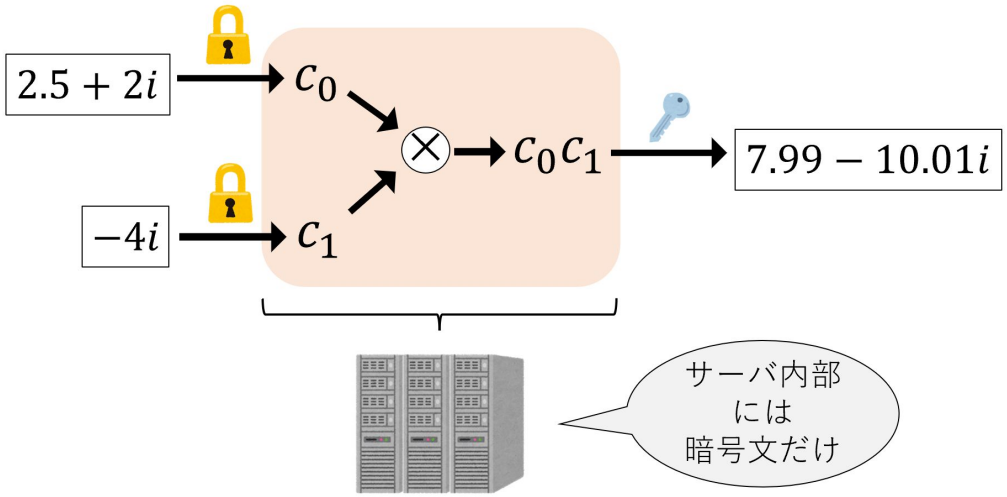
属性ベース暗号

- IDベース暗号に基づいた暗号で、IDを属性の集合として捉えることで詳細なアクセス制御を組み込んだ暗号
- 属性を指定して、文書の暗号化ができる
  →会社などの人数の多い組織におけるアクセス制御が容易
- CP-ABE：暗号文に条件を、
  秘密鍵に属性を埋め込む方式

  KP-ABE ：暗号文に属性を、
  秘密鍵に条件を埋め込む方式



CKKS方式準同型暗号



- 準同型暗号とは
- 暗号文のまま足し算や掛け算ができる
- サーバーに生データを置く必要がない
  → 情報漏洩のリスク低減

- CKKS方式準同型暗号 [Cheon+17]
- 複素数の暗号化：より実用的に
- 安全性：RLWE仮定による耐量子計算機性
- rescailing：乗算時のノイズを抑制

まとめ

- 暗号の役割は単なる暗号化，復号に留まらない
- 求められる安全性は変化し，より高い機能性を求められる

付録：暗号技術の紹介スライド

以下のスライドは、各メンバーが
活動を通じて作成した暗号技術の紹介スライドです。

量子暗号

背景

- 近年量子コンピュータの実用化に向けて開発が進められています。
- 量子コンピュータの処理速度は現代のコンピュータよりも高速であるため、RSA 暗号などが現実的な時間で破られる恐れが出てきました。

量子暗号とは

- 量子暗号は，通信を盗聴しようとするどんな盗聴者に対しても盗聴を防止できる安全な暗号方式
- 具体的に言うと、量子鍵配送によって事前に暗号鍵を伝送共有し、ワンタイムパッドと呼ばれる暗号方式で通信を暗号化すること

量子暗号が無いと・・・

- どんな暗号化された通信も攻撃者が解読できてしまう

RSA暗号の鍵の長さが2048bit（デジタル署名（なりすましを防ぐために使われる）の場合）であるとき、最高性能のスーパーコンピューターで解読に1億年以上かかると言われています

ですが、（十分な性能を持つ）量子コンピュータでは数時間で4096bitの鍵を解読することができてしまうと言われています

※数時間で解くためにショアのアルゴリズムというのを使う必要があります

通信が解読されることで起きうること

- クレジットカードの番号が流出してしまい、攻撃者に利用される
- 企業の機密情報を漏洩させ、企業が倒産に追い込まれる
- 相手国の防衛戦略を入手し、弱点を攻撃し侵略

・・・などの恐ろしい出来事が起こると言えます

自分には関係ない

- スライドで挙げた「RSA暗号」が使われている場面

SSL/TLS サーバー証明書の「ドメイン認証」

（そのほかにもメール送信者の認証に使われるS/MIME証明書、電子契約時の電子署名にも使われています）

自分には関係ない

- SSL/TLS サーバー証明書の「ドメイン認証」

SSL/TLS サーバー証明書とは

ウェブサイトにアクセスする際に行われる通信の暗号化、

ウェブサイトがなりすまされていないかのチェック（ドメイン認証）に使われます

国内上場企業における常時SSL化対応サイトは **91.4%** ！

→ RSA暗号が今日の社会に浸透していることがわかります

※常時SSL化 調査レポート 上場企業サイト対応状況（2023年10月版），
https://www.feedtailor.jp/report_aossl/

量子暗号の全体図

※補足
共通鍵：平文を暗号化、復号化をするのに用いられる鍵のこと
平文：文章のこと（メールと考えるとわかりやすいですね！）
暗号化：平文を他人がわからないようにすること
復号化：暗号化された平文をもとの平文に戻すこと

平文ゲット！

復号化

共通鍵

平文

暗号化

共通鍵で暗号化
された平文

送信

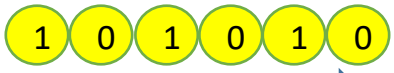
共通鍵で暗号
化された平文

共通鍵

送信機

平文の暗号化に共通鍵
を使用後、送信機を用
い
て共通鍵を光子に変換

共通鍵を光子に乗せる



送信

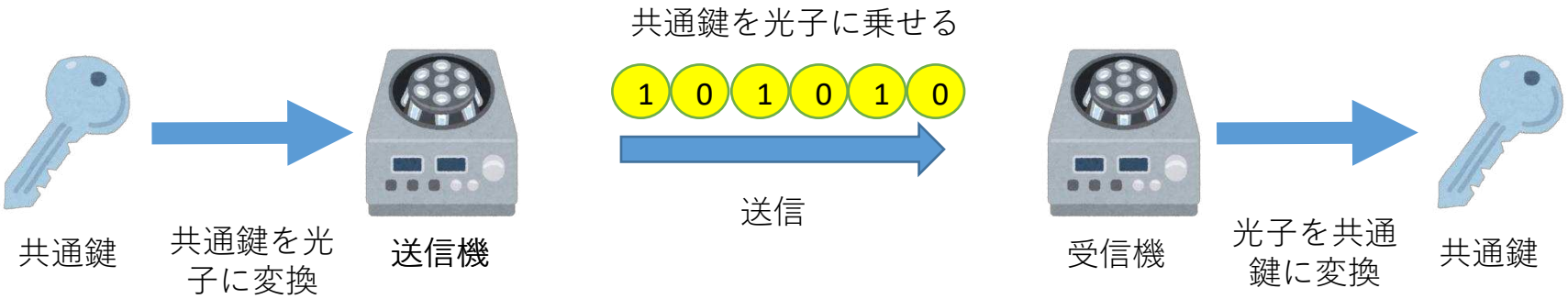
受信機

受信機を用いて
光子を共通鍵に
変換

量子暗号とは

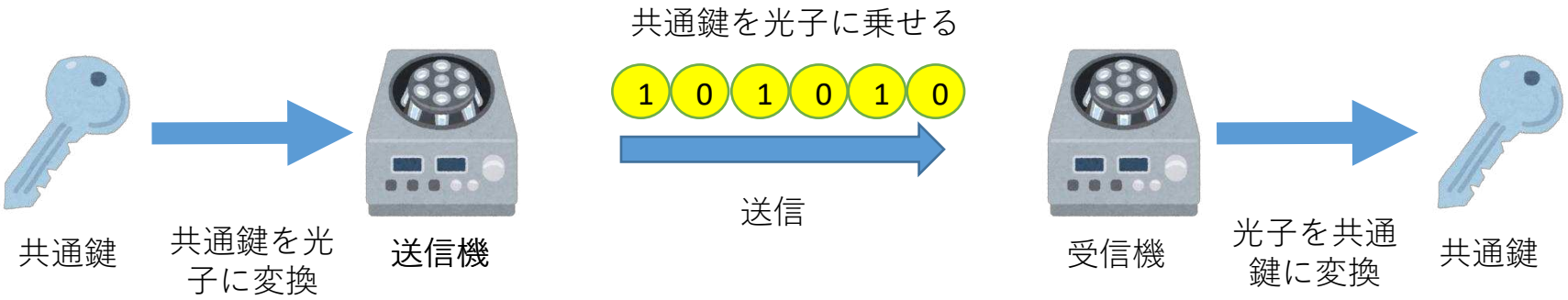
- 量子暗号は，通信を盗聴しようとするどんな盗聴者に対しても盗聴を防止できる安全な暗号方式
- 具体的に言うと、**量子鍵配送**によって事前に暗号鍵を伝送共有し、ワンタイムパッドと呼ばれる暗号方式で通信を暗号化すること

量子鍵配送の具体的説明



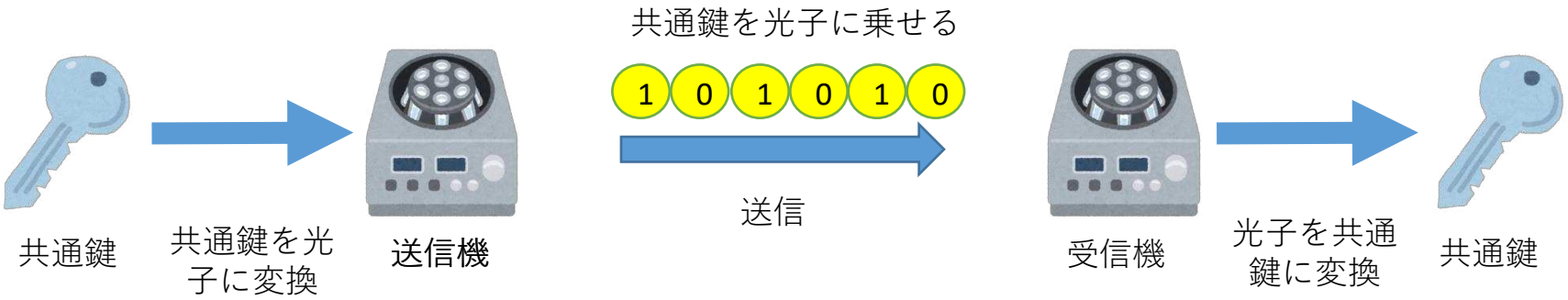
- 量子鍵配送は共通鍵を安全に相手に送信するための仕組み
- 量子鍵配送の手順
  - 送信機は送信方法を、受信機は受信方法をランダムに選択します
  - 送信機は共通鍵の情報を1ビットずつエンコードし、送信機が送信を行い, 受信機が受信します
  - (1),(2) を繰り返し, 送信機が鍵のすべての情報（ビット）の送信を終えた後、送信機、受信機が事前に選択した送信方法, 受信方法を開示します
  - 送信機の送信方法と受信機の受信方法が一致すれば終了し, 一致しなければ再度 (1) に戻ります

量子鍵配送の具体的説明



- 注意点
  - 送信および受信方法は2種類あり、送信機と受信機はその中からランダムに選択します

量子鍵配送の安全性



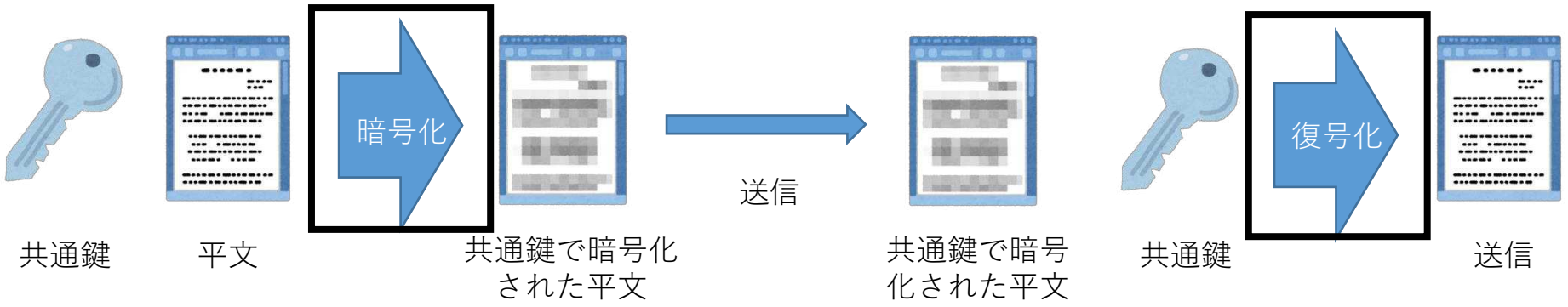
- 3つの観点で安全性を考えてみます

①盗聴の検知はできるのか？
　光子の性質を用いて可能。光子は抜き取られた場合、光子の総数が変化するため検知が可能。
　また光子を盗聴すると光子の状態が変化するため検知が可能。
②そもそも盗聴できるのか？
　盗聴は可能ですが、盗聴の検知が可能です。そのため結果として盗聴されてない光子の送信が可能。
③光子の改ざんは可能なのか？
　理論上可能です。ですが、盗聴者は検知回避のためすべての光子の送信方法を知っておく必要があり、それを知ることは現実的に不可能と言えます。

量子暗号とは

- 量子暗号は，通信を盗聴しようとするどんな盗聴者に対しても盗聴を防止できる安全な暗号方式
- 具体的に言うと、量子鍵配送によって事前に暗号鍵を伝送共有し、**ワンタイムパッド**と呼ばれる暗号方式で通信を暗号化すること

ワンタイムパッドの具体的説明



- ワンタイムパッド（バーナム暗号とも呼ばれる）は、上記の図で言うと暗号化、復号化で使われる暗号方式です。

ワンタイムパッドの具体的説明

- ワンタイムパッドは排他的論理和（XOR）を用いて平文を暗号化する

A	B	A XOR B
0	0	0
1	0	1
0	1	1
1	1	0

排他的論理和（XOR）：AとBが与えられた時AとBが等しければ0、そうでなければ1となる演算のこと

※コンピュータは0と1しか扱うことができないため入力は0,1のいずれかとなります

ワンタイムパッドの具体的説明

- Aを平文、Bを共通鍵とし、AとBの排他的論理和を計算すると暗号文が得られます

平文	共通鍵	暗号文
0	0	0
1	0	1
0	1	1
1	1	0

ワンタイムパッドの具体的説明

- 平文は共通鍵と暗号文の排他的論理和を計算することで得られます

暗号文	共通鍵	平文
0	0	0
1	0	1
1	1	0
0	1	1

ワンタイムパッドの安全性

- **情報理論的に安全**であることが証明されている
- 情報理論的に安全であるとは？

「攻撃者が暗号文 を入手している場合としていない場合で平文に関して得られる情報量が等しい」ことを指している

→ 攻撃者がどんなに頑張っても暗号文から得られる手掛かりがゼロ

ワンタイムパッドって本当に安全？

- 「情報理論的に安全であること」と「暗号解読」の関係

暗号解読とは「総当たり攻撃よりも,速く（効率的に）解読すること」を指します

※ 総当たり攻撃とは考えられる答えをすべて試す攻撃

例）0-9までの数字が用いられる4桁の暗証番号

この場合総当たり攻撃すると

10（通り）× 10 × 10 × 10 = 10000

回試す必要があります

ワンタイムパッドって本当に安全？

- 「情報理論的に安全であること」と「暗号解読」の関係

ここまでを整理すると

①暗号解読とは総当たり攻撃よりも効率的に解読すること

②情報理論的に安全＝「総当たり攻撃」以外に解読する方法はない

→つまり情報理論的に安全であれば暗号解読はできない！

まとめ

- 量子暗号は非常に強力な暗号技術である
- 量子暗号が実用化されればみんなが安心して情報のやり取りができるようになる
- これを機に量子暗号に興味を持っていただければ嬉しいです

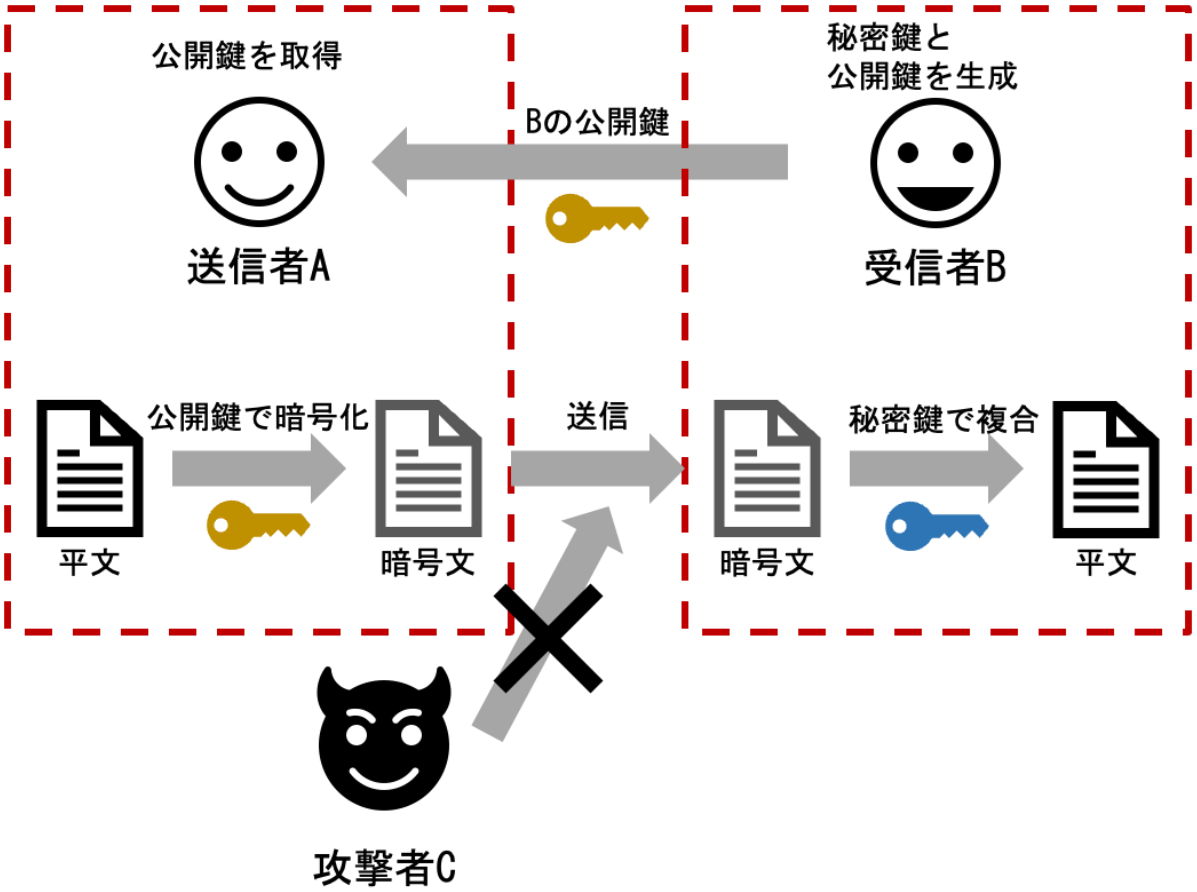
格子暗号

目次

- 現代暗号技術
- 量子コンピュータ
- 耐量子計算機暗号
- 格子暗号

➢ 公開鍵暗号方式

- 公開鍵eと秘密鍵dに暗号化と復号
- 秘密鍵を持つBは解読可能
- 秘密鍵を知らない攻撃者は解読不可能



RSA暗号

➢ RSA暗号

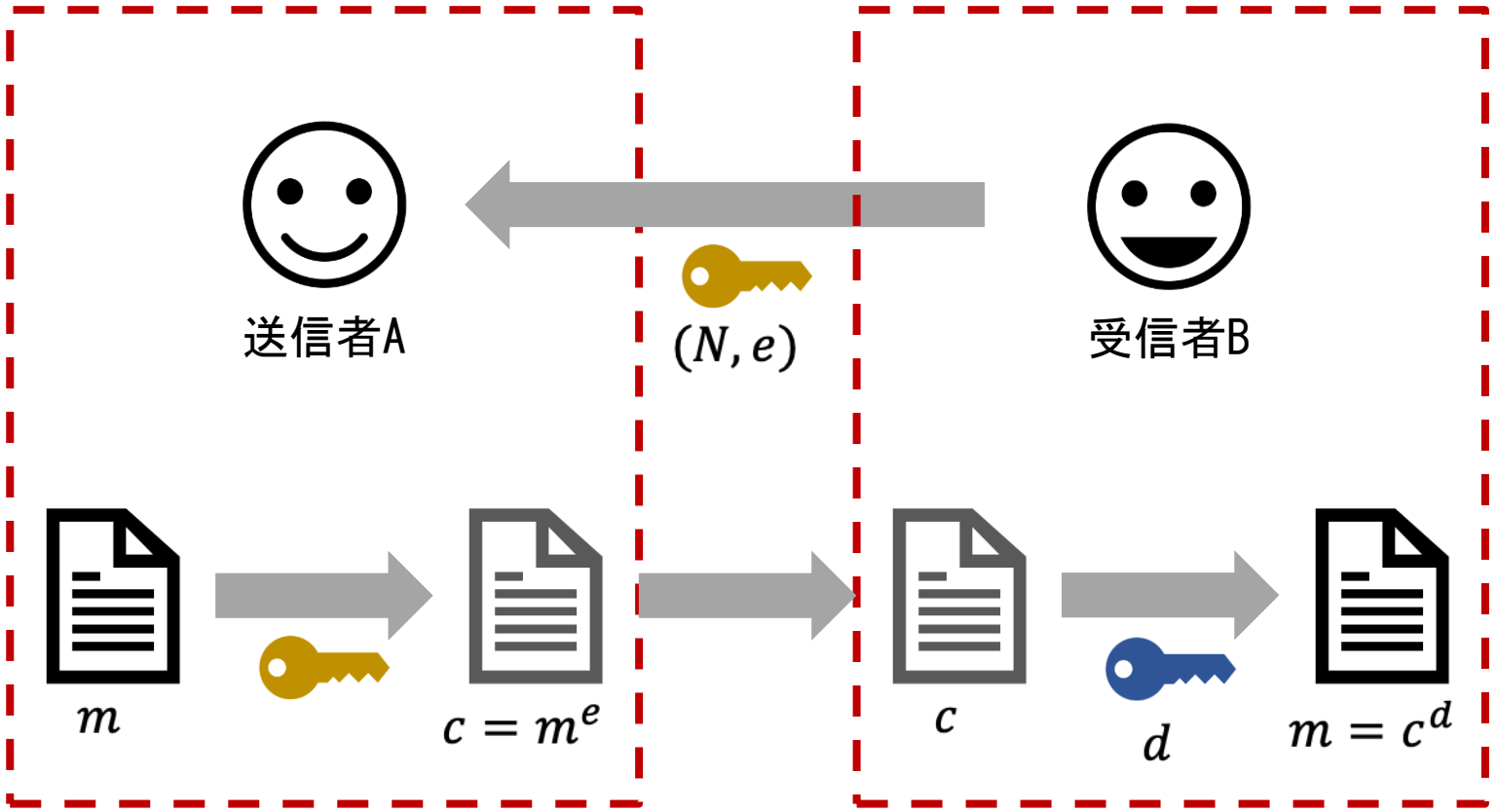
- 広く普及している公開鍵暗号の一つ
- 素因数分解の困難性が根拠
- 自然数Nは公開．$N = pq$の素因数p, qは非公開
- 公開鍵と秘密鍵には以下を満たす自然数e, dを使用

$$\gcd\big(e, (p-1)(q-1)\big) = 1$$
$$ed = 1 \pmod{(p-1)(q-1)}$$

- 平文は$m \in N\mathbb{Z}$

RSA暗号の仕組み

- 暗号化：$c = m^e \pmod N$
- 復号：$c^d = m^{ed} = m \pmod N$
- 復号の妥当性はEulerの定理より認められる



RSA暗号の安全性

- RSA暗号への攻撃方法

①公開鍵から秘密鍵を計算（完全解読）
②自然数Nを素因分解する
③公開鍵，暗号文から平文を計算（一方向性の解読）

②から①，③が計算可能
③は未解決問題

素因数p, qを求めることでRSAが破れる

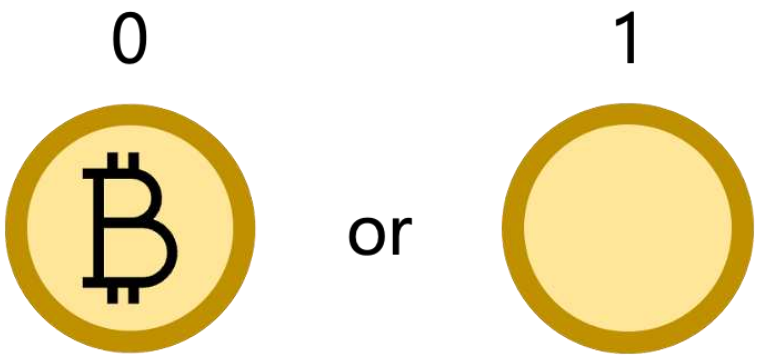
しかし，素因数分解は現実的な時間で計算できない

素因数分解の困難性が
RSA暗号の安全性の根拠

量子コンピュータ

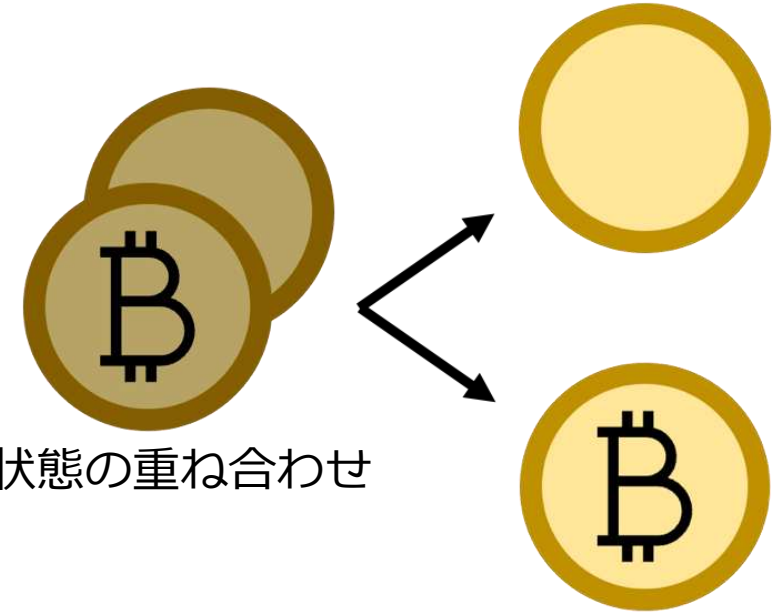
- 通常のコンピュータ
  - 2つの状態を0, 1によって表現
- 量子コンピュータ
  - 観測前は2つの状態の重ね合わせ．観測によって状態を決定

通常のコンピュータ



どちらかが確定

量子コンピュータ



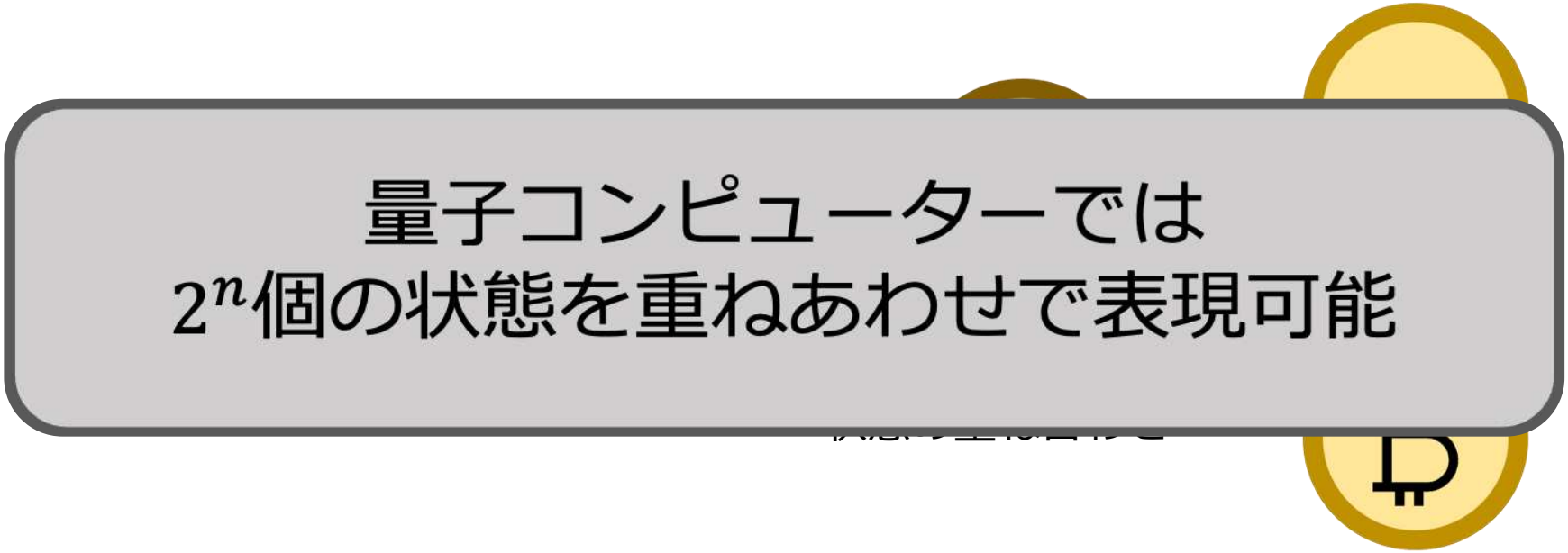
観測で確定

量子コンピュータ

- 通常のコンピュータ
  - 2つの状態を0, 1によって表現
- 量子コンピュータ
  - 観測前は2つの状態の重ね合わせ．観測によって状態を決定

通常のコンピュータ

量子コンピュータ



どちらかが確定

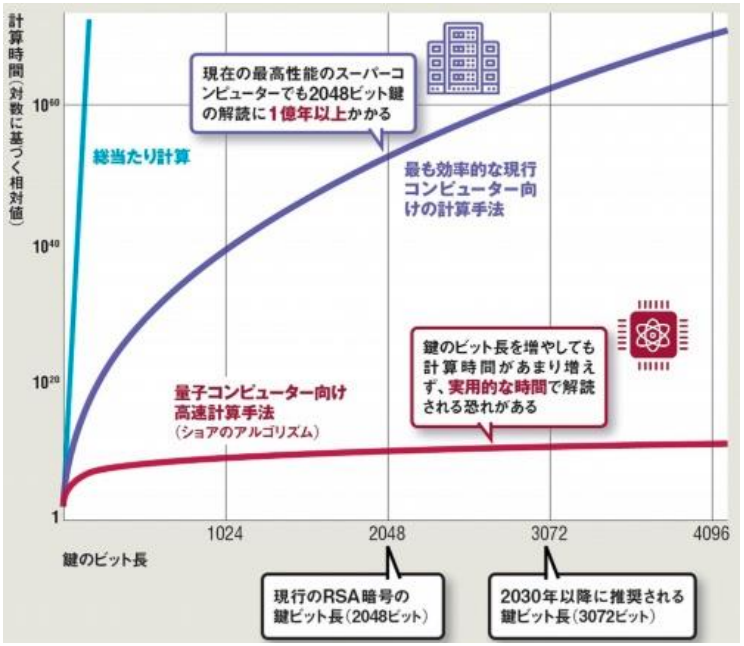
観測で確定

RSA暗号の高速解読

- ショアのアルゴリズム

- 量子フーリエ変換を利用したアルゴリズム.
  高速で素因数分解可能が可能になる
- スーパーコンピュータでも
  １億年以上かかる計算を
  実用的な時間で計算可能に

**将来的にRSA暗号の
安全性が破られる**



https://xtech.nikkei.com/atcl/nxt/column/18/01191/012900001/

RSA暗号の高速解読による影響

➢ 情報漏洩

- 暗号通信が完全に覗ける
- インターネットユーザは約44億人．そのすべてのキャッシュカード情報などが晒せる
- 情報漏洩による損害賠償の見積もりは１人あたり約2万3000円

➢ 電子署名の偽造

- RSAを利用した署名
- 多くのドメイン認証に利用
- 秘密鍵が流出した場合，SSL証明書が失効
  →多くのサービスが利用できなくなる

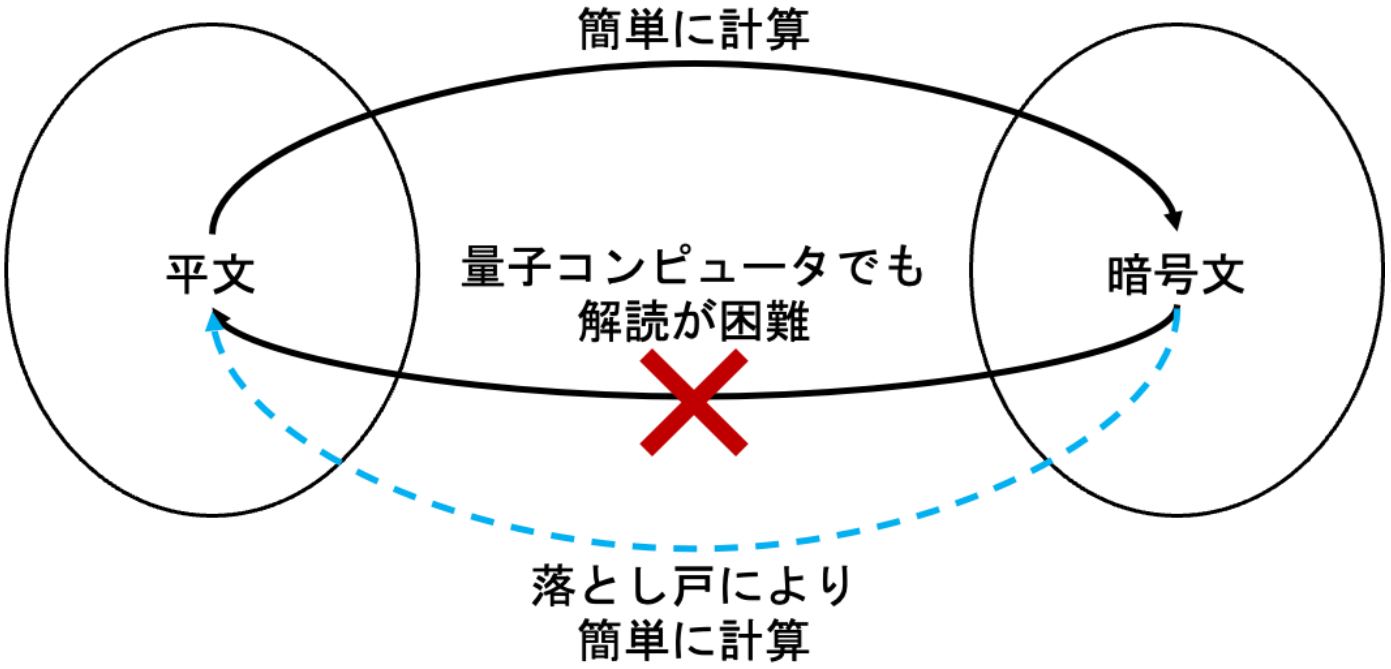
耐量子計算機暗号

➢ 耐量子計算機暗号

- 量子コンピュータでも解読困難な数学的な問題

➢ 求められる機能

- 平文が高速に暗号化
- 落とし戸により暗号文を高速で復元
- 鍵，暗号文のサイズが小さい



耐量子計算機暗号

□ 格子暗号
格子の問題の困難性に基づいた暗号

□ 多変数多項式暗号
多変数多項式を利用した暗号

□ 符号暗号
誤り訂正符号を利用した暗号

□ 同種写像暗号
楕円曲線における同種写像に基づく暗号

NP困難な問題のワーストケースへの帰着により高い安全性

格子暗号は最も注目されている

格子暗号

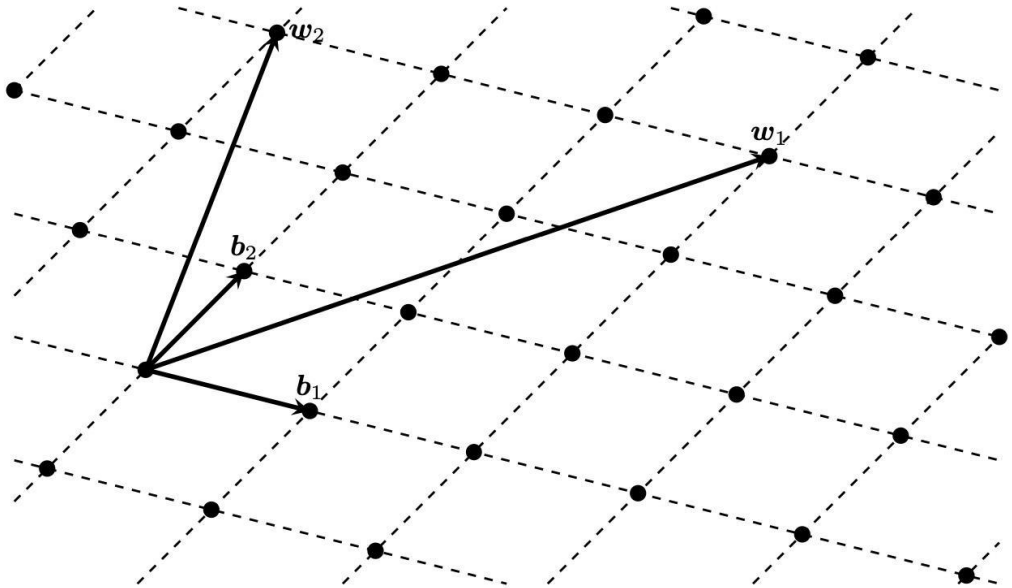
➢ 格子暗号

- 格子問題に基づく暗号
- 安全性はSVP/CVPの困難性に基づく

➢ 格子

- nこの線型独立なベクトル$\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_n$の線形和の集合

$$\mathcal{L} = \{\sum a_i \boldsymbol{b}_i \,|a_i \in \mathbb{Z}^n\}$$



LWE問題

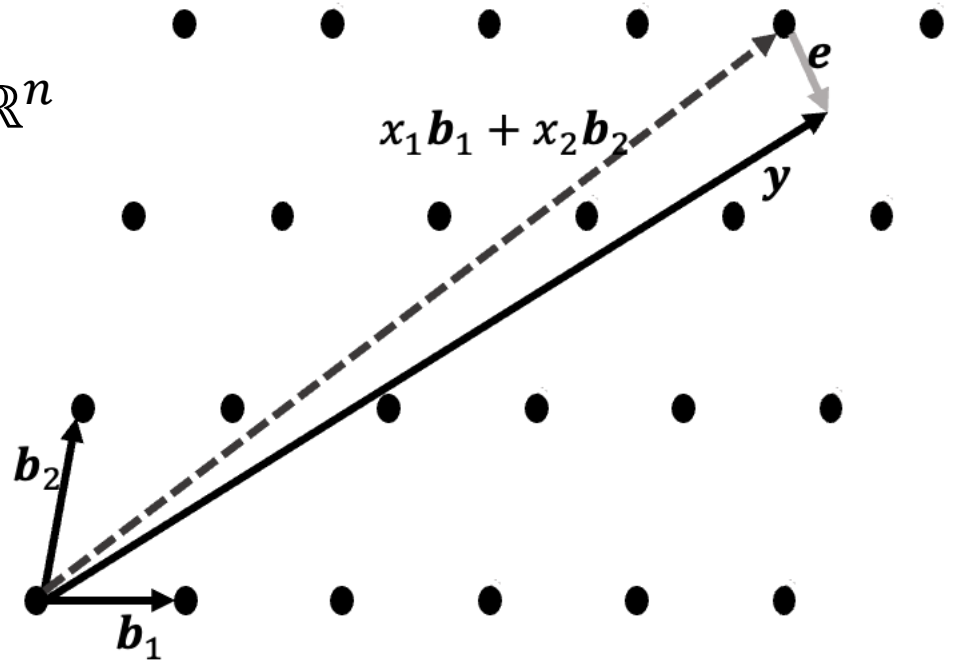
- Learning with Error (LWE)問題
- 微小なエラー$\boldsymbol{e}$を含む$\boldsymbol{y}$から$\boldsymbol{x}$を求める問題
- CVPやSVPに帰着されるため，LWEによる暗号構成ではそれらの困難性を安全性の根拠とする

$$B = [\boldsymbol{b}_1\ \boldsymbol{b}_2\ \cdots\ \boldsymbol{b}_n], \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{e} \in \mathbb{R}^n$$

$$\boldsymbol{y} = B\boldsymbol{x} + \boldsymbol{e}$$

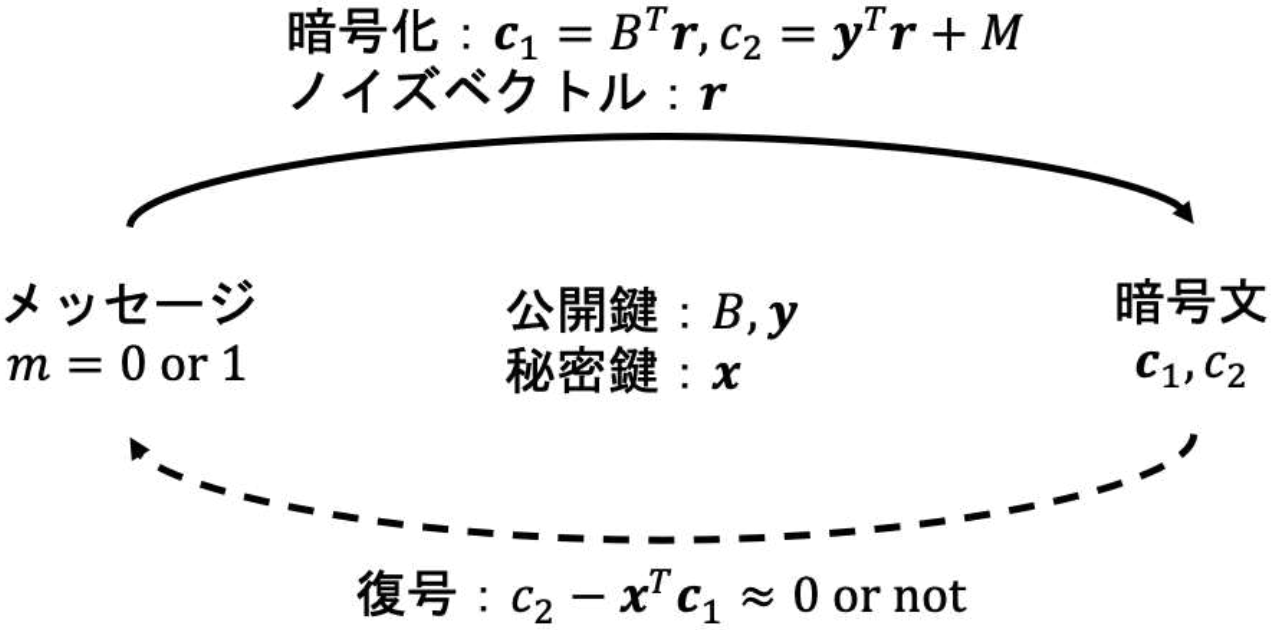
$$\downarrow$$

$$\boldsymbol{x} = ?$$



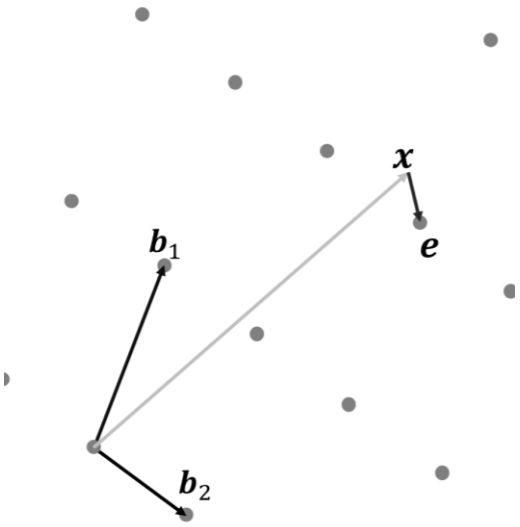
格子暗号の構成

- 公開鍵は基底Bとベクトル$\boldsymbol{y} = B\boldsymbol{x} + \boldsymbol{e}$、秘密鍵は$\boldsymbol{x}$
- 平文をm，ノイズを$\boldsymbol{r}$とする
- 暗号化：$\boldsymbol{c}_1 = B^T\boldsymbol{r}, c_2 = \boldsymbol{y}^T\boldsymbol{r} + m$
- 復号：$c_2 - \boldsymbol{x}^T\boldsymbol{c}_1 = \boldsymbol{e}^T\boldsymbol{r} + m$

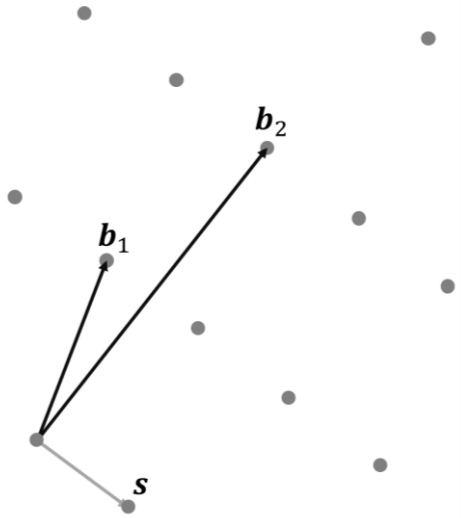


SVP/CVP

- 最近ベクトル問題（CVP）
  - 位置$\boldsymbol{x}$から最も**近い**格子点を求める問題

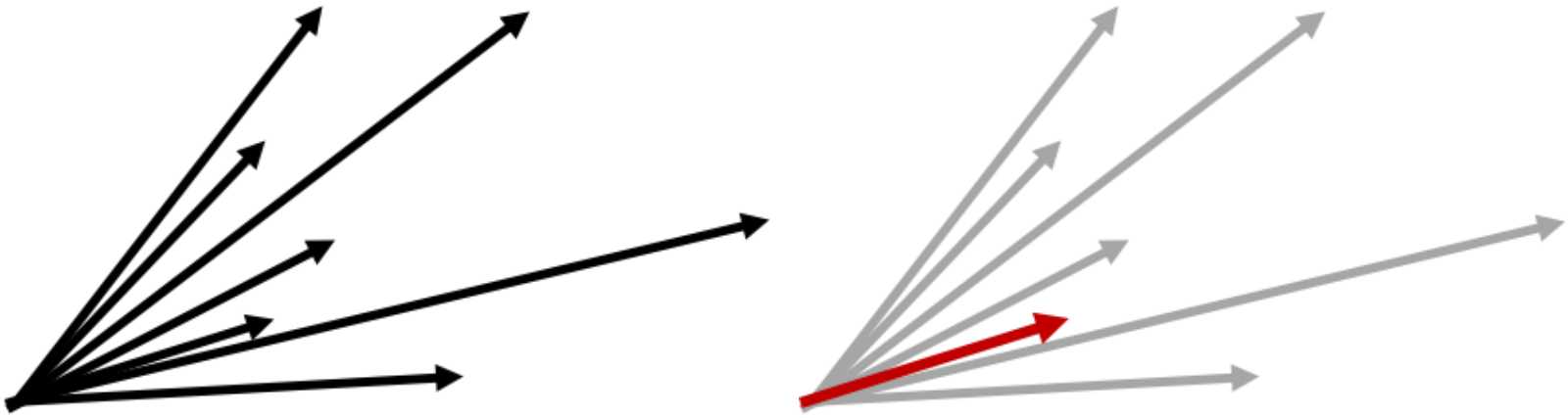


- 最短ベクトル問題（SVP）
  - $\boldsymbol{s} \in \mathcal{L}$のうち**最小**となるベクトル$\boldsymbol{s}$を求める問題



CVP/SVP

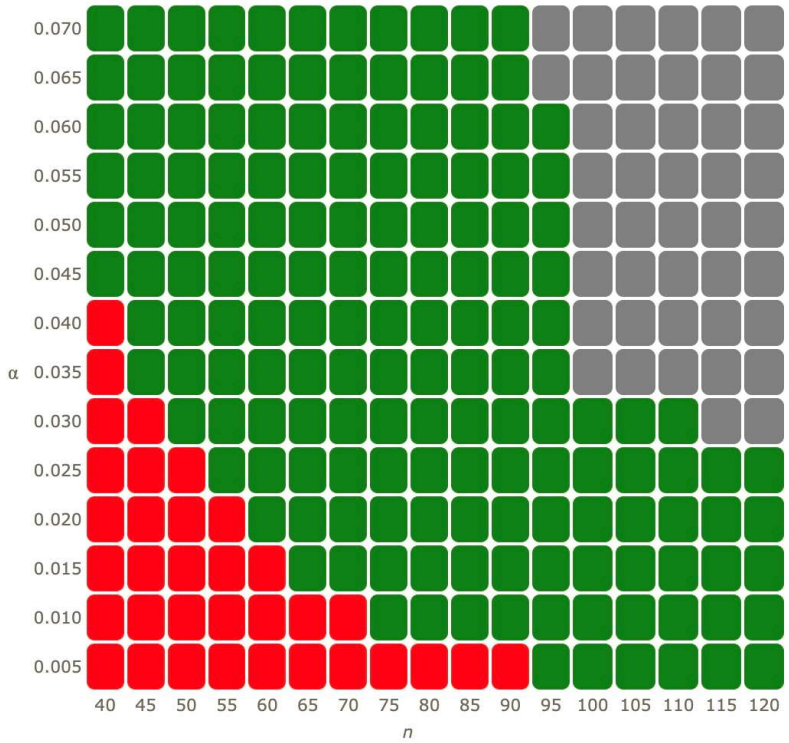
- 総当たりでの計算
- NP困難であることが知られている



無数にあるベクトルの中から，ベクトルの大きさを
計算して最も短いものを探すのは時間がかかる

LWEチャレンジ

- ダルムシュタット工科大学主催のチャレンジ
- 耐量子計算機暗号の標準化にむけた安全な暗号化のパラメータ選定の技術的な根拠として利用



n is dimension

α is the relative error size

解決済み

未解決

11月15日現在の解読状況

https://www.latticechallenge.org/lwe_challenge/challenge.php

格子暗号の標準化

- NIST により2017年から耐量子計算機暗号の標準化に向けた動き
- 2022年7月時点での標準化状況

➢ PKE/KEM

-CRYSTALS-KYBER (格子ベースKEM)

➢ Signature

-CRYSTALS-DILITHIUM (格子ベース署名)

-FALCON (格子ベース署名)

-SPHINCS (ハッシュベース署名)

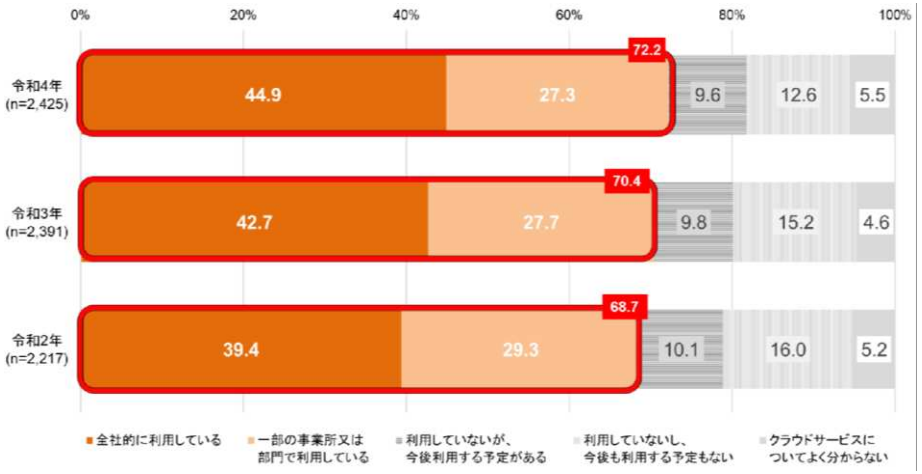
まとめ

- RSA暗号は素因数分解の困難性を安全性の根拠とした暗号方式
- 量子コンピュータによってRSA暗号が高速で解読可能
- 耐量子計算機暗号として格子暗号に注目
- 耐量子計算機暗号の標準化に向けた動向

プロキシ再暗号化について

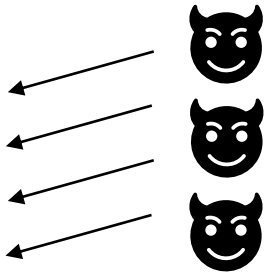
背景① クラウドサービスは便利

- コスト削減・生産性向上につながりうる
- 場所の制約を受けない
- 広く利用されている
- 利用者数も増加傾向にある



クラウドサービスの利用状況(総務省, 令和4年)

背景② クラウドサービスは安全か？



多数の利用者
↓
攻撃対象に
なりやすい

機密情報の流出

プライバシー上の問題も

背景③ クラウドサービスは安全か？

クラウドサービスに限らないが，サーバやサービス事業社は必ずしも**信頼できるとはいえない**

繰り返されるフェイスブックの個人データ流出問題

#木内 登英

2018/12/21

Share

プラットフォーマー規制元年

5月に欧州連合（EU）で個人データの保護強化を狙った一般データ保護規則（GDPR）が導入されるなど、GAFA（グーグル、アップル、フェイスブック、アマゾン）に代表される巨大デジタル・プラットフォーマーを規制する動きが、2018年に世界で強まった。まさに、プラットフォーマー規制元年であった。

規制強化の動きを加速させたきかっけの一つは、2018年3月に発覚した、フェイスブックの個人データ漏洩事件だ。同社が管理する利用者データ最大8,700万人分が、英コンサルティング会社のケンブリッジ・アナリティカに流れ、それが政治利用されていたことが明らかとなった。利用者の友人のデータまで利用されていたことも分かり、同社は強い批判にさらされた。

2018年の年末にかけて、フェイスブックの個人データ漏洩問題がまた増えている。同社は12月14日、プログラムの欠陥が原因で、外部のアプリ開発者が最大680万人の利用者の写真を共有できる状態が生じたと発表した。欠陥は修復されたが、9月13日から25日の間は、最大1,500のアプリが利用者の写真を入手できる状態にあり、写真が外部に流出した恐れがある。利用者がアプリ開発者に写真へのアクセスを許可した場合、本来は自身の投稿が時系列で表示される「タイムライン」で共有された写真のみが対象だが、プログラムの欠陥によって、まだ投稿していなかったサイト内の写真などがアクセスされた疑いがあるという。

出典: https://www.nri.com/jp/knowledge/blog/lst/2018/fis/kiuchi/1221_2

富士通クラウドテクノロジーズ、脆弱性を突いた不正アクセスを確認

BY DLP · 公開済み 2022年5月16日 · 更新済み 2022年6月28日

- 元記事：ロードバランサーへの不正アクセスについて
- HP：富士通クラウドテクノロジーズ

発表日時2022/5/16

富士通クラウドテクノロジーズにて同社のFJcloud-Vおよび二フクラサービスにおいて、一部のロードバランサーに存在する脆弱性を悪用した不正アクセスが確認された。

この脆弱性は2022年5月4日に公開されたもので、その3日後の5月7日 15時過ぎから5月9日の22時半過ぎまでロードバランサーの脆弱性及び多層防御の一部設定不備を悪用した不正アクセスの形跡が確認された。これにより、当該サービスのコントロールパネルやAPIへのアクセス情報、ロードバランサーを経由した通信情報、顧客の証明書データなどが盗取された恐れがある。
同社は影響対象となる顧客に本件のお知らせ及び対処に関するお願いを実施した。なお、現時点では当該ロードバランサーを踏み台としたクラウド基盤内部の不正侵入の形跡は確認されておらず、不正アクセスによる情報の外部流出なども確認されていない。同社は当該ロードバランサーの脆弱性回避のための設定を施し、ネットワーク機器のアクセス制御を実施した。
同社のロードバランサが不正アクセスを受けた件で、クラウド基盤内部の不正侵入の形跡は確認されていないとしていたが、2022年5月4日～2022年5月11日の期間、ロードバランサーを通過する復号化された通信パケットの一部が窃取可能な状態にあったことが確認された。対象のシステムはメール配信サービスや顧客からの問い合わせ、申し込み各種受付システム、契約管理システムで、これにより氏名やメールアドレス、会社名などが盗取された可能性がある。なお、現時点では情報を盗取された事実は確認されていない。（2022年5月31日追加）
同社のロードバランサが不正アクセスを受けた件で、情報が盗取された恐れがあることを受け、同社は改めて消去済みファイルの復元や調査などのフォレンジック調査を行い、その結果を公表した。調査の結果、すでに報告済みの範囲であるが、不正アクセスの手法は当該ロードバランサーの脆弱性を悪用したもので、ロードバランサー内で任意のコマンドを実行できる状態であった。また、ロードバランサー上のお客様証明書データ等が圧縮されたファイルの痕跡が確認され、ロードバランサーを経由した通信の情報を収集した痕跡も確認された。ただし、これらの情報に認証情報や個人情報は確認されていない。（2022年6月7日追加）

出典:
https://www.onebe.co.jp/dlpnews/archives/%e5%af%8c%e5%a3%ab%e9%80%9a%e3%82%af%e3%83%a9%e3%82%a6%e3%83%89%e3%83%86%e3%82%af%e3%83%8e%e3%83%ad%e3%82%b8%e3%83%bc%e3%82%ba%e3%80%81%e8%84%86%e5%bc%b1%e6%80%a7%e3%82%92%e7%aa%81%e3%81%84%e3%81%9f.html

Dropboxのアカウント情報流出、被害は6800万件超に

2012年に起きたDropboxの情報流出事案で、盗まれたアカウント情報は6800万件を超えていたことが分かった。

2016年09月01日 07時06分 公開　[鈴木聖子，ITmedia]

印刷　見る　Share　B!　34

Motherboardサイト

米Dropboxが2012年に起きた情報流出事案に関連して一部ユーザーにパスワードの変更を促していた問題で、ニュースサイトのMotherboardは8月30日、盗まれたDropboxユーザーのアカウント情報は6800万件を超えていたことが分かったと伝えた。

Motherboardによると、Dropboxユーザーのメールアドレスとハッシュ化されたパスワードを記録したファイルをデータベース取引関係筋から入手して調べたところ、ファイル4本に計6868万741件のアカウント情報が含まれていることが分かった。Dropboxの幹部もこれが同社のユーザーのものであることを確認したという。

出典: https://www.itmedia.co.jp/enterprise/articles/1609/01/news073.html

背景④ クラウドサービスは安全か？

パスワード管理サービス「LastPass」のバックアップ環境に不正アクセス

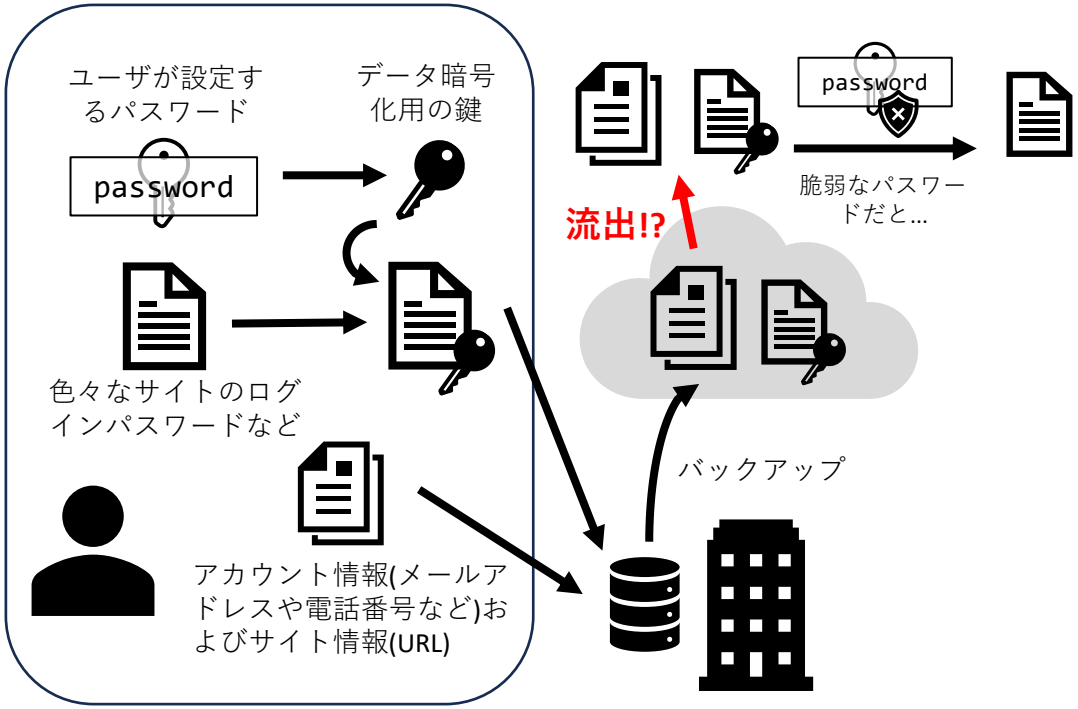
掲載日 2022/12/23 13:46　　著者：後藤大地

不正アクセス　サイバーセキュリティ

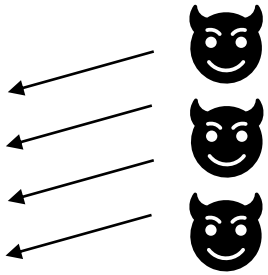
LastPassは12月22日(米国時間)、公式ブログ「Notice of Recent Security Incident」を更新し、同社が本番データのバックアップを保存するために使用しているサードパーティのクラウドストレージサービスに第三者による不正なアクセスがあったことを報告した。この不正アクセスによって、顧客アカウント情報に加えて、暗号化された機密フィールドを含む顧客のVaultsデータのバックアップが流出した恐れがあるという。

出典: https://news.mynavi.jp/techplus/article/20221223-2544442/

パスワードベースの暗号化はクラウドサービスも含め広く採用されている
→パスワードの強度次第では，暗号文の流出だけでも機密情報が危険に晒される



背景⑤クラウドサービスを利用してもらうには



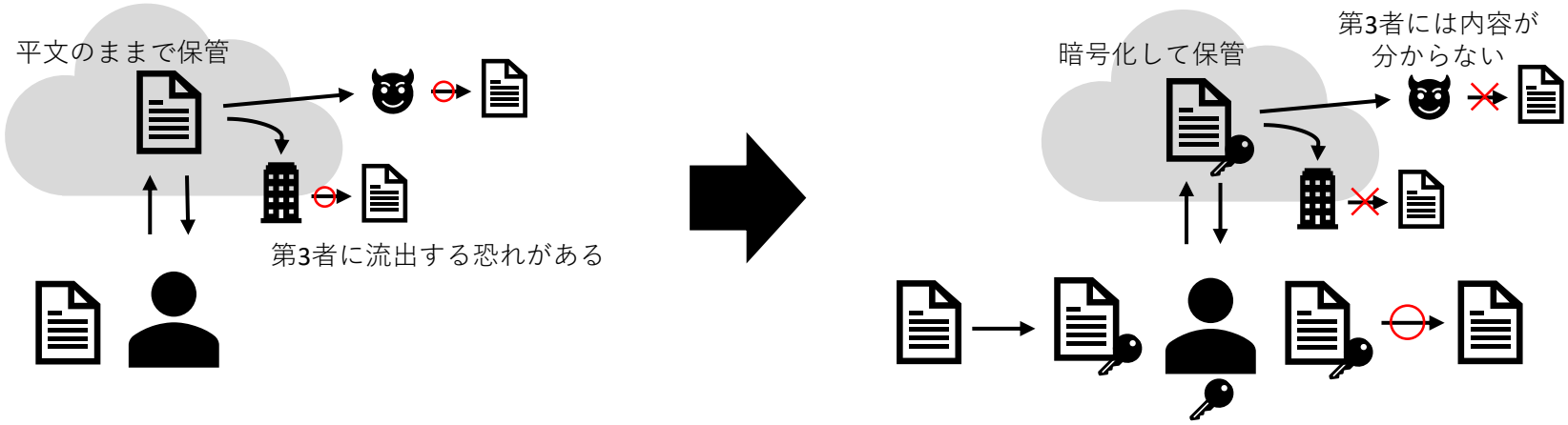
多数の利用者
↓
攻撃対象に
なりやすい

技術 → 機密情報の流出

高い安全性
↓
強力な宣伝材料

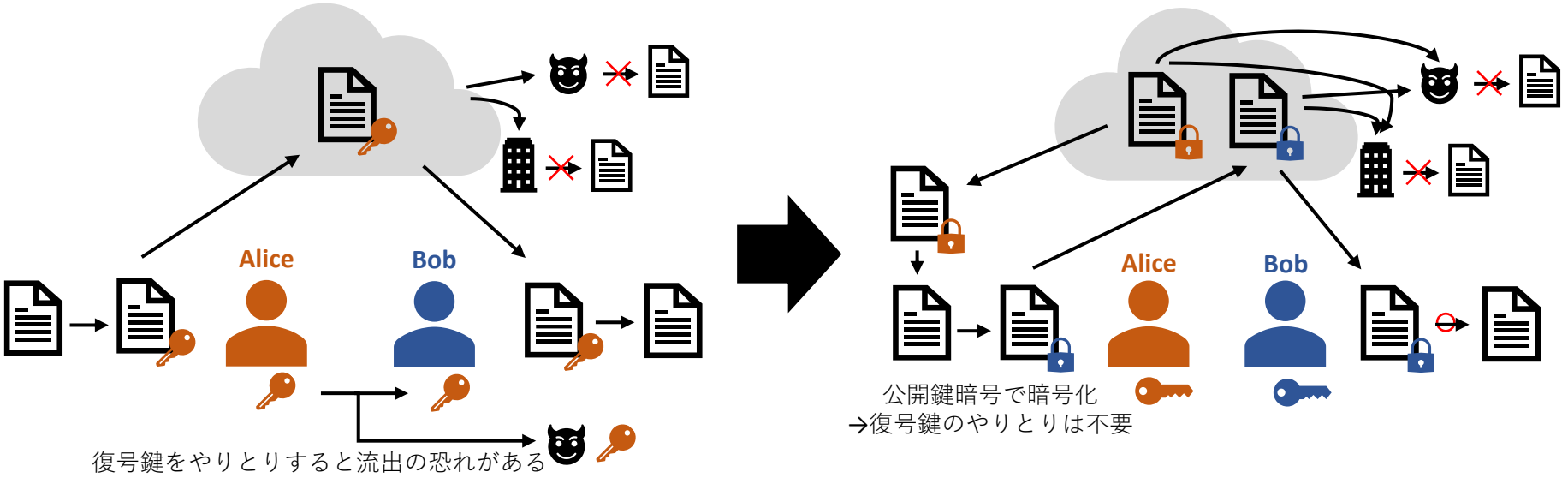
データを安全に管理する

- データ をそのままクラウドに保管→情報流出の恐れ
- 安全に保管するには暗号化が効果的
  - 復号鍵 を持たない人からは内容を見られない



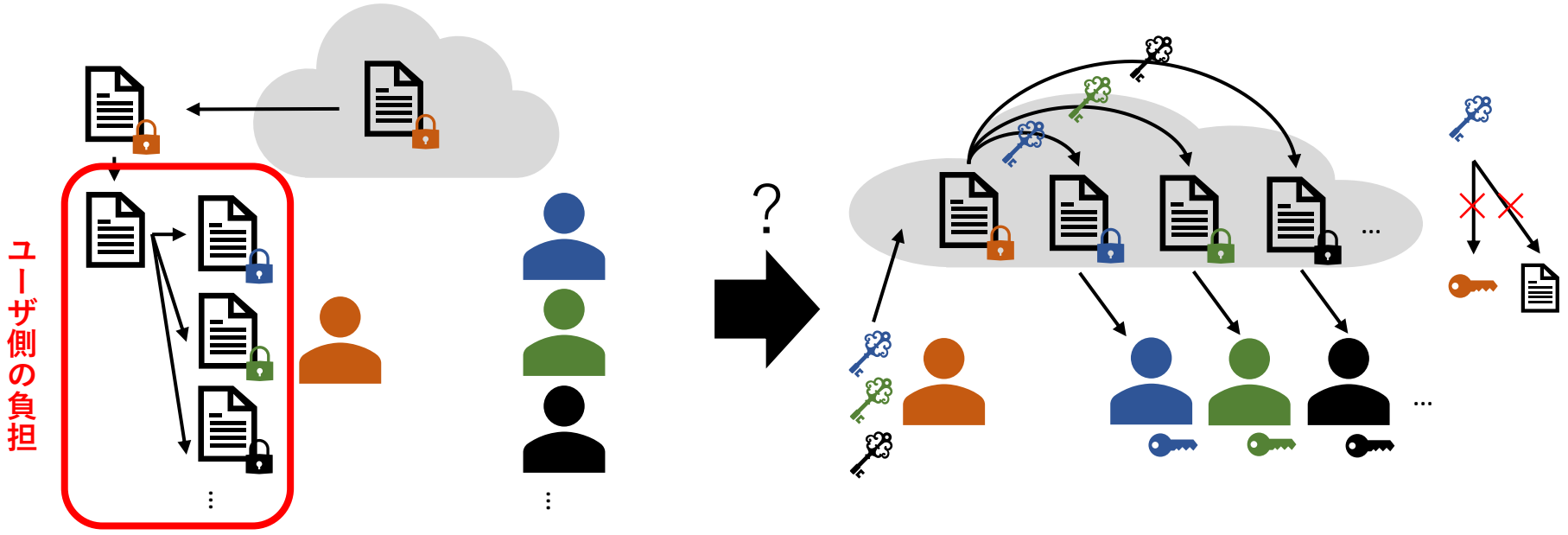
データを安全に共有する

- 公開鍵暗号を利用する
  - 暗号化鍵 🔒 と復号鍵 🔑 が別なので事前の鍵共有が不要



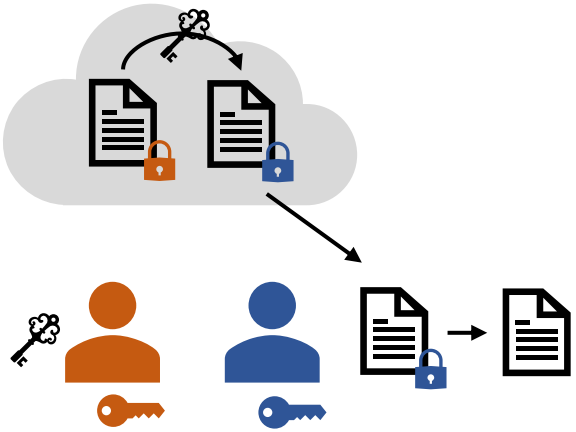
データを大勢と共有するときは？

- 暗号化するのは平文を持っているユーザ側→大きな負担
  - サーバに暗号文の生成を安全に行わせられないか？



プロキシ再暗号化とは

- 再暗号化を実現する
  - 再暗号化: Aliceが復号可能な暗号文→Bobが復号可能な暗号文
  - 再暗号化鍵 を用いて再暗号化が行われる
- その計算は**第3者が実行できる**
- 再暗号化鍵 や暗号文 が外部に流出しても秘密情報 は漏洩しない
  - サーバにも漏洩しない



プロキシ再暗号化の仕組み

① 暗号化 

② 通常の復号 

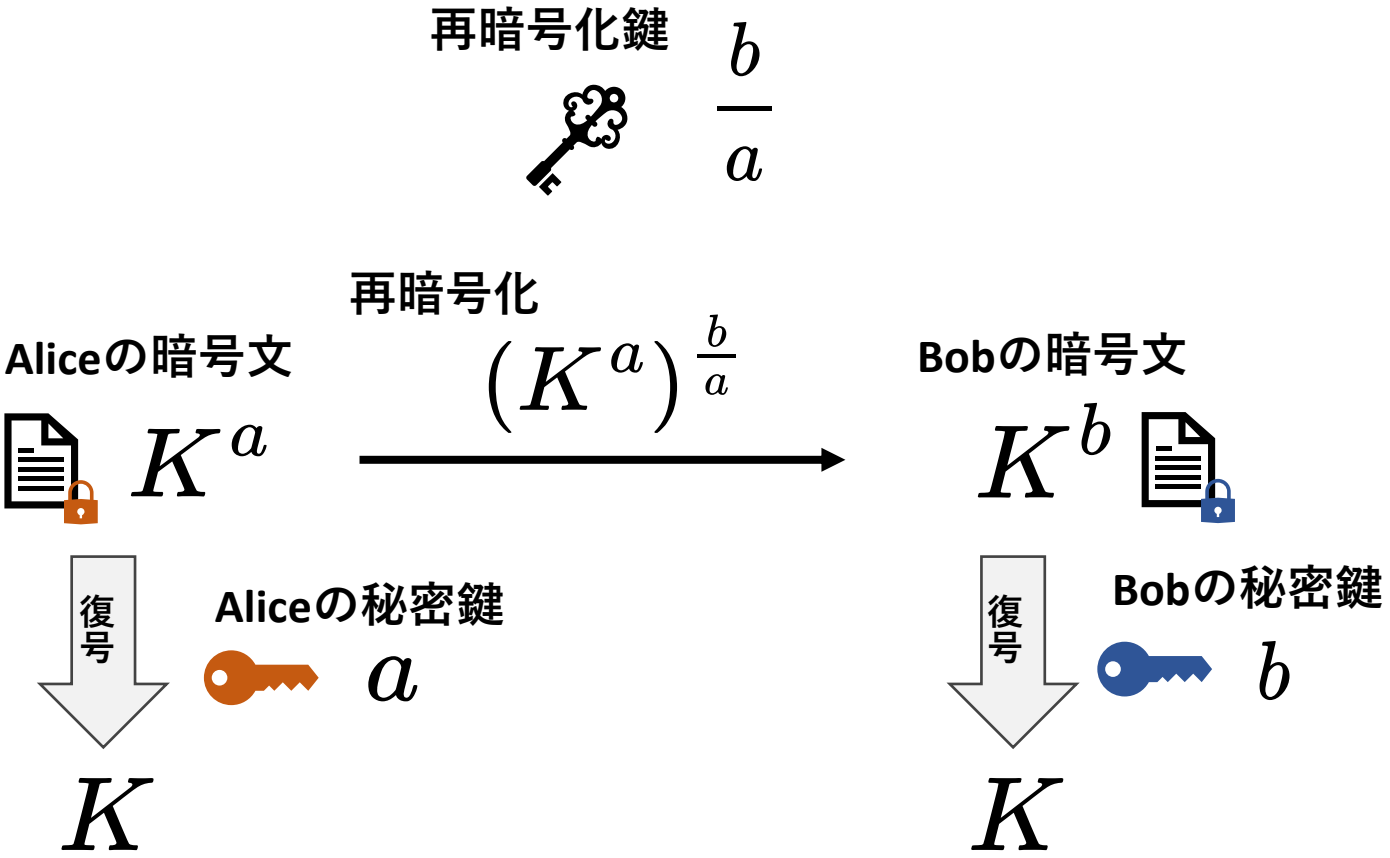
③ 再暗号化鍵の生成 

④ 再暗号化 

⑤ 再暗号化による暗号文の復号 (通常の復号と同じでも違ってもよい)

BBS暗号　：ElGamal暗号を派生させた暗号方式

※プロキシ再暗号化の先駆けである方式.
(プロキシ再暗号化としては安全性が不十分)

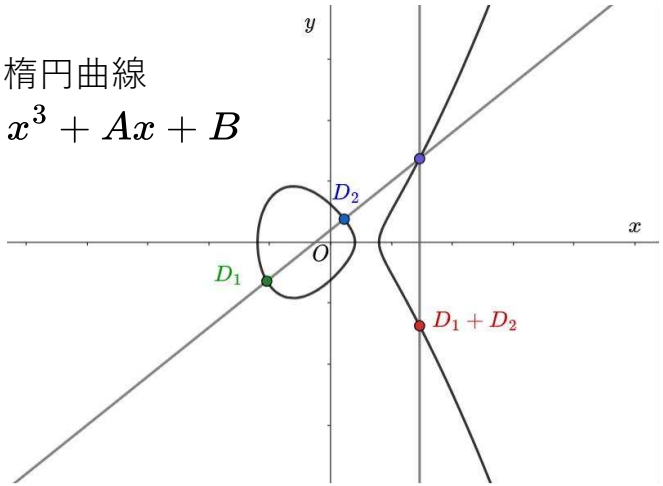


正式なプロキシ再暗号化技術の例

- AFGH(Ateniese-Fu-Green-Hohenberger)暗号
  - ペアリング(双線形写像)を利用 (ペアリングは楕円曲線などで利用可能)

楕円曲線

$y^2 = x^3 + Ax + B$

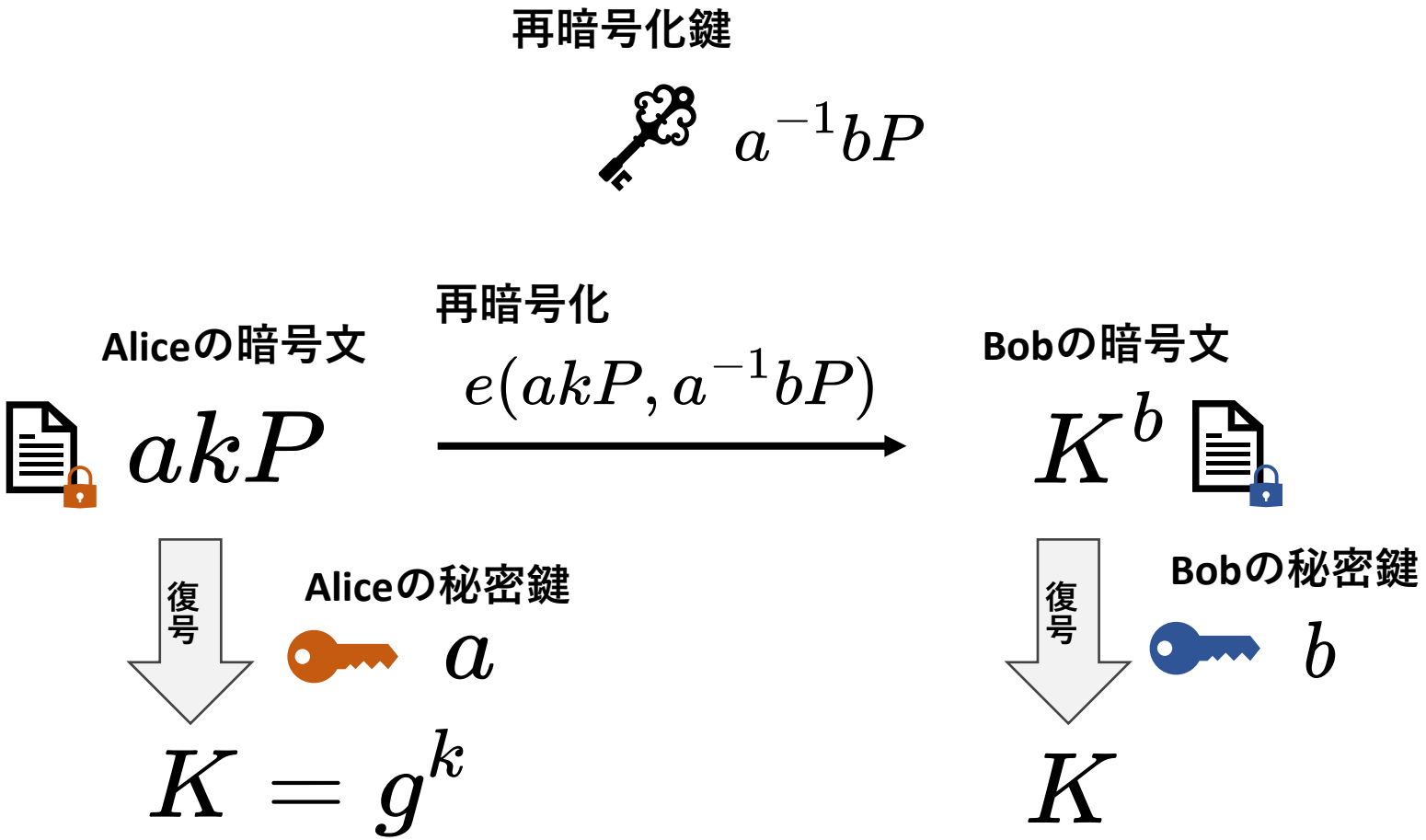


$$e(aP, Q) = e(P, Q)^a$$
$$e(P, bQ) = e(P, Q)^b$$
$$e(aP, bQ) = e(P, Q)^{ab}$$

e ：ペアリング

P, Q ：楕円曲線上の点

AFGH暗号


再暗号化鍵
$a^{-1}bP$
再暗号化
$e(akP, a^{-1}bP)$
Aliceの暗号文
akP
Bobの暗号文
K^b
復号
Aliceの秘密鍵
a
$K = g^k$
復号
Bobの秘密鍵
b
K

なぜプロキシ再暗号化は安全か？

- BBS暗号の公開情報から平文を当てる難しさ ≥ DH問題の難しさ
  - DH問題：g, g^a, g^b から g^{ab} を求める問題

$$g, g^a, g^b, c_1(= mg^k), c_2(= g^{ak}), a^{-1}b \xrightarrow{\text{難}} m$$

- 復号鍵が入手できる $\Longrightarrow$ 平文が入手できる
  → 平文が入手困難 $\Longrightarrow$ 復号鍵も入手困難

再暗号化鍵が流出しても秘密情報は漏洩しない

まとめ

- プロキシ再暗号化は，暗号文を

・第3者に　　・復号させずに

再暗号化させる暗号方式.

- 送信先の公開鍵と自身の秘密鍵で作成した再暗号鍵からは，第3者は自分平文や復号鍵などの秘密情報が分からない.

参考

1. 総務省, “令和4年通信利用動向調査の結果”, 2023, available at https://www.soumu.go.jp/johotsusintokei/statistics/data/230529_1.pdf
2. M. Blaze, G. Bleumer and M. Strauss, “Divertible Protocols and Atomic Proxy Cryptography”, EUROCRYPT, Springer-Verlag, pp. 127-144, 1998.
3. G. Ateniese, K. Fu, M. Green and S. Hohenberger, “Improved proxy re-encryption schemes with applications to secure distributed storage”, ACM Trans. Inf. Syst. Secur., Vol. 9, No. 1, pp. 1-30, 2006.

IDベース暗号とは

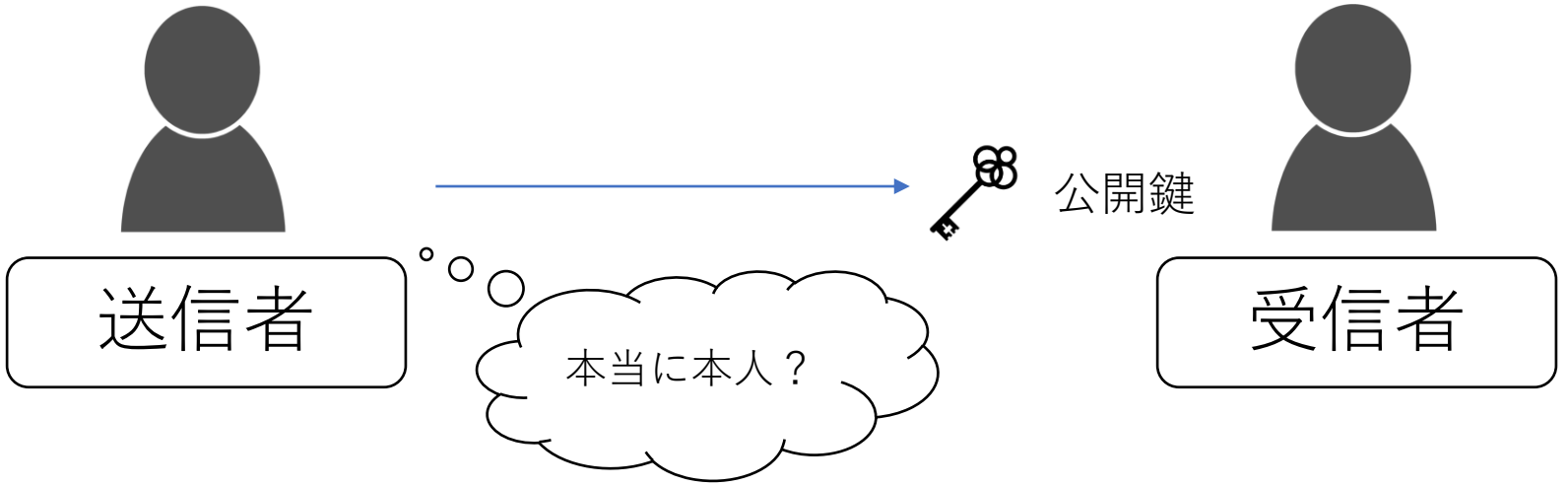
目次

目次

共通鍵暗号と公開鍵暗号

共通鍵暗号…暗号鍵と復号鍵が同じ
　　信用できる通信経路での鍵のやり取りが必要
公開鍵暗号…暗号鍵（公開鍵）と復号鍵（秘密鍵）が異なる
　　共通鍵暗号よりも計算時間が多くなる

公開鍵暗号が抱える問題

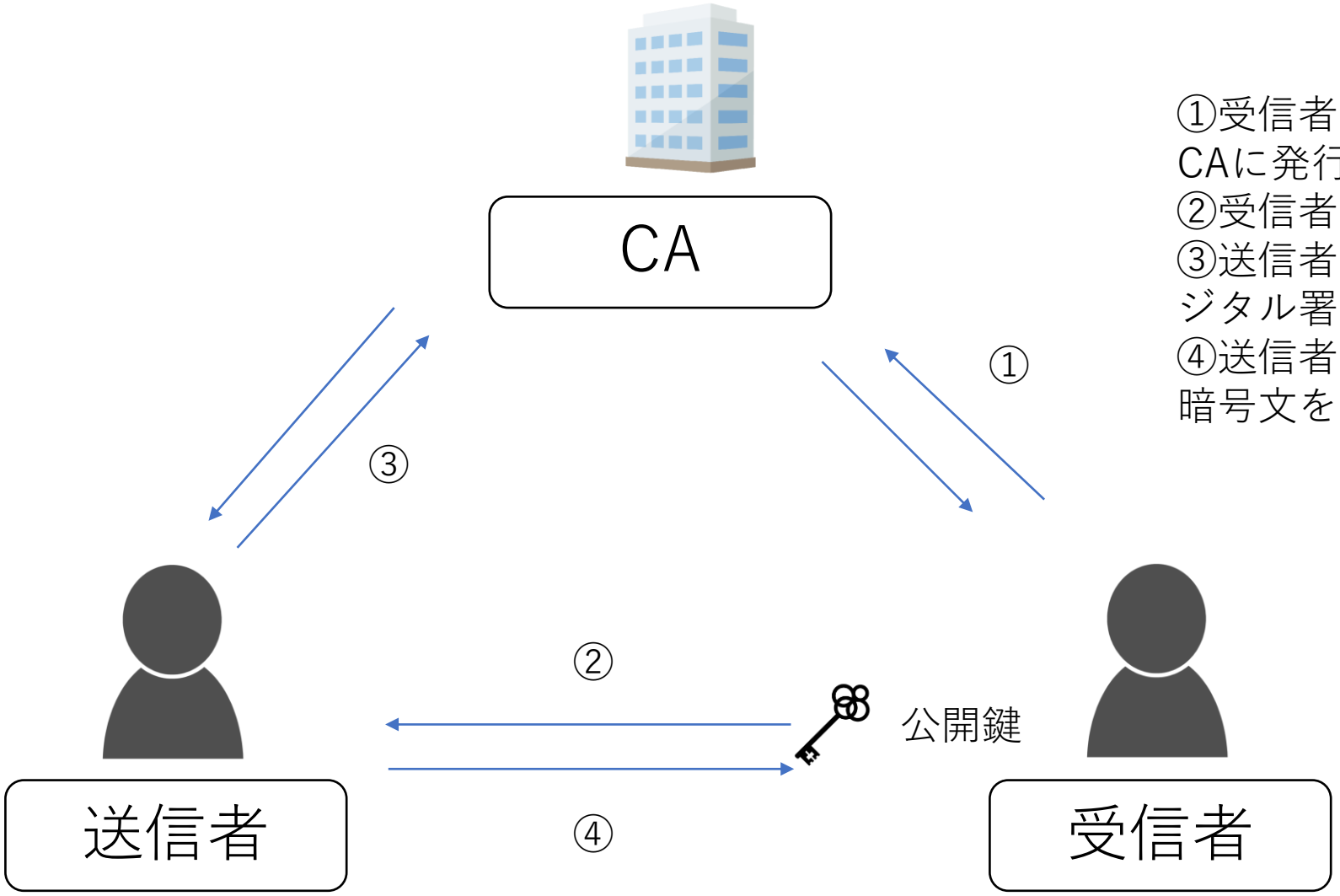


メッセージを送る際に、送信者からは公開鍵しかわからない

⇒本当に送りたい相手の公開鍵なのか？

公開鍵証明書認証局(CA)に問い合わせ

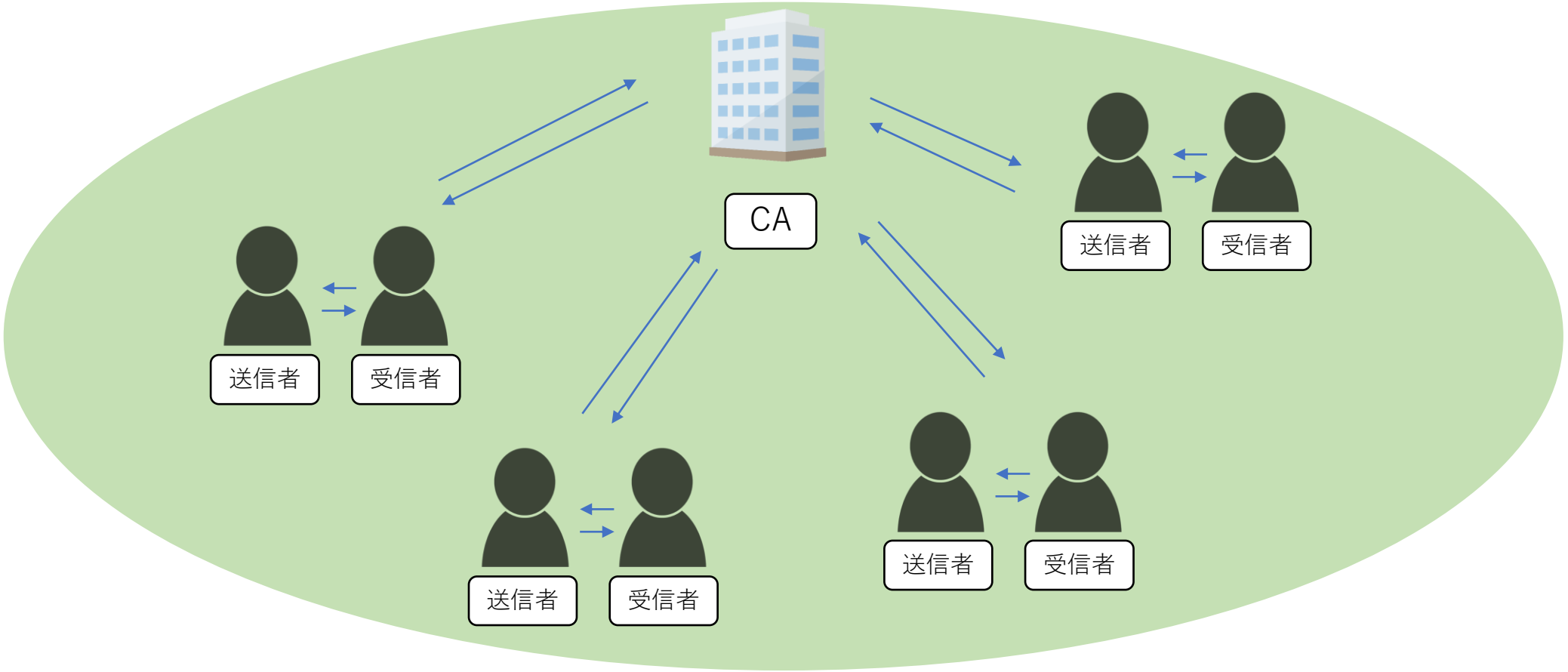
公開鍵暗号が抱える問題



①受信者がデジタル署名付きの公開鍵証明書をCAに発行してもらう(公開鍵公開時)
②受信者が公開鍵証明書を送信者へ送る
③送信者はCAに問い合わせて公開鍵証明書のデジタル署名用の鍵を入手し検証する
④送信者が受信者側の公開鍵を用いて暗号化し、暗号文を送る

目次

公開鍵暗号が抱える問題

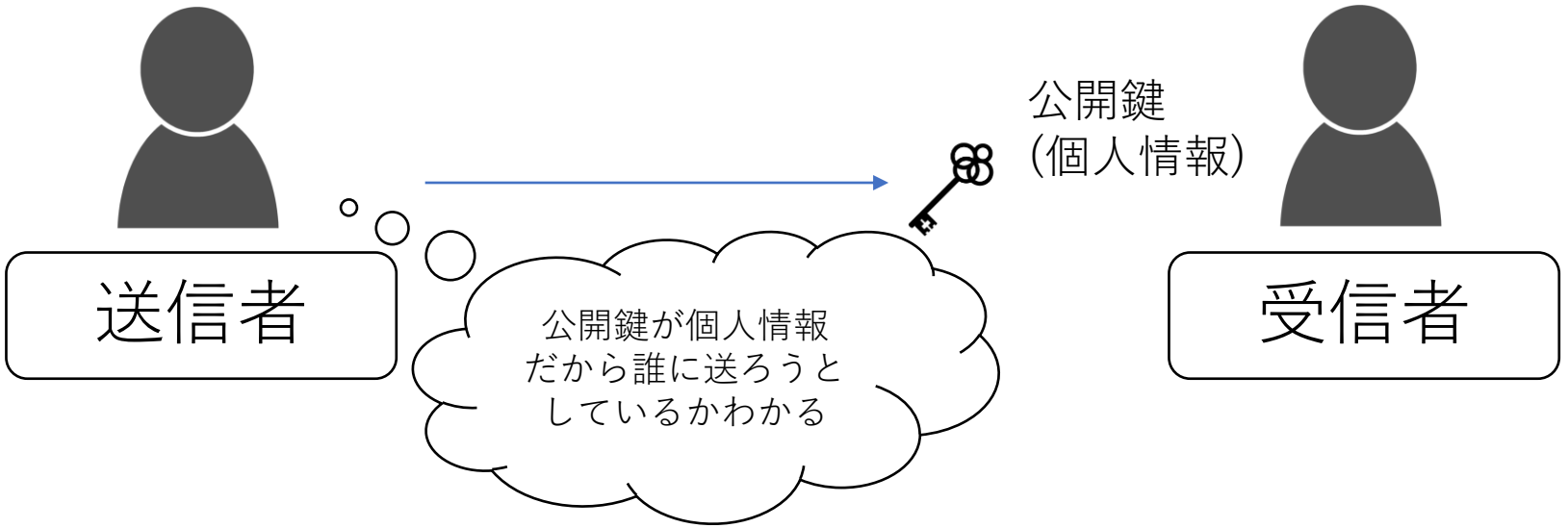


証明書の発行が社会全体のいたるところで
行われており大きな負担になっている

全世界での一日あたりの発行数

実際に大手認証局であるLet`s Encryptは10月28日にX(旧Twitter)上で一日の公開鍵証明書の発行数が3.5億件に至ったと述べている

個人情報を公開鍵にした暗号



公開鍵とID(identity)の同一化が求められ

“公開鍵の認証が必要ない暗号”として生まれたのがIDベース暗号

IDとして利用できるものは氏名・住所・メアド・ドメイン名など

目次

IDベース暗号の仕組み

ⅰ)Setup

-公開パラメータの集合　$MPK = (p, G_1, G_T, e, n, g, g_{pub}, H_1, H_2)$

-マスターキー　MSK

ⅱ)Extract

-ユーザーキー　$SK_{ID}(MPK, IDから生成)$

ⅲ)Encrypt

-平文　M

-暗号文　$C(MPK, ID, Mから生成)$

ⅳ)Decrypt

-平文　$M(MPK, SK_{ID}, Cから生成)$

IDベース暗号の仕組み

Setup

・このアルゴリズムでは公開パラメータとマスターキーを生成する
・素数位数の群は巡回群であるため、二つの素数位数の群を用いて双線形性を満たすPairing写像を定義できる
・巡回群には生成元が必ず存在するため，生成元とマスターキーなどを用いていくつかのパラメータを生成し，ハッシュ関数を含めて公開パラメータとする

IDベース暗号の仕組み

Setup

素数p
位数pの乗法群G_1, G_r
G_1の生成元g
Pairing写像$e: G_1 \times G_1 \rightarrow G_r$
ランダムな元$s \in {Z_p}^*$

IDベース暗号の仕組み

Setup

$g_{pub} = g^s$

ハッシュ関数$H_1: \{0,1\}^* \rightarrow G_1$

$H_2: G_r \rightarrow \{0,1\}^*$

公開パラメータ$MPK = (p, G_1, G_T, e, n, g, g_{pub}, H_1, H_2)$

マスターキー$MSK = s$

IDベース暗号の仕組み

Extract

・このアルゴリズムではユーザーキーを生成する
・公開鍵として用いたいIDを二進数で表し，ハッシュ関数を用いて巡回群の元に変換し公開鍵を生成する
・公開鍵にマスターキーを用いてユーザーキーを生成する

IDベース暗号の仕組み

Extract

個人情報$ID \in \{1,0\}^*$
公開鍵$Q_{ID} = H_1(ID) \in G_1$
ユーザーキー$SK_{ID} = {Q_{ID}}^s$

IDベース暗号の仕組み

Encrypt

・このアルゴリズムでは平文から暗号文を生成する
・公開パラメータと公開鍵にペアリング写像を適用し、巡回群の元を生成する
・生成した巡回群の元にハッシュ関数を適用し、二進数の暗号鍵を生成する
・$Z_p{}^*$からランダムな元を一つ選択し、生成した暗号鍵と公開パラメータ、平文を共に用いることで暗号文を計算できる

IDベース暗号の仕組み

Encrypt

平文M

$g_{ID} = e(Q_{ID}, g_{pub}) \in G_r$

ランダムな元$r \in {Z_p}^*$

暗号文$C = (g^r, M \oplus H_2\,({g_{ID}}^r))$

IDベース暗号の仕組み

Decrypt

・このアルゴリズムでは暗号文から平文を復号する

・送られてきた暗号文にハッシュ関数とPairing写像、ユーザーキーを用いることで復号できる

IDベース暗号の仕組み

Decrypt

暗号文$C = (U, V)$

平文$M = V \oplus H_2(e(SK_{ID}, U))$

IDベース暗号の安全性

IDベース暗号では双線形性を満たすペアリング写像が用いられている

ペアリング写像の性質から、盗聴者が暗号を解読するには

・送信者が持つランダムパラメータr
・受信者が持つユーザー鍵SK_{ID}

のどちらかが必要であるが、どちらも公開情報でないことから
解読は困難であることが知られている

目次

IDベース暗号によって得られた効果・性能

IDベース暗号はペアリングを用いて実装され、他の暗号に応用されている。

フォワードセキュア暗号

-鍵の漏洩に強い暗号

放送暗号

-多数の受信者に一括で暗号文を送信してもサイズが大きくならない暗号

タイムリリース暗号

-ある時刻が来るまでは復号することが出来ない暗号

キーワード検索暗号

-暗号化されたままキーワード検索を行うことが出来る暗号

属性ベース暗号

属性ベース暗号ができた背景

文書の管理

昔

紙媒体

現在

電子ファイル

電子ファイルの閲覧権限の付与には、公開鍵暗号が使用されている。

従来の公開鍵暗号の問題点

- 閲覧権限の付与の非効率性

☆組織の人数の大きさに依存

☆権限付与の手間が多く、複雑

鍵と属性を1対N、N対1対応できる属性ベース暗号が提案された。

属性ベース暗号に求められた機能

1. 暗号文の復号者を管理する機能

   - 復号条件を満たした人のみが復号可能。

2. 暗号鍵と復号鍵が1対N、N対1対応する機能

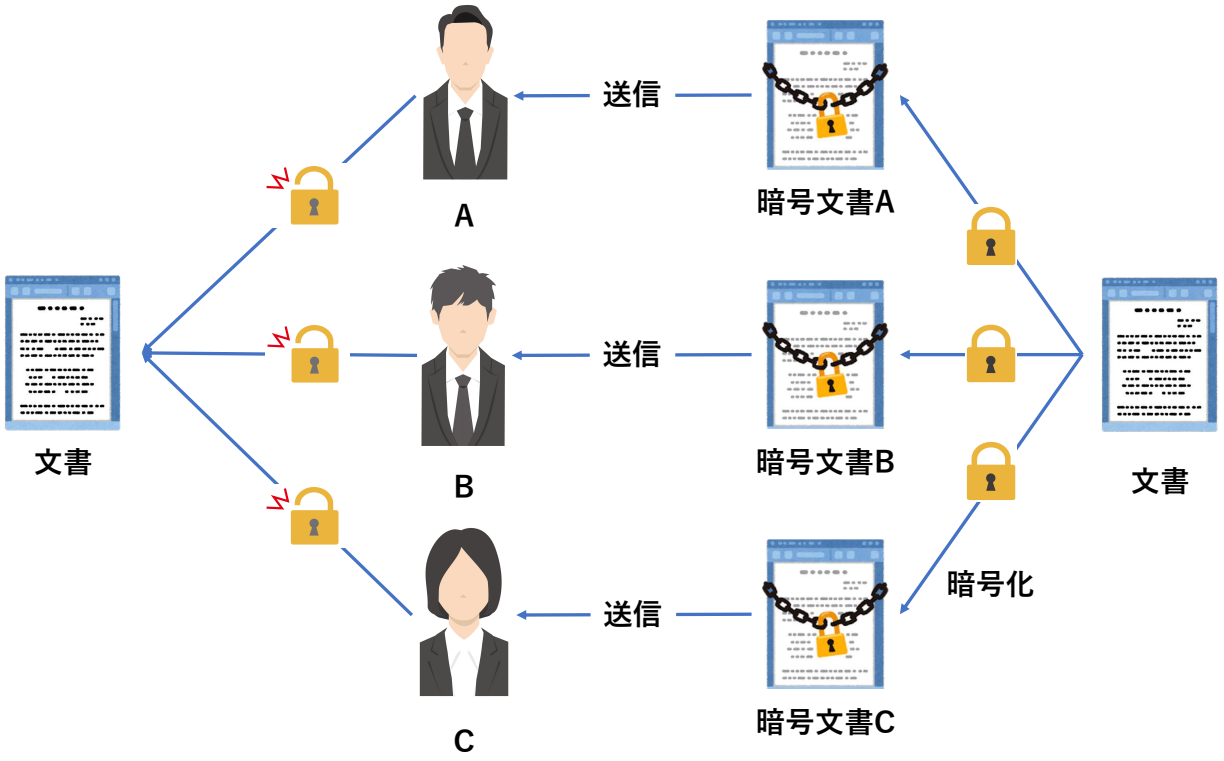
   - 一つの暗号文を複数の秘密鍵で復号可能。

   - 一つの秘密鍵で複数の暗号文を復号可能。

【機能例】ある会社での文書のアクセス権限の付与

社員100名に1MBの文書を読めるようにする。

<u>属性ベース暗号「なし」</u>



【問題点】

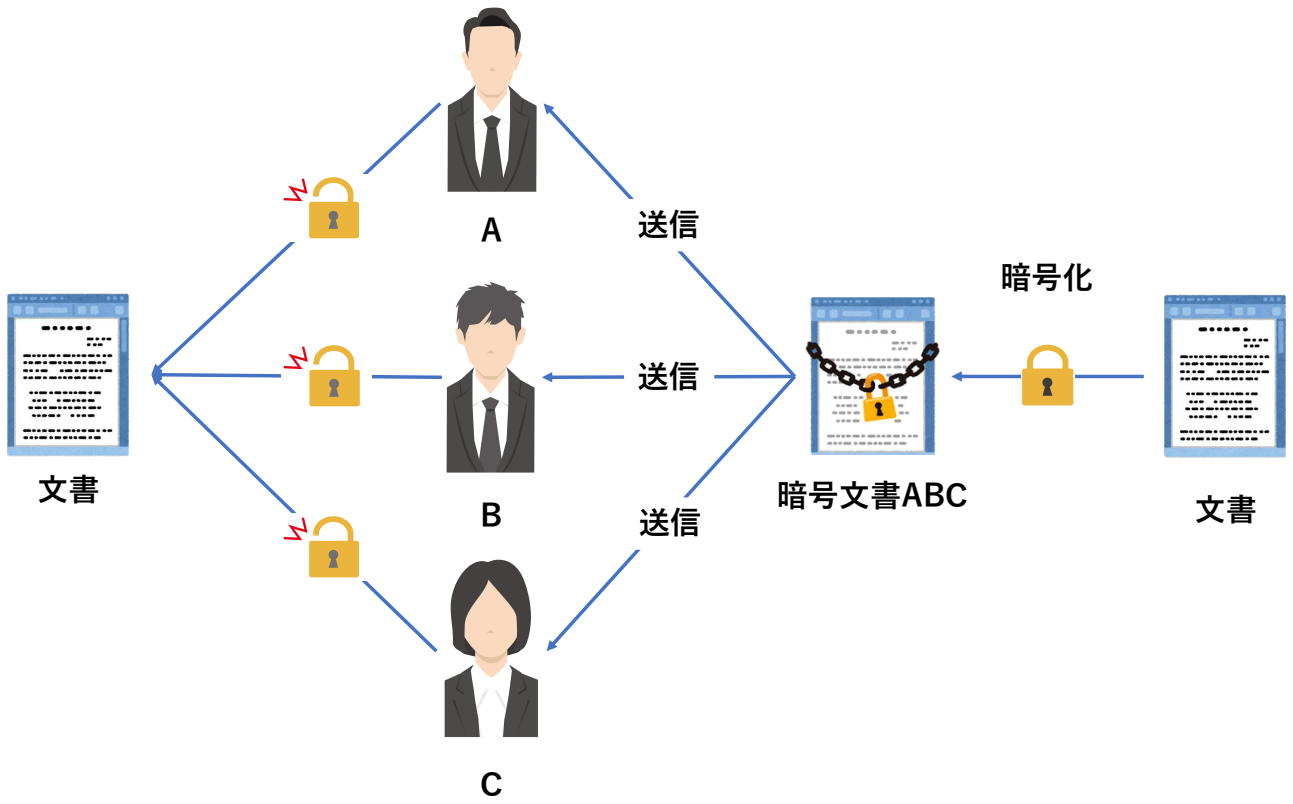
・社員それぞれに対して文書を暗号化しなくてはいけない。

→暗号化した文書は、1MB×100人分=100MB必要になる。
人数に比例して、データ量が大きくなる。

【機能例】ある会社での文書のアクセス権限の付与

社員100名に1MBの文書を読めるようにする。

<u>属性ベース暗号「あり」</u>



【改善点】

・1つの文書に対して、

必要な暗号化した文書は1つのみ。

→暗号化した文書は、
1MB×1人分=1MB必要。
人数に比例せず一定。

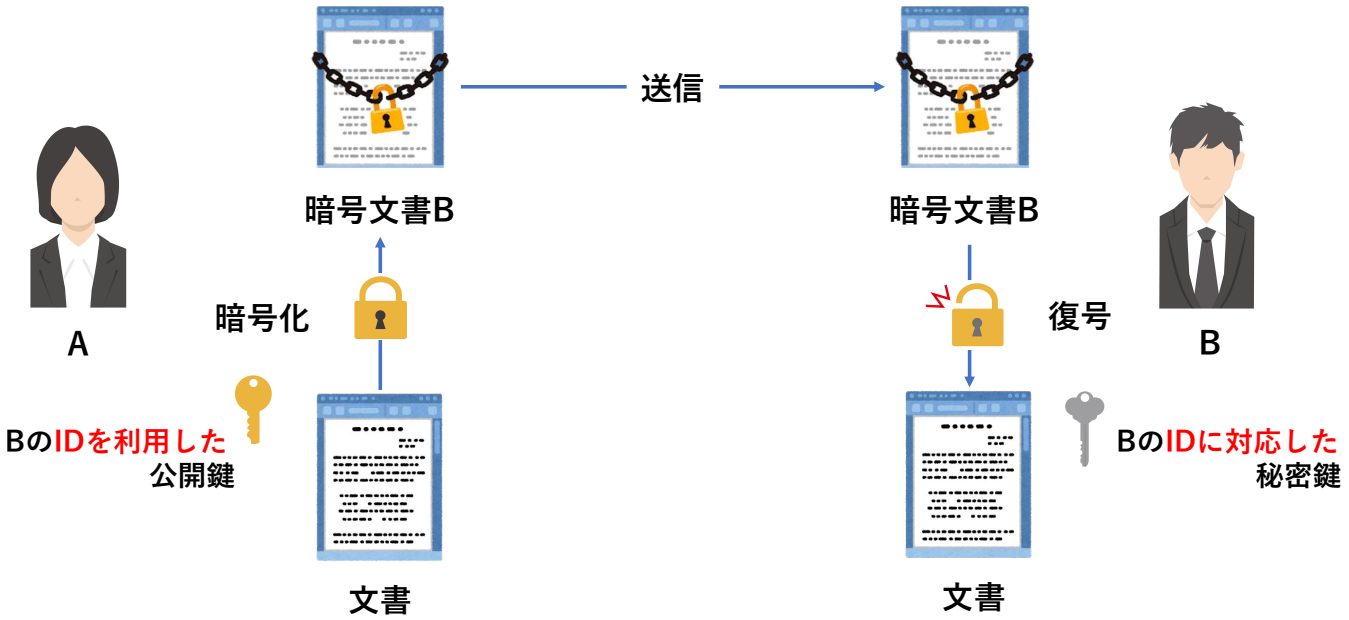
属性ベース暗号の仕組み (IDベース暗号の一般化)

IDベース暗号

個人を特定することが可能な情報であるIDを公開鍵として用いる暗号方式。

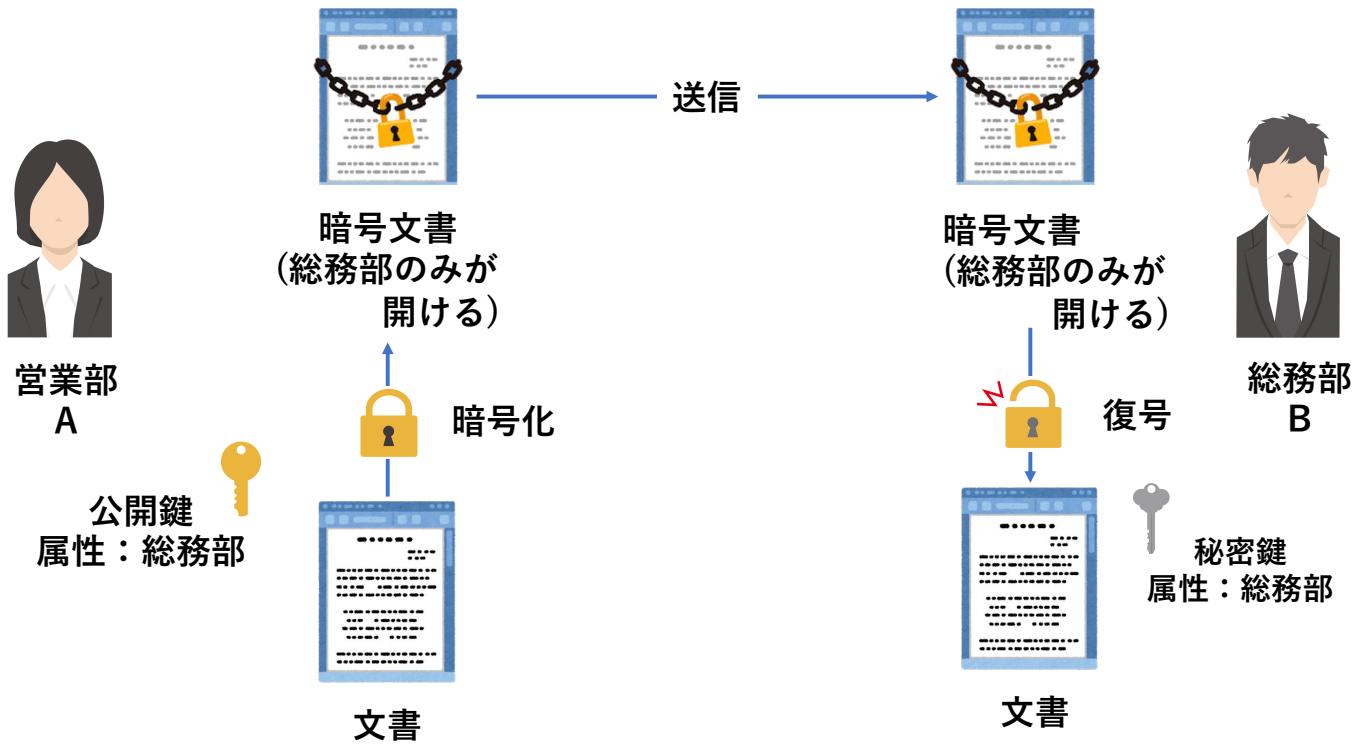
IDの例：emailアドレス、携帯番号 etc.

従来の公開鍵暗号は、公開鍵とその所有者を証明書で保証する必要があったが、IDを利用することで、保証が不要となった。



属性が一つの属性ベース暗号

【例】総務部の人だけが復号できる属性ベース暗号



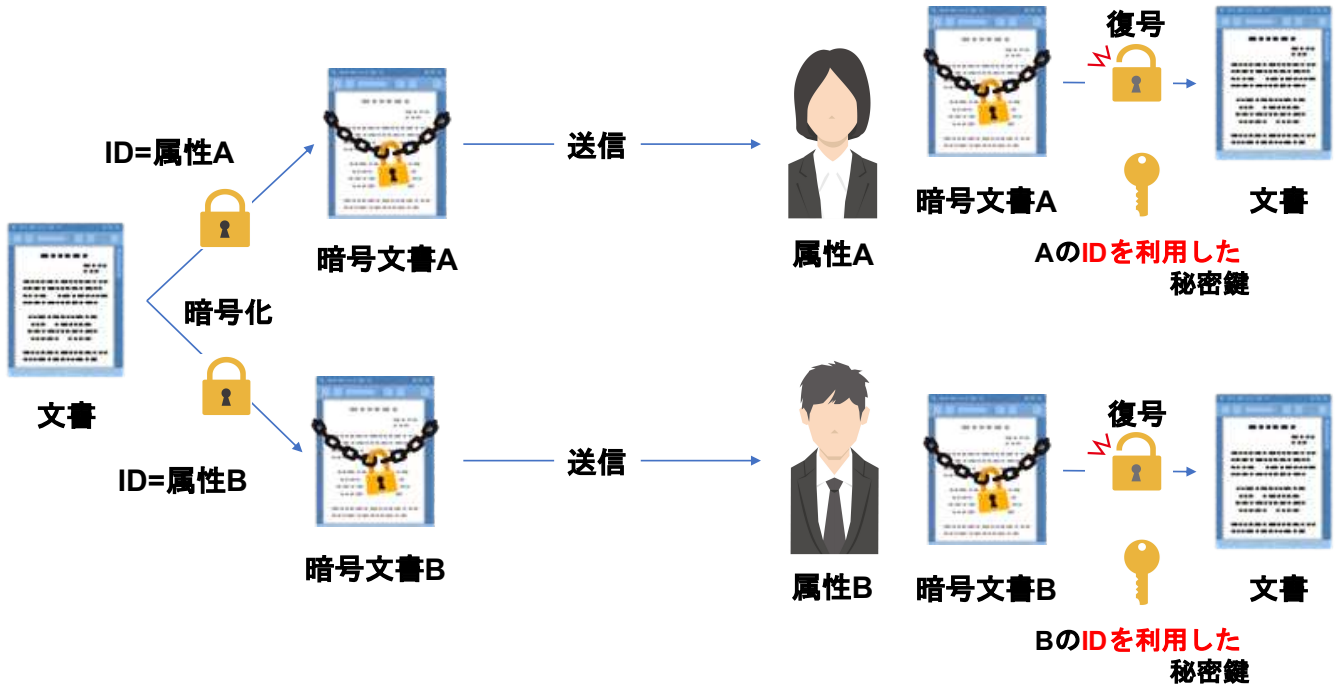
・「総務部」という属性を公開鍵として文書を暗号化し、
「総務部」という属性に対応した秘密鍵を用いて暗号文の復号を行う。

・IDベース暗号におけるIDと属性ベース暗号における属性が同じ働きをしている。

→属性が一つの属性ベース暗号は、属性がIDのIDベース暗号である。

属性ベース暗号の仕組み (1対N対応)

属性が複数ある属性ベース暗号



$A \cup B$

・AもしくはBの属性を持てば暗号を復号できる.

【IDベース暗号を用いた実現方法】

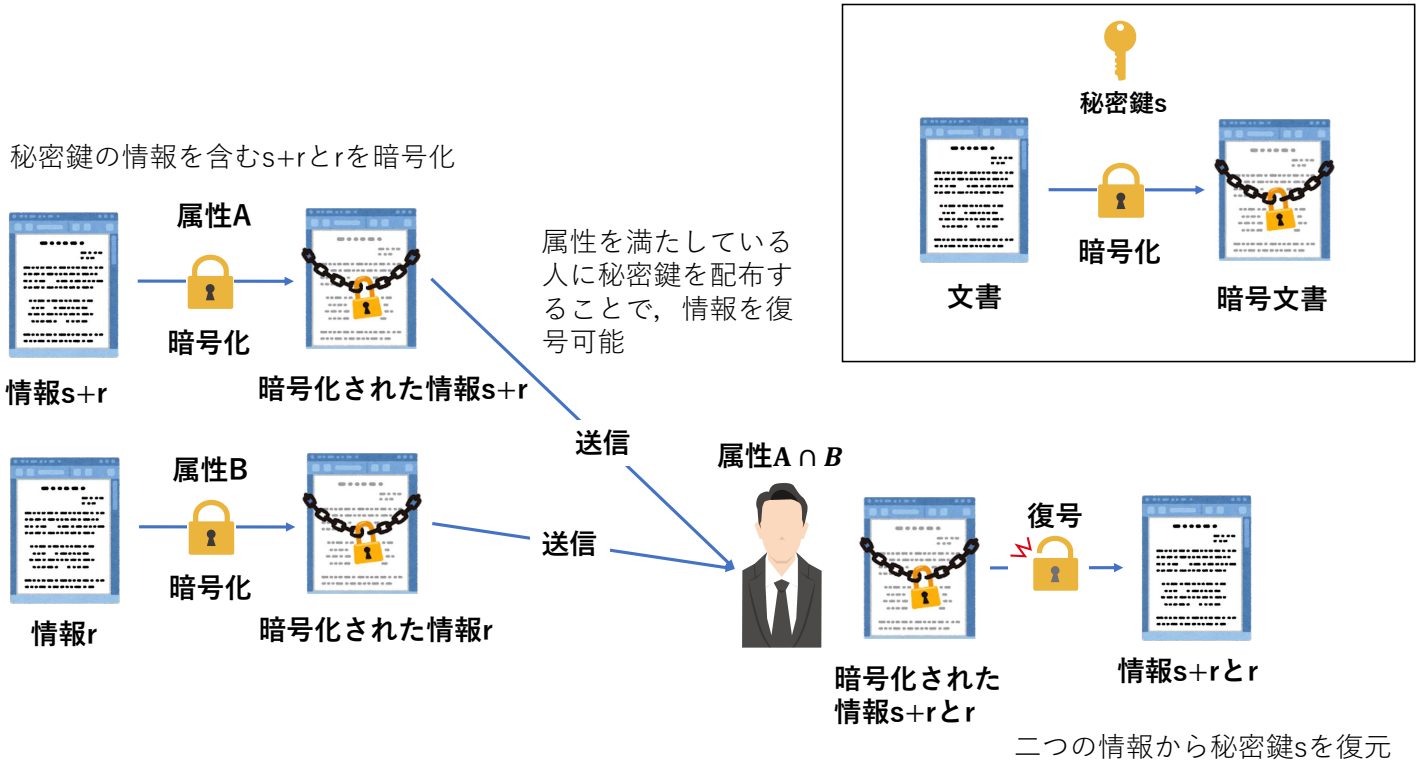
・属性AをIDとしたIDベース暗号で文書を暗号化.

・同様に，属性BをIDとしたIDベース暗号で文書を暗号化.

・属性を満たした者にそれぞれ秘密鍵を配布する.

属性ベース暗号の仕組み (1対N対応)

属性が複数ある属性ベース暗号



$A \cap B$

・AとB両方の属性を持たないと暗号を復号できる.

【IDベース暗号を用いた実現方法】

・属性AをIDとしたIDベース暗号で情報 “s+r”を暗号化.

・属性BをIDとしたIDベース暗号で情報 “r”を暗号化.

・両方の属性を満たしたものは，情報“s+r”と“r”を復号することができ，その二つの情報から，秘密鍵 “s”の情報を入手できる.

KP-ABE(Key Policy Attribute Based Encryption)

暗号文に属性を、秘密鍵にポリシーを埋め込む方式の属性ベース暗号

秘密鍵にポリシーを埋め込むことで、条件を満たした人に鍵を配布する。

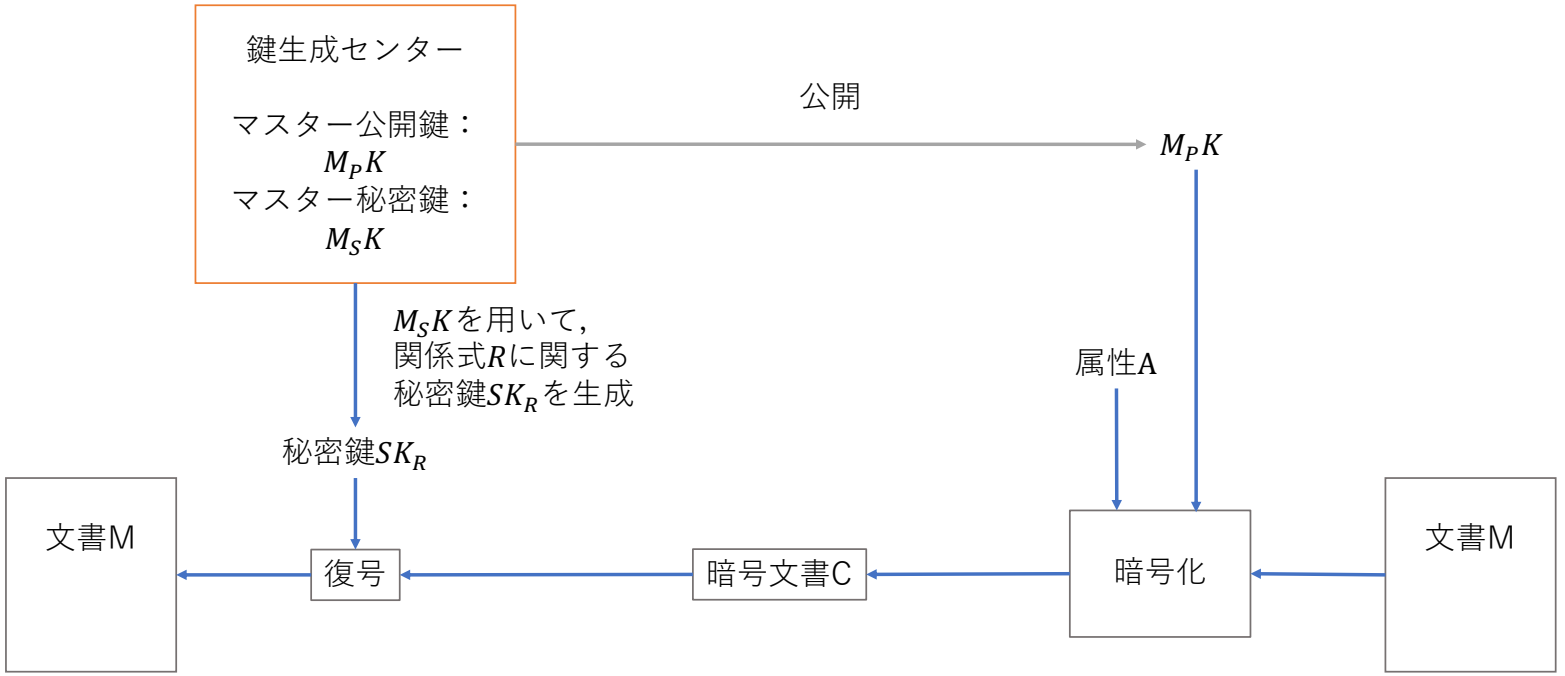


図1:KP-ABEの仕組み

CP-ABE(Ciphertext Policy Attribute Based Encryption)

暗号文にポリシーを、秘密鍵に属性を埋め込む方式

暗号化の際にポリシーを埋め込むことで、ポリシーを満たした人のみが暗号文を復号できる。

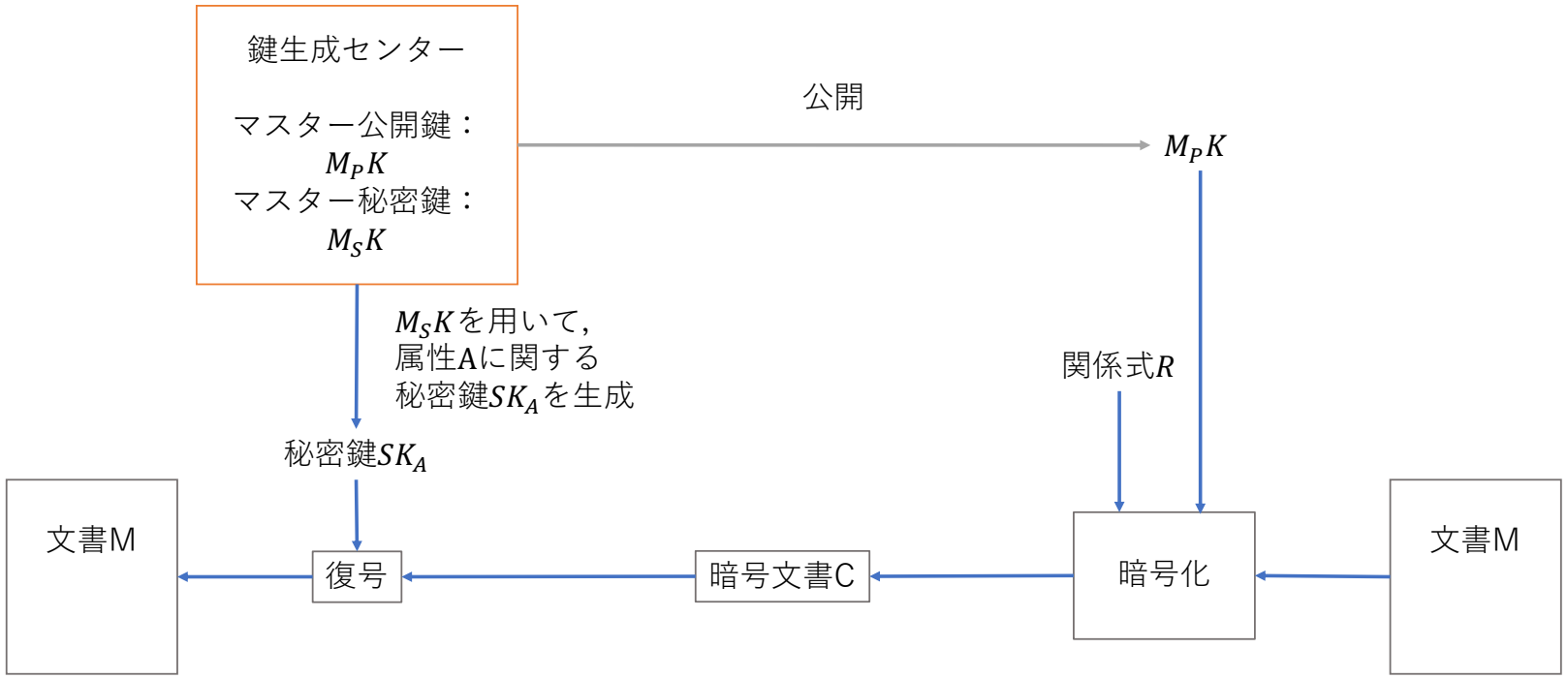


図2:CP-ABEの仕組み

属性ベース暗号によって得られた効果

①鍵の管理が効率的に行える

従来の公開鍵暗号		属性ベース暗号
個人一人一人に秘密鍵が必要	→	属性ごとの鍵生成

②アクセス制御が容易

従来の公開鍵暗号		属性ベース暗号
個人一人一人に アクセス権限を付与	→	属性ごとに アクセス権限を付与

属性ベース暗号の安全性

DBDH仮定

「楕円曲線上の点Pと整数$\mathrm{a}, \mathrm{b}, \mathrm{c}$と有限体の値$d$に対して、$(P, aP, bP, cP, d)$が与えられたときに$d = e(P, P)^{abc}$を判定する」という問題を解くことが難しいという仮定

CKKS方式の準同型暗号

目次

はじめに

近年、 Chatgptなどのニューラルベースの AI技術が猛威を振るっている.
AIを利用したサービスの提供において、サーバへのAIモデル搭載は必要不可欠
サーバが攻撃を受けるとAIモデル、学習データが漏洩してしまう恐れがある.
そこで、サーバ上の学習/推論を暗号化したままできないか、という議論がある
本活動では秘密計算もとい秘密学習/推論を実現する最先端暗号技術「CKKS方式の準同型暗号」について調査した.

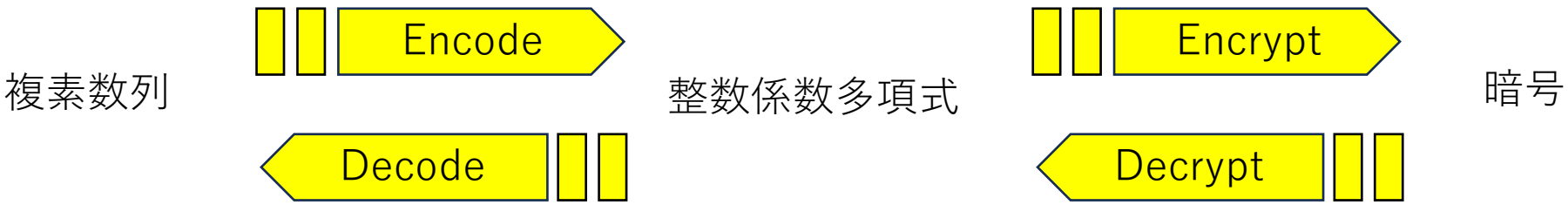
原論文紹介

論文名: Homomorphic Encryption for Arithmetic of Approximate Numbers.

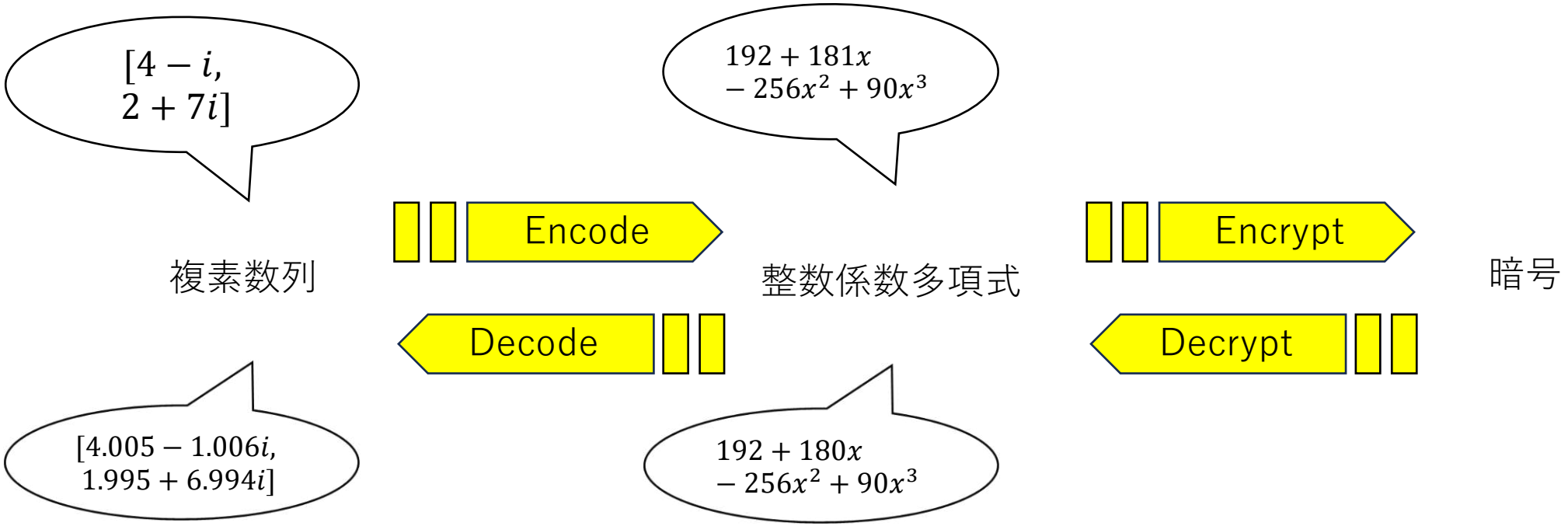
著者: Jung Hee Cheon, Andrey Kim, Miran Kim, Yong Soo Song

国際会議: ASIACRYPT 2017: 409-437

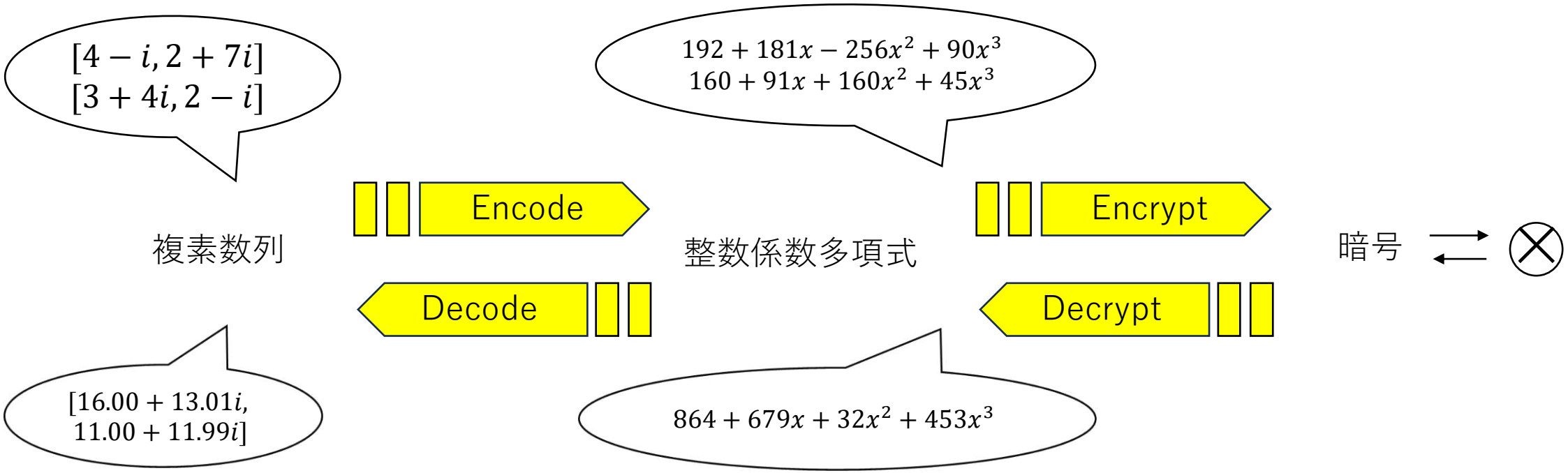
CKKS方式の暗号化/復号フレームワーク



暗号化、復号の例

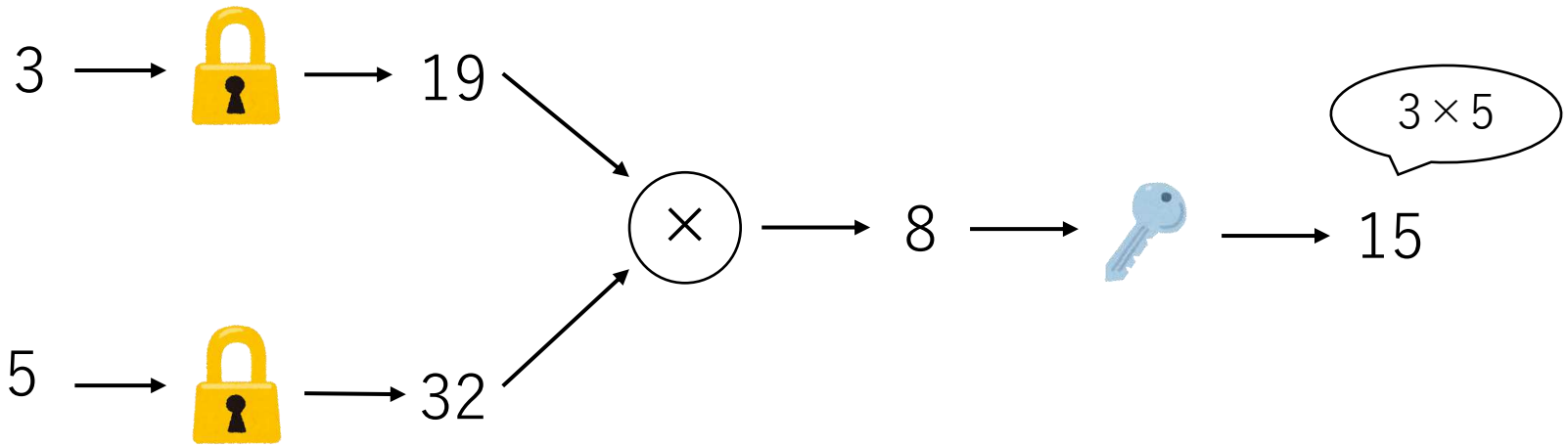

$[4-i,$
$2+7i]$
$192+181x$
$-256x^2+90x^3$
Encode
Encrypt
複素数列
整数係数多項式
暗号
Decode
Decrypt
$[4.005-1.006i,$
$1.995+6.994i]$
$192+180x$
$-256x^2+90x^3$
誤差を伴う復号

乗算の例


$[4-i, 2+7i]$
$[3+4i, 2-i]$
$192 + 181x - 256x^2 + 90x^3$
$160 + 91x + 160x^2 + 45x^3$
Encode
Encrypt
複素数列
整数係数多項式
暗号
Decode
Decrypt
$[16.00 + 13.01i,$
$11.00 + 11.99i]$
$864 + 679x + 32x^2 + 453x^3$

準同型暗号とは

準同型暗号. 暗号文のまま加算や乗算が可能な暗号.

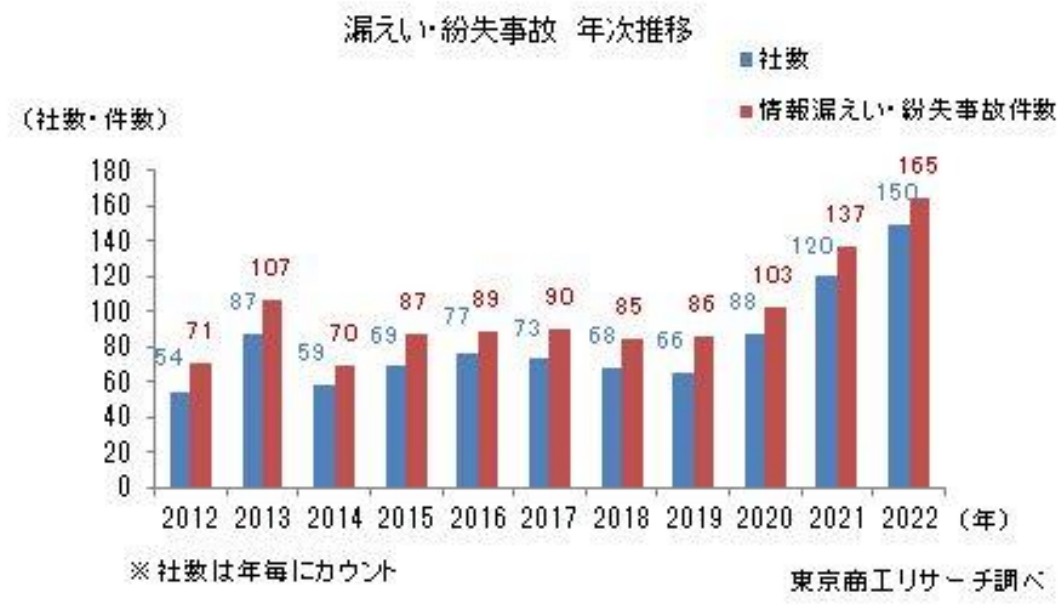


活用事例

加法準同型暗号を用いた化合物データの検索 https://www.aist.go.jp/aist_j/press_release/pr2011/pr20111101/pr20111101.html

情報漏洩件数

近年、情報漏洩件数は年々増加している.



東京商工リサーチのページから引用
https://www.tsr-net.co.jp/data/detail/1197322_1527.html

情報漏洩の事例と被害規模

個人情報の漏洩は過去に多く起こっている. 規模が大きいほど被害も大きい.

株式会社ベネッセコーポレーションの個人情報漏洩

株式会社ベネッセコーポレーションは、2014年7月、顧客の個人情報が外部に漏洩したことを公表した. 原因は業務委託先の元社員による不正行為. 流出した個人情報の数は、約2,895万件. 特別損失260億円を計上.

森永製菓株式会社の顧客情報漏洩

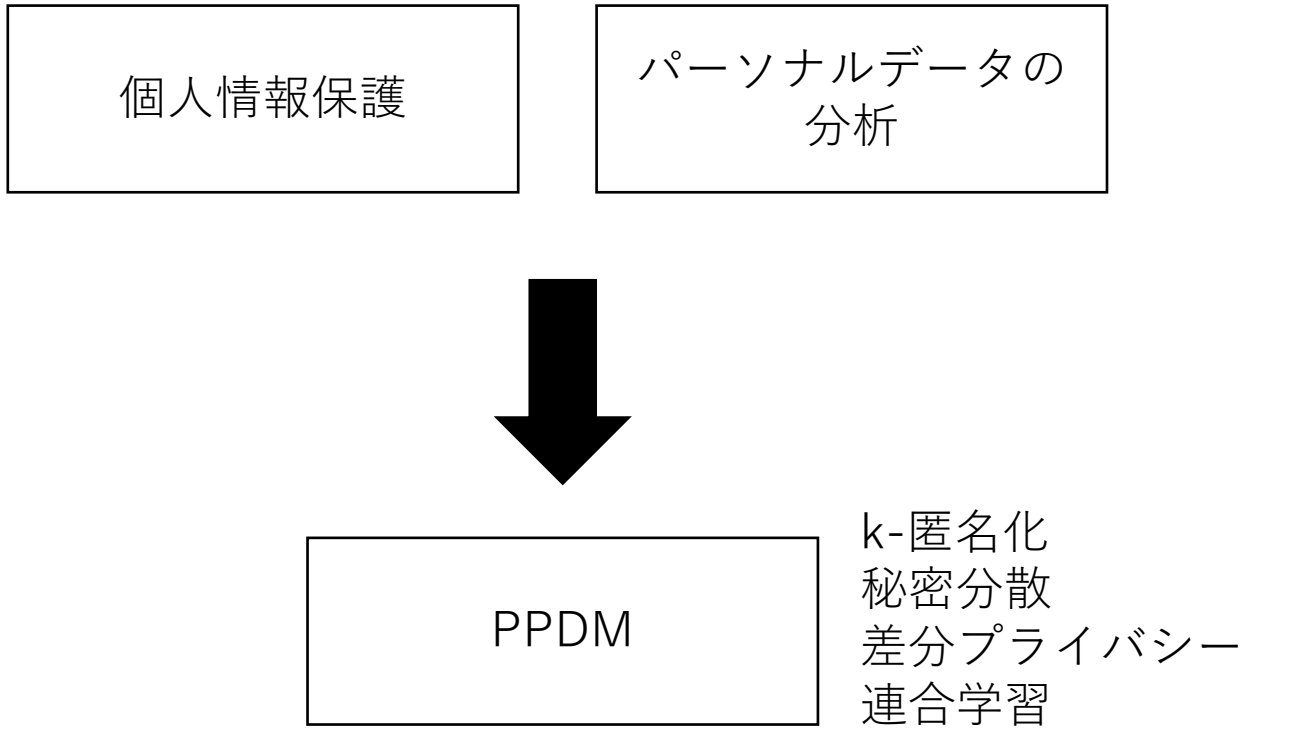
森永製菓株式会社は、2022年3月、管理運用する複数のサーバに対して、不正アクセスが発生し、顧客情報が流出した可能性があることを発表した。最大で、1,648,922人の個人情報が漏洩した可能性があるとのこと.

課題: データ分析と情報漏洩のジレンマ

データ分析の専門家の育成には膨大なコストがかかる.
データ分析の外部委託はデータを外部に渡すことから情報漏洩のリスクが高まる.

プライバシー保護データマイニング

前述の課題はプライバシー保護データマイニング技術（PPDM）により解決される.



PPDMの1技術としての準同型暗号の特徴

- 秘匿データの分析、機械学習技術への応用が可能.
  - 暗号化したまま計算を行えるため、計算サーバには暗号化済みのAIモデルとデータのみを置いておけばよく、従来と比較してデータ漏洩のリスクが少ない.

活用例1. データ分析の業務委託

- 近年機械学習を用いたデータサイエンス技術は進歩しており、高精度な予測が期待できる
- 業務委託により自社でデータサイエンティストを育成するコストを削減できる

- 個人情報の漏洩が大きな損失に
- 情報漏洩の可能性は委託先の情報管理体制に依存し、委託元は関与しづらい
- 委託先がデータを漏洩した場合、委託元の損失に

PPDM

- 委託元は暗号化したデータだけを委託先に渡し、委託先は暗号文のままデータ分析を行う
- 委託先が知れるのは暗号文のみ→委託先からのデータ漏洩を防ぐことができる
- 準同型暗号を用いることでデータ分析における機械学習技術をそのまま活用できる

活用例2. 個人情報含むデータの販売

- 自社で有益なビッグデータを保有する会社は「データ販売」を1つの事業にできる
- JRなどは加工済パーソナルデータの外部販売に意欲的

- 個人情報が特定できる形でデータを販売できない

PPDM

- データの暗号化により外部販売が可能になる
- 準同型暗号を用いることで分析時に機械学習の活用が可能になる

従来の準同型暗号の問題点

実数の暗号化に対応していない

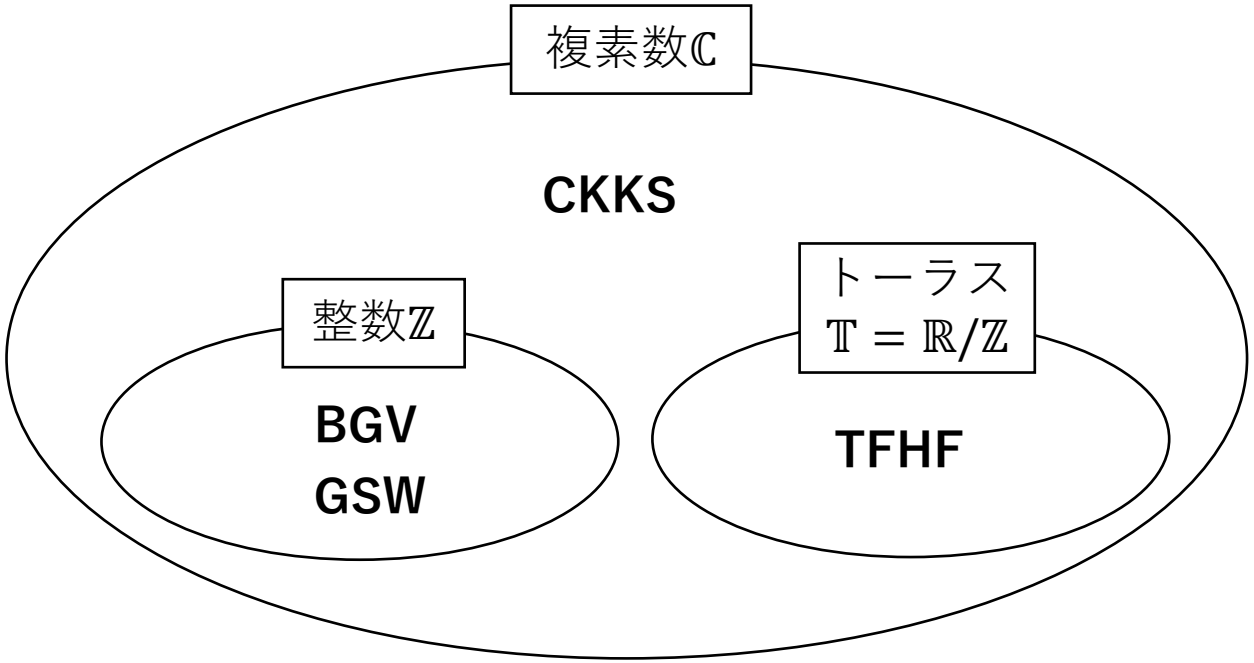
浮動小数点数を値に持つデータの分析の秘密計算に使いにくい

乗算のたびに暗号文のサイズが大きくなる

サイズmの暗号とnの暗号を乗算するとサイズ$m+n-1$の暗号が得られる
例えば、a^nを暗号化しようとするとサイズ$O(n)$の暗号となってしまう

従来の準同型暗号との違い その1

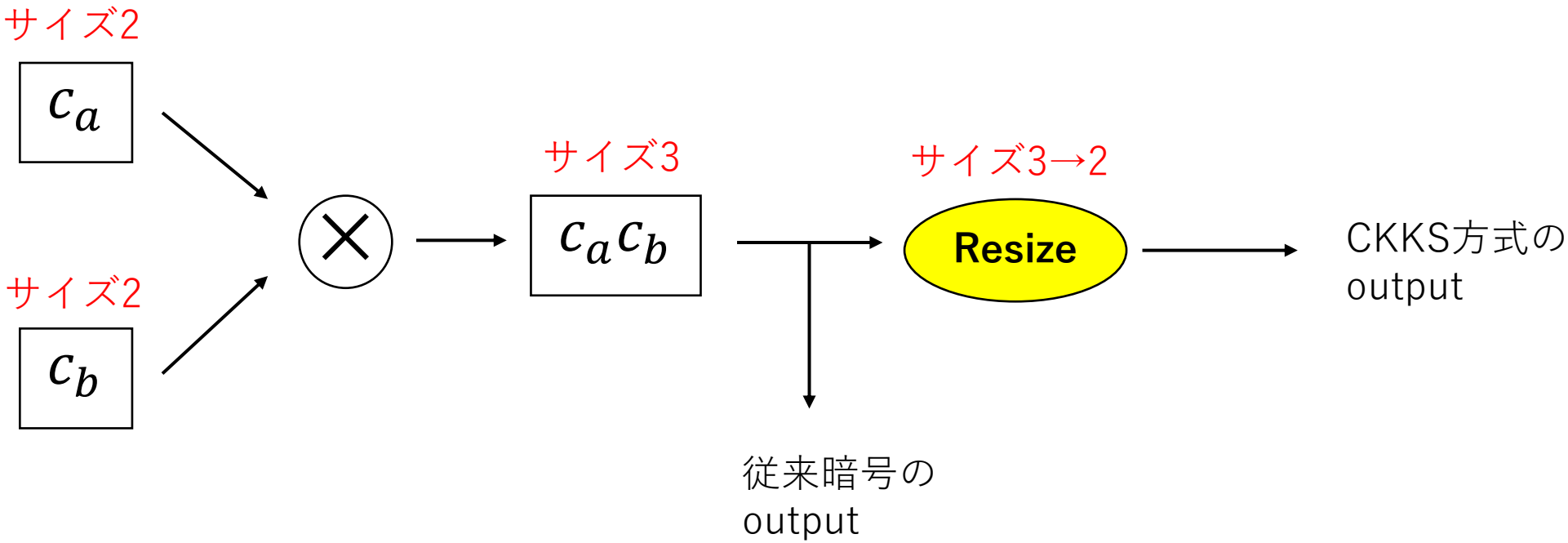
各暗号の平文空間をあらわした図.
従来の暗号とは違い, CKKS方式の暗号は複素数を平文にとることができる.
これにより、準同型暗号の深層学習への適用が現実味を帯びた.



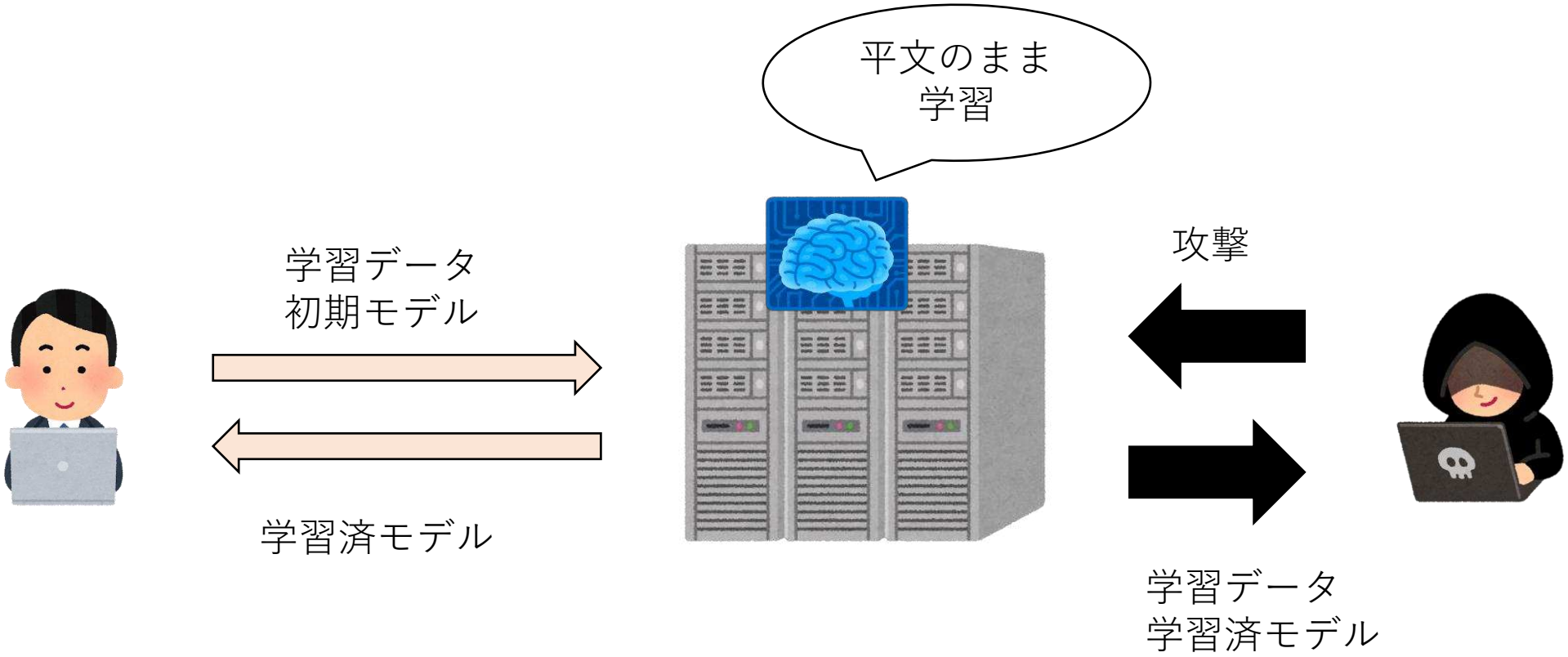
BGV [Zvika Brakerski, 2011]
GSW [Craig Gentry et al., 2013]
TFHF [Ilaria Chillotti et al., 2020]
※TFHEは[Ilaria Chillotti et al., 2016]をベースにつくられている
CKKS [Jung Hee Cheon et al., 2017]

従来の準同型暗号との違い その2

CKKS方式の暗号では, リサイズ処理を用いて乗算時の暗号文サイズ増加を抑制した.

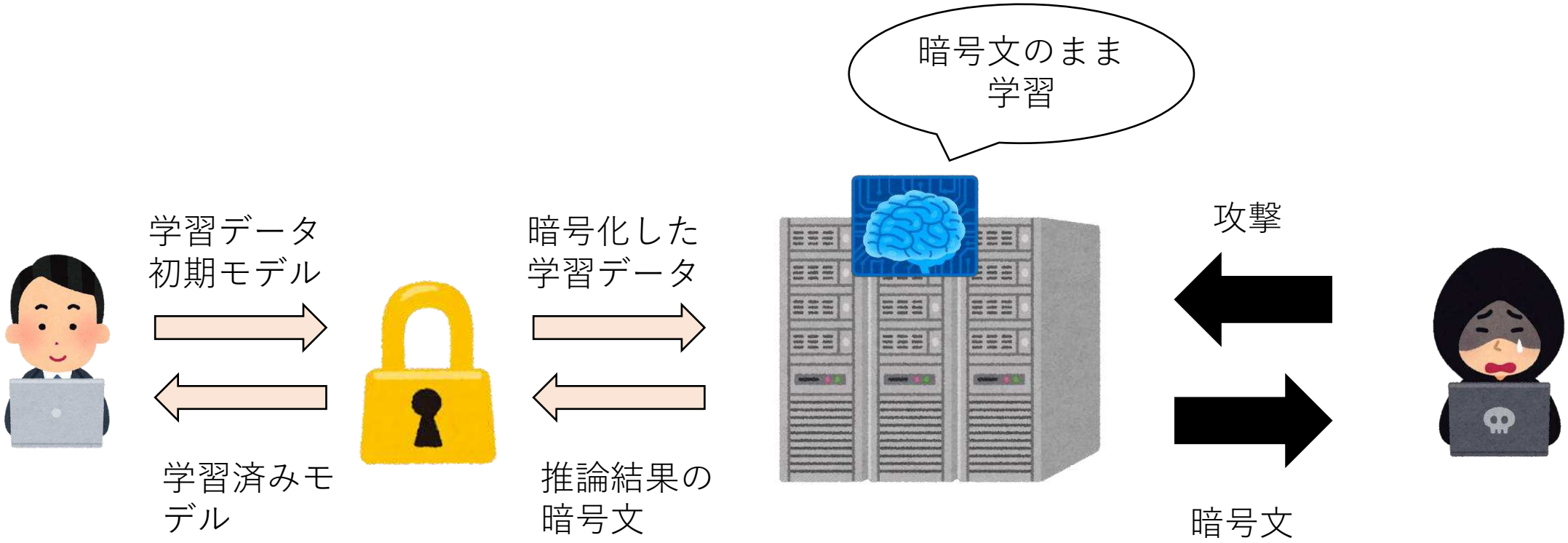


有用性（開発者視点）



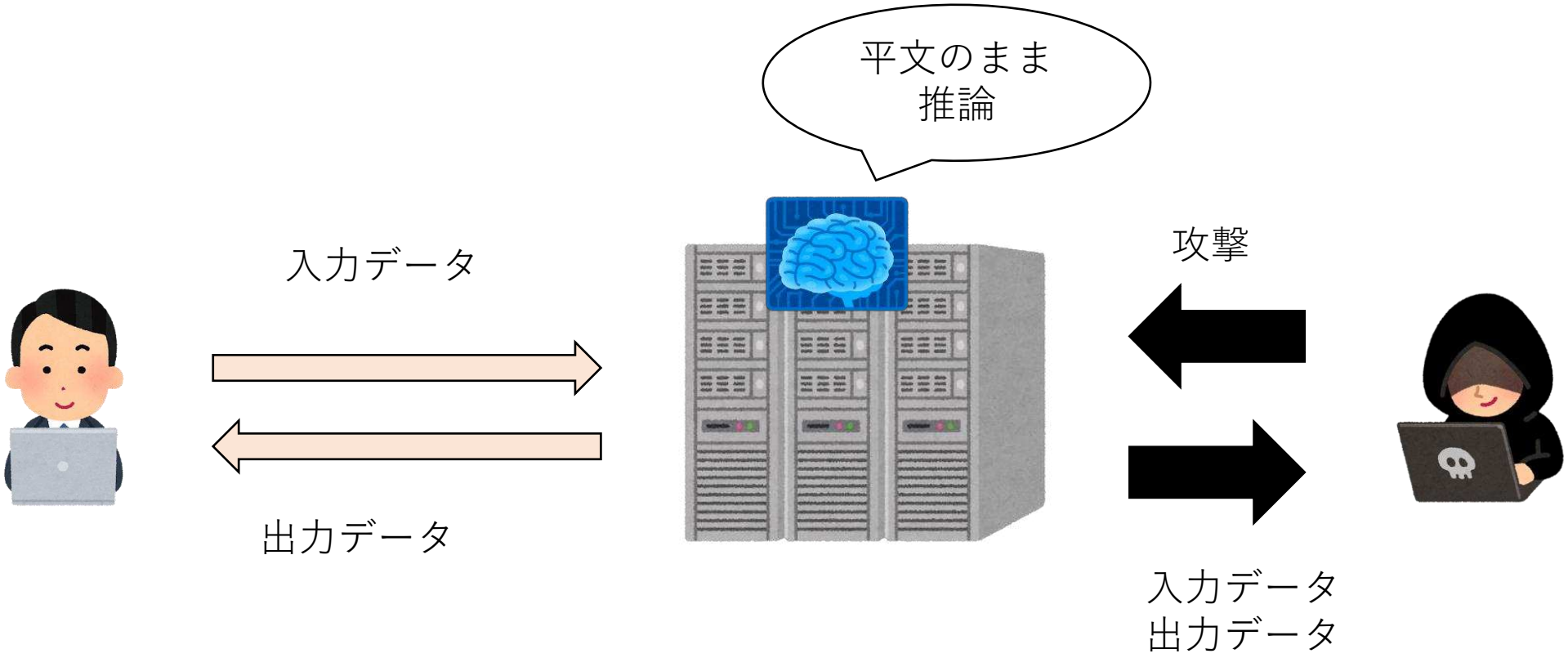
攻撃者に学習データ、学習済みモデルが漏洩する

有用性（開発者視点）



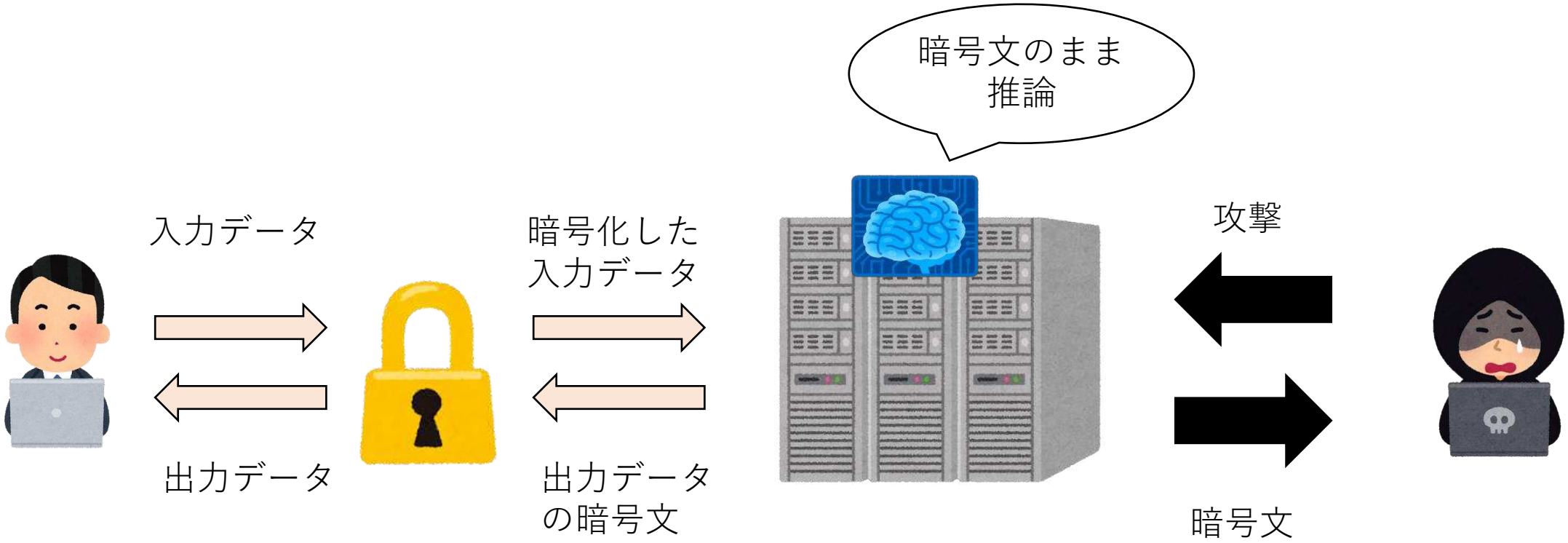
攻撃者は暗号文しか入手できない

有用性（ユーザ視点）



攻撃者に入出力データが漏洩する

有用性（ユーザ視点）



攻撃者は暗号文しか入手できない

CKKS方式の準同型暗号の設計思想

- RLWEベースの暗号フレームワークに落とし込むために、複素ベクトルと整数係数多項式を1対1に対応させ、近似的な変換（Encode）と逆変換（Decode）を可能にした.
- 乗算時の暗号文サイズ増加を抑えるために、評価鍵を用いたリサイズ処理を導入した.
- 平文に含まれるノイズを小さくするため、BGV方式の暗号[Brakerski et al., 2012]で提案されたmodulus switchingと同様の処理であるRescaleを導入した.

定義

- $J \coloneqq \sqrt{-1}$
- $N \in 2 * \mathbb{N}$ 平文の次数の2倍
- $\xi \coloneqq e^{\frac{\pi}{N}J} \in \mathbb{C}$ 1の$2N$乗根
- $A = \left\{A_{ij} \coloneqq \xi^{(2i-1)(j-1)}\right\} \in \mathbb{C}^{N \times N}$ ヴァンデルモンド行列
- $b_j \coloneqq \left(A_{1j}, \ldots, A_{Nj}\right) = (\xi^{j-1}, \xi^{3(j-1)}, \ldots, \xi^{(2N-1)(j-1)})$ Aのj列目
- $\Delta_{scale} \in \mathbb{N}$ 十分大きな数
- $\mathbb{Z}^{(n)} \coloneqq \mathbb{N} \cap [-\frac{n}{2}, \frac{n}{2})$

Encode, Decodeの定義

Encode

$$\mathbb{C}^{\frac{N}{2}} \underset{\pi}{\overset{\pi^{-1}}{\rightleftarrows}} \mathbb{C}^{N} \underset{\sigma}{\overset{\sigma^{-1}}{\rightleftarrows}} R \underset{\div \Delta_{scale}^{k}}{\overset{\times \Delta_{scale}}{\rightleftarrows}} R$$

Decode

※ $R = \mathbb{Z}[X]/(X^N + 1)$

π^{-1}: $[z_1, z_2, \ldots, z_{\frac{N}{2}}] \rightarrow [z_1, \ldots, z_{\frac{N}{2}}, \overline{z_{\frac{N}{2}}}, \ldots, \overline{z_1}]$

π: $[z_1, \ldots, z_{\frac{N}{2}}, \overline{z_{\frac{N}{2}}}, \ldots, \overline{z_1}] \rightarrow [z_1, z_2, \ldots, z_{\frac{N}{2}}]$

σ^{-1}: $z \rightarrow \frac{<z,b_1>}{<b_1,b_1>} + \frac{<z,b_2>}{<b_2,b_2>} X + \cdots + \frac{<z,b_N>}{<b_N,b_N>} X^{N-1}$

σ: $f \rightarrow (f(\xi), f(\xi^3), \ldots, f(\xi^{2N-1}))$

鍵生成、暗号化、復号の定義

鍵生成

$q_l \leftarrow q_0 p^l \quad (l = 0, \dots, L)$
$a \in \mathbb{Z}_{q_L}[X]/(X^N + 1),\ a' \in \mathbb{Z}_{P \cdot q_L}[X]/(X^N + 1)$
$e, e' \sim \mathcal{N}(0, \sigma^2)$
秘密鍵: $sk = s \in \mathbb{Z}^{(3)}[X]/(X^N + 1)$
公開鍵: $pk = (p_1, p_2) = (-a \cdot s + e, a)_{q_L}$
評価鍵: $evk = (-a' \cdot s + e' + Ps^2, a')_{P \cdot q_L}$

公開鍵pkから秘密鍵skを求めるのは困難（後述）

暗号化(Encrypt)

暗号文: $c = (c_1, c_2) = ([v \cdot p_1 + e_1 + \mu]_{q_L}, [v \cdot p_2 + e_2]_{q_L})$

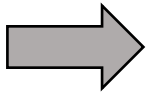
復号(Decrypt)

復号文: $\mu' = [c_1 + c_2 \cdot s]_{q_l}$

$L - l$回の
リスケール

安全性の確認

公開鍵 $pk = (p_1, p_2) = ([-a \cdot s + e]_{q_L}, a)$ から秘密鍵 s が求められるか



$[-a \cdot s + e]_{q_L}$ と $a = [a]_{q_L}$ から s が求められるか

難しい

RLWE仮定

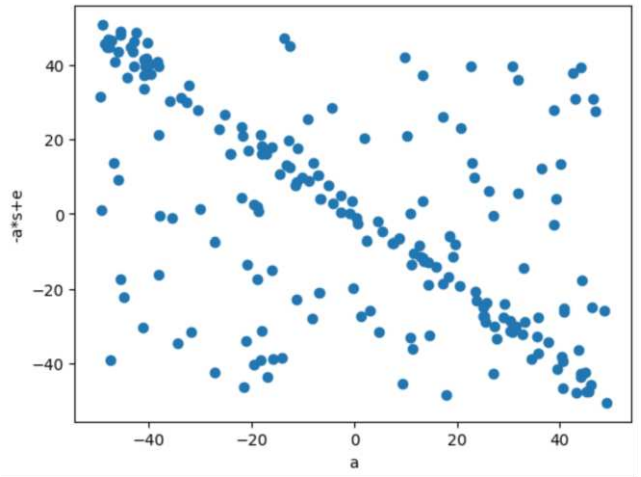
集合$\{([a_i]_{q_L}, [-a_i \cdot s + e_i]_{q_L})\}$ と集合$\{([a_i]_{q_L}, [b_i]_{q_L})\}$を混ざっている状態から分離することは難しい。

CKKS方式の暗号はRLWE仮定をもとに安全性が保証されている.

RLWE仮定にはちょうどいいσが重要

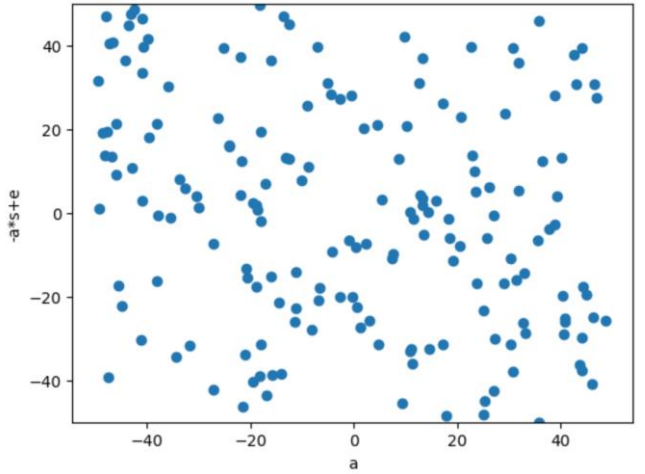
簡単のため$a, s, e \in \mathbb{R}$とする.

σ小さい



- 分離しやすい＝秘密鍵sの値が解読されやすい
- 平文に含まれるノイズが小さい＝誤差が無視できるくらい小さい

σ大きい



- 分離しにくい＝秘密鍵sの値が解読されにくい
- 平文に含まれるノイズが大きい＝誤差が無視できないほど大きい

Resize処理を用いた暗号文の乗算の流れ

暗号文c_a, c_bの乗算を考える.

$$
\begin{aligned}
&Decrypt(c_a) \cdot Decrypt(c_b) \\
&= [c_{a1} + c_{a2} \cdot s]_{q_l} \cdot [c_{b1} + c_{b2} \cdot s]_{q_l} \\
&= [c_{a1} \cdot c_{b1} + (c_{a1}c_{b2} + c_{b1}c_{a2}) \cdot s + c_{a2} \cdot c_{b2} \cdot s^2]_{q_l} \\
&= [c'_0 + c'_1 s + c'_2 s^2]_{q_l}
\end{aligned}
$$

このままだと乗算暗号文は(c'_0, c'_1, c'_2)となり, c_a, c_bと比較してサイズが1増える.
サイズを2に保つために, 以下の式であらわされる評価鍵evkを用いる.

Resize

$$evk = (-a's + e' + Ps^2, a') \quad ※P\text{は十分大きくとる}$$

ここでc_a, c_bを乗算して得られる暗号として以下に示す(d_0, d_1)を出力する.

$$(d_0, d_1) = (c'_0, c'_1) + [P^{-1} \cdot c'_2 \cdot evk] \qquad \rightarrow \text{出力}$$

加算・乗算暗号の復号可能性

Decryptにより生じる誤差が十分小さいかどうかの確認

加算暗号の場合

$$
\begin{aligned}
&Decrypt(c_a + c_b) \\
&= Decrypt(c_a) + Decrypt(c_b) \\
&= (p_a + e_a) + (p_b + e_b) \\
&= p_a + p_b + (e_a + e_b)
\end{aligned}
$$

小さい

➡ $\approx p_a + p_b$ にできる

乗算暗号の場合

$$
\begin{aligned}
&Decrypt(c_a) \cdot Decrypt(c_b) \\
&= (p_a + e_a)(p_b + e_b) \\
&= p_a p_b + (p_a e_b + p_b e_a + e_a e_b)
\end{aligned}
$$

小さくない…

➡ $\approx p_a p_b$ にできない

乗算暗号文のノイズを減らす方法

rescaling	$c \leftarrow \left\lfloor \frac{q_{l'}}{q_l} c \right\rceil \bmod q_{l'}$
simple modular reduction	$c \leftarrow c \bmod q_{l'}$

- rescalingにより平文に含まれるノイズを$\frac{q_{l'}}{q_l}$の比で減らすことができる.
- 同時に、平文そのものも$\frac{q_{l'}}{q_l}$の比で減少する. 復号するときは注意が必要.
- レベルの異なる暗号どうしで計算するときは計算前にレベルの高い暗号を低いほうに合わせる必要があるが、原論文ではrescalingではなくsimple modular reductionを用いている.

深層学習への適用に立ちはだかる壁

- CKKS方式の準同型暗号は乗算回数に制限がある（制限回数を超えて乗算を行うと誤差が無視できないほどに大きくなる）
- 一般に, ニューラルベースのAIモデルの秘密学習/推論が普及していないのは乗算回数に制限があることが理由として挙げられる.
- 以降では, ニューラルネットワークで用いる計算に対して加算, 乗算回数を見積もる.

NNで要求される関数

計算グラフ	**必要な計算（準伝播）**	**必要な計算（逆伝播）**
加算グラフ	加算	
乗算グラフ	乗算	
指数グラフ	指数計算	
対数グラフ	対数計算	除算
ReLUグラフ	max(0,x)計算	

NNで要求される関数

機能	関数名	必要な計算
線形変換	Affine変換	加算、乗算
活性化関数	ReLU	Max計算
	Softmax	指数計算、除算、加算
損失関数	CrossEntropyLoss	対数計算、乗算、加算
	MeanSquaredError	乗算、加算

加算、乗算以外は秘密計算が難しい…

⬇

すべての計算を加算、乗算のみを用いた計算にすればよい.

和積のみでの計算（多項式近似）

指数, 対数, max計算は多項式に近似できる.

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \cdots$$

$$\log(1+x) = \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{n} x^n = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \cdots \quad (|x| < 1)$$

$$\max(0, x) \approx Swish(x) = \frac{x}{1 + e^{-x}} = \quad \text{（略）}$$

※Swish [Ramachandran et al., 2017]

乗算回数の多さから、実用的ではないとする意見もある.

和積のみでの計算（除算）

$\hat{x} = 1 - x, |\hat{x}| \leq \frac{1}{2}$ を満たすとき, 十分大きい r に対して以下の式が成り立つ.

$$x^{-1} \approx \prod_{j=0}^{r-1} (1 + \hat{x}^{2^j})$$

和積のみでx^{-1}をあらわすことができている.

乗算回数（多項式）

求めるもの: d次の多項式の代入計算に必要な乗算回数

乗算回数: $(\lfloor \log d \rfloor + 1) + \sum_{i=1}^{d} f(i)$ （$f(i)$: iの2進数表記における1の個数）

導出: 高速指数計算法を用いて $x, x^2, x^4, x^8, \ldots, x^{2^{\lfloor \log d \rfloor}}$ を計算する. この計算は$\lfloor \log d \rfloor + 1$回の乗算が必要になる。
その後, 多項式の代入計算を行う. i次の項の計算には$f(i)$回の乗算が必要になる.

$$a_0 + \underbrace{a_1 x}_{1\text{回}} + \underbrace{a_2 x^2}_{1\text{回}} + \underbrace{a_3 x^3}_{2\text{回}} + \underbrace{a_4 x^4}_{1\text{回}} + \cdots + a_d x^d$$

例えば, $d = 4$のときは合計8回の乗算が必要となる.

まとめ

- 準同型暗号の活用によりデータのプライバシーを保護しながら分析を行うことができる.
- CKKS方式の準同型暗号はRLWE仮定に基づく複素数暗号化フレームワーク.
- CKKS方式の暗号は近似準同型暗号.
- リサイズ処理により乗算暗号文のサイズ爆発を抑えることに成功.
- リスケール処理により計算精度を保ったまま誤差を小さくできる. （回数制限あり）
- ニューラルネットワークへの応用には乗算回数の壁がある. 実用化にはまだ遠い.

Appendix

Encode(z) の各係数が実数であることの証明

命題. $Encode(z) \in \mathbb{R}[x]/(x^{N+1}+1)$

以下、命題の証明

$$
\begin{aligned}
&\frac{<z', b_i>}{<b_i, b_i>} \\
&= \frac{1}{N}(z_1\overline{A_{1i}} + \cdots + z_{\frac{N}{2}}\overline{A_{\frac{N}{2}i}} + \overline{z_{\frac{N}{2}}}\overline{A_{\left(\frac{N}{2}+1\right)i}} + \cdots + \bar{z_1}\overline{A_{Ni}}) \\
&= \frac{1}{N}(z_1\overline{A_{1i}} + \cdots + z_{\frac{N}{2}}\overline{A_{\frac{N}{2}i}} + \overline{z_{\frac{N}{2}}}A_{\frac{N}{2}i} + \cdots + \bar{z_1}A_{1i}) \\
&= \frac{1}{N}\{(z_1\overline{A_{1i}} + \bar{z_1}A_{1i}) + \cdots + (z_{\frac{N}{2}}\overline{A_{\frac{N}{2}i}} + \overline{z_{\frac{N}{2}}}A_{\frac{N}{2}i})\} \\
&= \frac{1}{N}\{Re(z_1\overline{A_{1i}}) + \cdots + Re(z_{\frac{N}{2}}\overline{A_{\frac{N}{2}i}})\} \\
&\in \mathbb{R}
\end{aligned}
$$

$Encode(z)$ において x^{i-1} の係数は $\frac{<z', b_i>}{<b_i, b_i>}$ とあらわされるので, $Encode(z)$ の各係数は実数である. ■

Decode(Encode(z))≈z の証明 補題1

補題1. $1 \leq k_1 \leq N, 1 \leq k_2 \leq N$ のとき, $\sum_{i=1}^{N} A_{k_1 i} A_{k_2 i} = \begin{cases} N, & (k_1 + k_2 = N + 1) \\ 0, & (k_1 + k_2 \neq N + 1) \end{cases}$

以下補題1の証明

$$\sum_{i=1}^{N} A_{k_1 i} A_{k_2 i} = \sum_{i=1}^{N} \xi^{(2k_1+2k_2-2)(i-1)}$$

これは$k_1 + k_2 = N + 1$ のときN となる. 以降ではそうでないときを考える. $\xi^{(2k_1+2k_2-2)(i-1)}, i = 1, \ldots, N$ について、周期を$N'(\leq N)$とおく. $\xi^{(2k_1+2k_2-2)(i-1)}, i = 1, \ldots, N'$ について、これらは互いに異なる1のN'乗根であるから, 方程式$z^{N'} = 1$ の解である. よって, 解と係数の関係によりその総和$\sum_{i=1}^{N'} \xi^{(2k_1+2k_2-2)(i-1)}$は$0$ になる. したがって, $\sum_{i=1}^{N} \xi^{(2k_1+2k_2-2)(i-1)} = 0$ が成り立つ. ■

Decode(Encode(z))≈z の証明 補題2

$p(x) \coloneqq \sum_{i=1}^{N} \lfloor \frac{<z', b_i>}{<b_i, b_i>} \rceil x^{i-1}$ とおく.

補題2. $p(A_{k2}) \approx \Delta_{scale} z_k$

以下補題2の証明

$$
\begin{aligned}
& p(A_{k2}) \\
&= \sum_{i=1}^{N} \lfloor \frac{<z', b_i>}{<b_i, b_i>} \rceil A_{k2}^{i-1} \\
&\approx \sum_{i=1}^{N} \frac{<z', b_i>}{<b_i, b_i>} A_{k2}^{i-1} \\
&= \sum_{i=1}^{N} \frac{<z', b_i>}{<b_i, b_i>} A_{ki} \\
&= \frac{\Delta_{scale}}{N} \sum_{i=1}^{N} \{ z_1 \overline{A_{1i}} + \cdots + z_{\frac{N}{2}} \overline{A_{\frac{N}{2} i}} + \overline{z_{\frac{N}{2}}} \overline{A_{\left(\frac{N}{2}+1\right) i}} + \cdots + \bar{z_1} \overline{A_{Ni}} \} A_{ki} \\
&= \frac{\Delta_{scale}}{N} \sum_{i=1}^{N} \{ z_1 A_{Ni} + \cdots + z_{\frac{N}{2}} A_{\left(\frac{N}{2}+1\right) i} + \overline{z_{\frac{N}{2}}} A_{\frac{N}{2} i} + \cdots + \bar{z_1} A_{1i} \} A_{ki} \quad (\because \overline{A_{ji}} = A_{(N-j+1)i}) \\
&= \frac{\Delta_{scale}}{N} \{ z_1 \sum_{i=1}^{N} A_{Ni} A_{ki} + \cdots + z_{\frac{N}{2}} \sum_{i=1}^{N} A_{\left(\frac{N}{2}+1\right) i} A_{ki} + \overline{z_{\frac{N}{2}}} \sum_{i=1}^{N} A_{\frac{N}{2} i} A_{ki} + \cdots + \bar{z_1} \sum_{i=1}^{N} A_{1i} A_{ki} \} \\
&= \frac{\Delta_{scale}}{N} z_k \sum_{i=1}^{N} A_{(N-k+1)i} A_{ki} \qquad (\because \text{補題1より}) \\
&= \frac{\Delta_{scale}}{N} z_k N \\
&= \Delta_{scale} z_k \ \blacksquare
\end{aligned}
$$

Decode(Encode(z))≈z の証明

命題. $Decode\big(Encode(z)\big) \approx z$

以下、命題の証明

$$
\begin{aligned}
& Decode(Encode(z)) \\
&= Decode(p(x) \coloneqq \textstyle\sum_{i=1}^{N} \frac{<z', b_i>}{<b_i, b_i>} x^{i-1}) \\
&= \Delta_{scale}^{-1}(p(A_{12}), \dots, p\left(A_{\frac{N}{2}2}\right)) \\
&\approx \Delta_{scale}^{-1}(\Delta_{scale} z_1, \dots, \Delta_{scale} z_{\frac{N}{2}}) \qquad (\because \text{補題2より} p(A_{k2}) = \Delta_{scale} z_k) \\
&= (z_1, \dots, z_{\frac{N}{2}}) \\
&= z \ \blacksquare
\end{aligned}
$$

Decrypt(Encrypt(p))≈p の証明

命題. $Decode(Encrypt(p)) \approx p$

以下、命題の証明

$$
\begin{aligned}
&Decrypt(Encrypt(p)) \\
&= [c_1 + c_2 \cdot s]_{q_l} \\
&= \left[[v \cdot p_1 + e_1 + p]_{q_L} + [v \cdot p_2 + e_2]_{q_L} \cdot s\right]_{q_l} \\
&= \left[\left[v \cdot [-a \cdot s + e]_{q_L} + e_1 + p\right]_{q_L} + [v \cdot a + e_2]_{q_L} \cdot s\right]_{q_l} \\
&= \left[\left[\left[v \cdot [-a \cdot s + e]_{q_L} + e_1 + p\right]_{q_L} + [v \cdot a + e_2]_{q_L} \cdot s\right]_{q_L}\right]_{q_l} \\
&= \left[[v \cdot (-a \cdot s + e) + e_1 + p + (v \cdot a + e_2) \cdot s]_{q_L}\right]_{q_l} \\
&= \left[[v \cdot e + e_1 + p + e_2 \cdot s]_{q_L}\right]_{q_l} \\
&= [v \cdot e + e_1 + p + e_2 \cdot s]_{q_l} \qquad (\because q_l \text{は} q_L \text{の約数}) \\
&= p + e' \\
&\approx p \ \blacksquare
\end{aligned}
$$

Decode(p+e)≈Decode(p) の証明

命題. $e \ll p$ のとき, $Decode(p+e) \approx Decode(p)$

以下、命題の証明

$e'(a)$ は$p(a)$ に比べて小さいので, $p(a) + e'(a) \approx p(a)$ が成り立つ.
よって,

$$
\begin{aligned}
&Decode(p+e') \\
&= \Delta_{scale}^{-1}((p+e')(A_{12}), \dots, (p+e')\left(A_{\frac{N}{2}2}\right)) \\
&\approx \Delta_{scale}^{-1}(p(A_{12}), \dots, p\left(A_{\frac{N}{2}2}\right)) \\
&= Decode(p) \ \blacksquare
\end{aligned}
$$

加法準同型性の証明

命題. $Decrypt(c_a + c_b) = Decrypt(c_a) + Decrypt(c_b)$

以下、命題の証明

$$
\begin{aligned}
&Decrypt(c_a) + Decrypt(c_b) \\
&= [c_{a1} + c_{a2} \cdot s]_{q_l} + [c_{b1} + c_{b2} \cdot s]_{q_l} \\
&= [(c_{a1} + c_{b1}) + (c_{a2} + c_{b2}) \cdot s]_{q_l} \\
&= Decrypt(c_a + c_b) \ \blacksquare
\end{aligned}
$$

暗号文の乗算の流れ

暗号文c_a, c_bの乗算を考える.

$$
\begin{aligned}
&Decrypt(c_a) \cdot Decrypt(c_b) \\
&= [c_{a1} + c_{a2} \cdot s]_{q_l} \cdot [c_{b1} + c_{b2} \cdot s]_{q_l} \\
&= [c_{a1} \cdot c_{b1} + (c_{a1}c_{b2} + c_{b1}c_{a2}) \cdot s + c_{a2} \cdot c_{b2} \cdot s^2]_{q_l} \\
&= [c'_0 + c'_1 s + c'_2 s^2]_{q_l}
\end{aligned}
$$

このままだと乗算暗号文は(c'_0, c'_1, c'_2)となり, c_a, c_bと比較してサイズが1増える.
サイズを2に保つために, 以下の式であらわされる評価鍵evkを用いる.

$$evk = (-a's + e' + Ps^2, a') \quad ※P\text{は十分大きくとる}$$

ここでc_a, c_bを乗算して得られる暗号として以下に示す(d_0, d_1)を出力する.

$$(d_0, d_1) = (c'_0, c'_1) + [P^{-1} \cdot c'_2 \cdot evk]$$

乗法準同型性の証明

命題. $Decode((d_0, d_1)) \approx Decrypt(c_a) \cdot Decrypt(c_b)$

以下、命題の証明

$$
\begin{aligned}
&Decrypt\big((d_0, d_1)\big) \\
&= d_0 + d_1 s \\
&\approx \{c_0' + P^{-1}c_2'(-a's + e') + c_2's^2\} + (c_1's + P^{-1}c_2'a's) \\
&= (c_0' + c_1's + c_2's^2) + P^{-1}c_2'e' \\
&\approx Decrypt(c_a) \cdot Decrypt(c_b) \ \blacksquare
\end{aligned}
$$