ランプ型秘密分散の部分情報漏洩の性質を利用した低通信量で実現できる安全な情報共有法の検討

アフマド アクマル アミヌディン[1,a)]　藤沢 匡哉[1,b)]

概要：災害時における安否情報共有システムの実現において，被害者の個人情報を守りつつ安全及び効率的に情報を送受信できる方法が求められる．機密性を向上させる方法として，暗号鍵を用いたデータを暗号化する方法とデータを複数の断片に分割する秘密分散が知られている．(k, n) しきい値秘密分散は，n 個の分散値を送信する必要があるため，符号化効率が悪く，通信量，送受信機での記憶容量も大きいという課題がある．この改善法として，L 個の情報を同時に送信できる (k, L, n) ランプ型秘密分散や非対称秘密分散などが提案されているが，依然として通信量や記憶容量が大きく，省電力やコストの面で問題が大きい．本研究では，通信量や記憶容量を削減しながら，安全性の向上も実現できる方法を提案する．具体的には，$(4, 2, 4)$ ランプ型秘密分散法における部分的な情報漏洩の性質を活用し，非対称秘密分散における一部の分散値をローカルで計算できる性質と組合わせ，暗号文サイズが平文サイズと同程度となる情報共有方式を検討する．また，必要なパラメータを事前に設定することによって，事前処理の負担を大きくする代わりに情報共有と復元処理の計算を軽減し，消費電力を極力小さくすることができた．次に，提案方式の安全性評価を行い，強い安全性を実現できることを示す．最後に，IoT デバイスを用いて，計算量の低減をシミュレーションにより確認し，提案方式の有効性を示す．

キーワード：秘密分散，非対称処理，情報共有，低通信量，符号化効率，IoT デバイス

Communication-Efficient Information Sharing using Partial Information Leakage in Ramp Secret Sharing

Ahmad Akmal Aminuddin[1,a)]　Masaya Fujisawa[1,b)]

Abstract: In this study, we propose a secure and communication-efficient information sharing method. Specifically, we exploit the partial information leakage property of $(4, 2, 4)$ ramp secret sharing and combine it with the property of asymmetric secret sharing that some shares can be calculated locally to reduce the transmission cost. Furthermore, by precomputing the necessary parameters, we were able to reduce the computation costs for information sharing and reconstructions processes, thereby minimizing power consumption. Finally, we confirmed the reduction in computational complexity through simulations using IoT devices.

Keywords: Secret sharing, asymmetric process, information sharing, low communication, coding efficiency, IoT device

1. はじめに

日本では，地震や火山噴火などの自然災害が多発する国であり，これらの災害による，広範囲で通信が途絶する状況が発生した場合，被災者と災害対策本部との間で，情報を確実に安全に交換することが救助活動の支援に不可欠である．そのため，近年では，ドローンを用いて一時的な通信網を構築し，被害者の安否情報を効率的に収集できる研究が盛んに行われる [1], [2]．ドローンを用いた通信は簡単

[1] 東京理科大学
Tokyo University of Science
[a)] ahmad.amin@rs.tus.ac.jp
[b)] fujisawa@rs.tus.ac.jp

に実現できるが，ドローンの墜落などにより通信データへのアクセスも容易である．特に，本研究で想定しているシステムでは，安否確認用小型端末から個人 ID，位置情報，健康状態情報などの個人情報を送信することを考えているため，送信する個人情報を保護する必要がある．さらに，災害時におけるユーザの通信端末のバッテリー寿命を考慮すると，複雑な暗号化処理を必要とせず，軽量で実現できる情報共有法が求められる．

情報の保護を実現できる情報送信手法として，Diffie-Hellman 鍵共有方式に基づく手法が知られている．しかし，これらの手法は，Diffie-Hellman 問題に基づく計算量的な安全性しか実現できない．これに対し，Blom は情報理論的安全な情報共有法を提案したが，信頼できる機関が，ユーザごとに一意の多項式を構築し，その $k+1$ 個の係数を安全な方法を介して送信する必要があり，高い計算と通信量が必要ため，本研究で想定する低計算能力を持つ小型端末のバッテリーの長寿命化に向いてない．一方，無人航空機 (UAV) ネットワークの特性に注目した安全な情報共有法の研究も近年注目を集めている．Altawy らは，UAV システムのセキュリティとプライバシーに関するさまざまな懸念事項を紹介した [3]．Liu らは，盗聴や悪意のある妨害から UAV 通信システムを保護するための物理層セキュリティの対策法を提案した [4]．Abdallah らは，軽量の Ring Learning with Error (RLWE) を使用し，低計算量で送受信する情報を保護する方法を提案した [5]．しかし，これまでのほとんどの研究は，公開鍵と共通鍵暗号を基づいた研究が多い．対して，岩村らは，鍵を必要としない，低計算量を持つ Shamir の (k,n) 法を用いた秘密計算を活用し，IoT 環境に適する安全な認証および通信方法を提案した [6]．Shamir の (k,n) 法は，(k,n) しきい値秘密分散の一つであり，ある秘密情報 m を複数の異なる値（分散値と呼ぶ）に分割し，複数の独立したサーバに分散するため [7]，従来の鍵を用いた暗号方式よりも通信量が多いと知られている．川原らは，Shamir の $(2,n)$ 法を用いた，ユーザアクセス制御を備えた安全な情報共有システムを提案した [8]．また，[8] における安全性の課題を解決するために，ランダム化処理や改ざん検知機能を追加した手法も提案された [9][10] が，いずれの手法でも，ドローンネットワークに送信する分散値の数が大きいという課題がまだ残っている．

そこで，本研究で想定する小型端末の情報共有と送信する処理に必要となる消費電力を極力小さくするために，事前処理の負担を大きくする代わりに情報共有と復元処理を軽減し，かつ，ランプ型秘密分散の部分的な情報漏洩と非対称秘密分散の特徴を組合わせて，暗号文サイズが平文サイズと同程度となる情報共有法を検討する．本研究の主な貢献は以下に示す．

- 非対称秘密分散の特性を拡張し，事前に共有される鍵のペアを用いて，ローカルで分散値の計算ができる処理を導入する方法を検討し，部分的に情報が漏洩する $(4,2,4)$ ランプ型秘密分散法の特徴と組み合わせて，送信する分散値を平文とほぼ同程度にできる情報共有法を 3 つ提案する．これにより，秘密分散における低計算量の利点を実現しながら，問題であった通信量も減らすことができる．また，提案した 3 つの手法を従来法と比較し，小型 IoT デバイスを用いたシミュレーションにより実行時間を評価し，高速であることを示す．計算処理の低減に伴って回路規模や端末の基盤をさらに小型化が可能となることにより，消費電力が抑えられることも期待できる．

2. 事前準備

2.1 記号定義

本論文で使用される記号及びその意味を以下にまとめる．

- Z_p ：位数が素数 p である有限体
- n ：分散値の数
- k ：しきい値
- $\mathcal{C}$ ：信頼できる災害センター
- $\mathcal{D}$ ：ドローン集合 $\mathcal{D}=\{D_1,\ldots,D_h\}$, $|\mathcal{D}|=h$
- $\mathcal{P}$ ：ユーザの集合 $\mathcal{P}=\{P_1,\ldots,P_m\}$, $|\mathcal{P}|=m$
- $id[i]$: ユーザ P_i の識別子．$1\leq i\leq m$ とする

2.2 システムモデル

本研究では，災害時の通信途絶した際でも，低電力モードでドローン中継器に接続し，安否情報を情報収集センターに送信できる小型な情報発信端末に適する情報共有法の開発を目指している．想定するシステムモデルとしては，[11] をもとに，3 つのエンティティを想定する：(1) 信頼できるセンター $\mathcal{C}$，(2) m 人のユーザ集合 $\mathcal{P}$ と (3) h 台のドローン集合 $\mathcal{D}$．被害を受けた地域で，h 台ドローンによって一時的なネットワークが構成されるとする．また，ユーザ $P_i\in\mathcal{P}$ は，他のユーザ $P_j\in\mathcal{P}$，また信頼できる災害センター $\mathcal{C}$ と直接通信ができず，送信したい安否情報をすべてドローンに転送する必要があると仮定する．さらに，ユーザ P_i は，t 回目の情報共有する際に，自身の安否情報 $m_{i,t}$ を $\mathcal{C}$ とのみ共有したいとし，他のユーザとの共有を想定しないとする．また，$\mathcal{C}$ は信頼できる者とし，受信した情報を改ざんしたりすることはない（ユーザ P_i も同様）．対して，ドローンは空中に位置しているため，信頼できないものとする．本論文では，semi-honest な攻撃者を想定し，事前に定めたプロトコルに従って処理を行うが，送受信される公開情報より，ユーザの秘密情報を含む，それ以上の情報を得ようとする．

2.3 秘密分散

(k,n) しきい値秘密分散法とはユーザが持っている秘密

情報を n 個の異なる値（以降，分散値）に変換し，分散する手法である．次の 2 つの条件を満たす秘密分散法を (k, n) しきい値秘密分散法という．ただし，$n \geq k > 1$.

- $k-1$ 個以下の分散値からは, 秘密情報に関する情報は一切得ることはできない.
- 任意の k 個以上の分散値から, 秘密情報を復元することができる.

代表的な (k, n) しきい値秘密分散法として Shamir の (k, n) 法や，加法的秘密分散法などがある．ただし，秘密分散法の問題点として 1 つの秘密情報が n 個の分散値になるためメモリ容量が増大し，符号化効率が良くないという問題点がある．そのため，秘密情報を部分的に漏らす集合を許し，安全性を緩めるかわりに符号化効率を高くするランプ型秘密分散法や一部の分散値を鍵サーバに任せる非対称秘密分散が提案されている．例えば，(k, L, n) ランプ型秘密分散では，L 個の秘密情報 $\mathcal{S} = \{s_0, \ldots, s_{L-1}\}$ に対して，ディーラは以下の分散式 $f(x)$ を生成し，n 個の分散値 $f(\alpha_0), \ldots, f(\alpha_{n-1})$ を計算する．

$$f(x) = s_0 + s_1 x + \ldots + s_{L-1} x^{L-1} + a_L x^L + \ldots + a_{k-1} x^{k-1} \quad (1)$$

n 個の分散情報のうち，任意の k 個以上集めれば秘密情報を完全に復号できるが，任意の $k-\ell$ $(1 \leq \ell \leq L)$ 個からは秘密情報 $\mathcal{S}$ に関して $(\ell/L)H(\mathcal{S})$ の曖昧さが残り，$k-L$ 個以下では秘密情報 $\mathcal{S}$ が全く漏洩しない [12]．対して，非対称秘密分散では，n 台のサーバから t 台 $(t < k)$ を選択し，鍵サーバとし，残りの $n-t$ サーバをデータサーバとする．鍵サーバは分散値を保存せず，擬似乱数を生成するための鍵のみを持ち，データサーバは分散値を保管する（詳細は [13] 参照）.

3. 関連研究：川原らの方式

川原らは，被災者と防災センターの間で信頼性が高く安全な情報交換手段を提供するために，地上の被害が影響しづらいドローンを用いた無線中継ネットワークに着目し，Shamir の $(2, n)$ 法を用いて，計算コストが低く，かつアクセス制御可能な手法を提案した [8]．ユーザ P_i から送られる秘密情報 m を以下のように定義する．ここで，id_i, loc_i, ts_i, $stat_i$ をそれぞれ P_i の通信端末の識別子 (128bit)，位置情報 (48bit)，メッセージ作成時刻 (32bit) と状態 (48bit) を表している．

$$m_i = (m_{i,1} || m_{i,2}) := (id_i || loc_i, ts_i, stat_i)$$

川原らの方式は，(1) 事前処理，(2) 情報共有処理，と (3) 復号処理から構成される．事前処理では，共有鍵やユーザ識別子などの情報が事前に共有される．また，生成される秘密 $m_{i,2}$ に対して，P_i はランダムな暗号鍵 key_i を生成し，$\mathrm{Enc}(m_i) = m_i \oplus key_i$ を計算し，生成された暗号鍵 key_i を，Shamir の $(2, n)$ 法を利用し，分散式 $f_i(x) = key_i + ax$ により，必要な分散値を計算し，ドローンネットワークに送信する．詳細な処理は [8] を参照してください.

しかし，[8] で提案された方法では，固定の共通鍵による情報漏洩などの課題があったため，それらの課題を解決・緩和するために，いくつかの改良法を提案された（例えば，分散値をランダム化できる手法に巡回冗長検査（CRC）を導入する手法 [9] や $(3, 2, 4)$ ランプ型秘密分散を用いた効率的な情報伝送を実現できる手法 [10] など）．しかし，いずれの手法でも，1 つの秘密情報に対する暗号文（送信する情報）のサイズを平文とほぼ同程度なサイズにすることができなかった．そこで，本研究では，より低通信量で実現できる情報共有システムの提案を目指す.

4. 提案法１：2 つの情報を同時に送信

非対称秘密分散法にけるローカルで計算できる情報を用いることで送信すべき情報量を削減し，弱い $(4, 2, 4)$ ランプ型秘密分散法における係数の 2 つを P_i の秘密情報と更新鍵にすることで同時に送信できる情報の「容量」を増やす手法を提案する．提案法 1 では，2 つの秘密情報（安否情報 m_i と更新用鍵 r_i）に対して，2 つの暗号文（分散値と補助乱数）を送信する必要があるが，5 章では，送信する情報をさらに削減できる手法を紹介する．情報共有処理では，次の分散式 $f_{i,t}(x)$ を利用する.

$$f_{i,t}(x) = m_{i,t} + (\gamma_{i,t} r_{i,t})x + a_{i,t,2} x^2 + a_{i,t,3} x^3 \quad (2)$$

式 (2) では，P_i の安否情報 $m_{i,t}$ を定数項に割り当て，ランダムで選択される乱数 $r_{i,t}$（更新鍵として使用可能）を第 2 の係数に設定する．なお，第 2 の係数 r_t を P_i の秘密情報 $m'_{i,t}$ に置き換えることも考えられるが，ここではその詳細を省略する．また，添え字 i は，P_i が実行するものを表し，t は t 回目の実行を表す．しかし，以降では簡単のために，添え字 i を省略する.

また，提案法 1 では，鍵事前配布を想定し，信頼できる $\mathcal{C}$ が，すべてのユーザ P_i に対して，鍵のペアや識別子 $id[i]$ などを事前に配布するとする．なお, 共有される鍵は, PRNG 関数，またはハッシュ関数の入力として機能する．識別子 $id[i]$ のサイズは，衝突を回避できる十分に大きな値にする（例えば，Bitcoin のアドレスでは，衝突を回避するために，アドレスを 160bit に設定）．さらに，提案法 1 では，災害センターが受け取った情報を識別するために，P_i は t 回目の実行時にアドレス $addr_{i,t}$ を生成する．ここで，ver, ts_t, chk_t はそれぞれ提案手法のバージョン，t 回目の実行タイムスタンプとチェックサムを表している.

$$addrs_t = ver || t || id[i] || ts_t || chk_t$$

提案方式の詳細は以下に示す．なお，すべての計算は Z_p

内で実行されが，添字に関する計算は Z_4 内で実行されるとする．プロトコル 4.0 は事前に行うとする（例えば，定めたパラメータを事前に小型通信端末の TEE 領域に埋め込むなど）．さらに，手順 (1) から (3) の計算は一度だけ行えばよい．情報共有処理では，通常の (k, n) しきい値秘密分散の手順と異なり，ユーザ P_i は，分散式 $f_t(x)$ に対する分散値 $f_t(\alpha_{j-1})$, $f_t(\alpha_j)$ と $f_t(\alpha_{j+2})$ を先に生成し，定めた分散値を用いて，残りの係数 $a_{t,2}$, $a_{t,3}$ を決定し，分散式を決定する．最後に，定めた分散式を用いて，残りの分散値 $f_t(\alpha_{j+2})$ を計算し，ドローンネットワークに送信する．

プロトコル 4.0: 事前処理

(1) 以下の (4×4)Vandermonde 行列を $\mathbf{X}$ とする．$\mathcal{C}$ は，$\mathbf{X}$ の逆行列に対して，要素 $(1, j)$ が 0 ($\mathbf{X}^{-1}_{(1,j)} = 0 \mod p$) となるパラメータ α_0，α_1，α_2 と α_3 を選択する．ここで，$0 \leq j \leq 3$ とする．

$$\mathbf{X} = \begin{bmatrix} 1 & \alpha_0 & \alpha_0^2 & \alpha_0^3 \\ 1 & \alpha_1 & \alpha_1^2 & \alpha_1^3 \\ 1 & \alpha_2 & \alpha_2^2 & \alpha_2^3 \\ 1 & \alpha_3 & \alpha_3^2 & \alpha_3^3 \end{bmatrix} \tag{3}$$

(2) 以降では，簡単のために，$j = 1$ にし，$X^{-1}_{(1,1)} = 0$ が成り立つとする．よって，$\alpha_j = \alpha_1$ になり，残りのパラメータを $\alpha_{j-1} = \alpha_0$, $\alpha_{j+1} = \alpha_2$, $\alpha_{j+2} = \alpha_3$ とする．

$$\mathbf{X}^{-1} = \begin{bmatrix} X^{-1}_{(0,0)} & X^{-1}_{(0,1)} & X^{-1}_{(0,2)} & X^{-1}_{(0,3)} \\ X^{-1}_{(1,0)} & 0 & X^{-1}_{(1,2)} & X^{-1}_{(1,3)} \\ X^{-1}_{(2,0)} & X^{-1}_{(2,1)} & X^{-1}_{(2,2)} & X^{-1}_{(2,3)} \\ X^{-1}_{(3,0)} & X^{-1}_{(3,1)} & X^{-1}_{(3,2)} & X^{-1}_{(3,3)} \end{bmatrix}$$

(3) $\mathcal{C}$ は以下を計算する．

$$\mathbf{A}^{-1} = \begin{bmatrix} \alpha_0 & \alpha_0^2 & \alpha_0^3 \\ \alpha_1 & \alpha_1^2 & \alpha_1^3 \\ \alpha_2 & \alpha_2^2 & \alpha_2^3 \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1}_{(0,0)} & A^{-1}_{(0,1)} & A^{-1}_{(0,2)} \\ A^{-1}_{(1,0)} & A^{-1}_{(1,1)} & A^{-1}_{(1,2)} \\ A^{-1}_{(2,0)} & A^{-1}_{(2,1)} & A^{-1}_{(2,2)} \end{bmatrix}$$

(4) $\mathcal{C}$ は，各 P_i に対して次の鍵のペアと識別子 $id[i]$ を生成する．

$$\left(key_i, key_i^{\mathcal{C}}\right)$$

(5) $\mathcal{C}$ は以下の情報を各 P_i と共有する．

$$\left(\alpha_0, \alpha_1(=\alpha_j), \alpha_2, \alpha_3, \alpha_3^2, \alpha_3^3, key_i, key_i^{\mathcal{C}}, id[i], \mathbf{A}^{-1}\right)$$

以降は，上記で指定されたパラメータを利用するが，提案法 1 の有効性に関しては，これらのパラメータに限定されず，任意のパラメータにも適用できる（プロトコル 4.0 の手順 (1) の条件を満たす必要がある）．

プロトコル 4.1: 情報共有処理

(1) P_i は秘密情報 m_t を生成し，以下の 2 つ分散値を計算する．ここで，ts_t は秘密情報 m_t が生成される時刻を表す．なお，手順 (5) のチェックサム計算と区別するために，以下の分散値を計算する際に PRNG 関数を使用するとしたが，ハッシュ関数を使用しても良い．

$$f_t(\alpha_0) = PRNG(key_i, ts_t)$$
$$f_t(\alpha_2) = PRNG(key_i^{\mathcal{C}}, ts_t)$$

(2) P_i は乱数 $r_t \in Z_p^*$ を生成し，以下の式を用いて，分散値 $f(\alpha_1)$ と補助乱数 γ_t を計算する．

$$f_t(\alpha_1) = PRNG(key_i^{\mathcal{C}}, r_t)$$

$$\gamma_t = (A^{-1}_{(0,0)}(f_t(\alpha_0) - m_t) + A^{-1}_{(0,1)}(f_t(\alpha_1) - m_t) + A^{-1}_{(0,2)}(f_t(\alpha_2) - m_t))/r_t \tag{4}$$

(3) P_i は以下の関係を満たす分散式 $f_t(x)$ における残りの係数 $a_{t,2}, a_{t,3}$ を計算する．

$$f_t(\alpha_0) = m_t + (\gamma_t r_t)\alpha_0 + a_{t,2}\alpha_0^2 + a_{t,3}\alpha_0^3$$
$$f_t(\alpha_1) = m_t + (\gamma_t r_t)\alpha_1 + a_{t,2}\alpha_1^2 + a_{t,3}\alpha_1^3$$
$$f_t(\alpha_2) = m_t + (\gamma_t r_t)\alpha_2 + a_{t,2}\alpha_2^2 + a_{t,3}\alpha_2^3$$

(4) P_i は手順 (2)(3) で計算した情報を用いて，以下の分散式 $f_t(x)$ を生成し，残りの分散値 $f(\alpha_3)$ を計算する．

$$f_t(x) = m_t + (\gamma_t r_t)x + a_{t,2}x^2 + a_{t,3}x^3 \tag{5}$$

(5)（オプション）P_i は，識別子 $id[i]$，$f_t(\alpha_3)$ と γ_t を連結した結果に対して，ハッシュ関数（例えば，SHA-256）を 2 回実行し，その出力の最初の 32 ビットをチェックサム $chk_t^{'}$ として使用する．

$$h_{t,1} = h(id[i]||f_t(\alpha_3)||\gamma_t)$$
$$h_{t,2} = h(h_{t,1})$$
$$h_t^{'} = MSB_{32}(h_{t,2})$$

(6) P_i は，生成した情報を用いて，アドレス $addr_t$ を生成し，以下の情報をネットワークに送信する．

$$(addr_{i,t}, \gamma_t, f_t(\alpha_3))$$

プロトコル 4.2: 復元処理

(1) $\mathcal{C}$ はドローンネットワークから以下の情報を収集し，チェックサムの検証を行う．

$$(addr_{i,t}, \gamma_t, f_t(\alpha_3))$$

(2) $\mathcal{C}$ は $id[i]$ に一致する鍵のペア $(key_i, key_i^{\mathcal{C}})$ を検索し，以下の分散値を計算する．

$$f_t(\alpha_0) = PRNG(key_i, ts_t)$$
$$f_t(\alpha_2) = PRNG(key_i^{\mathcal{C}}, ts_t)$$

(3) $\mathcal{C}$ は以下の式より $r_t^{'}$ を復元する．

$$r_t^{'} = (X_{(1,0)}^{-1} f_t(\alpha_0) + X_{(1,2)}^{-1} f_t(\alpha_2) + X_{(1,3)}^{-1} f_t(\alpha_3))/\gamma_t \quad (6)$$

(4) $\mathcal{C}$ は手順 (3) で求めた乱数 $r_t^{'}$ を用いて，分散値 $f_t(\alpha_1)$ を計算する．

$$f_t(\alpha_1)^{'} = PRNG(key_i^{\mathcal{C}}, r_t^{'})$$

(5) $\mathcal{C}$ は P_i の秘密情報 $m_t^{'}$ を以下のように復元する．

$$m_t^{'} = X_{(0,0)}^{-1} f_t(\alpha_0) + X_{(0,1)}^{-1} f_t(\alpha_1) + X_{(0,2)}^{-1} f_t(\alpha_2) + X_{(0,3)}^{-1} f_t(\alpha_3) \quad (7)$$

5. 拡張法１：通信量の削減

5 章では，4 章で紹介した提案法 1 における γ_t を送信する手間を排除する方法を紹介し，送信する情報を 2 つのままで，ネットワークに伝達する分散値を 1 つにできる拡張法を紹介する．分散式 (5) における γ_t の送信に関わるコストを削減するためには，事前に共有された鍵を用いて，ローカルで計算できるようにする．しかし，これによって，式 (5) の r_t が $Z_p^* = Z_p \setminus \{0\}$ からランダムに選択されるではなく，ユーザ P_i が生成した秘密情報 m_t から導出されることになる．しかし，拡張法１は提案法１の機密性要件を維持し，計算された r_t が攻撃者に漏洩しないため，提案法１同様に，更新鍵として利用することも可能である．以下は，変更部分のみ記載する．

プロトコル 5.1: 情報共有処理

(1) P_i はプロトコル 4.1 の手順 (1) を実行し，秘密情報 $m_t \in Z_p$ を生成し，以下の式より r_t を計算する．ここで，$\gamma_t = PRNG(key_i, key_i^{\mathcal{C}}, ts_t)$ とし，$\gamma_t - A_{0,1}^{-1} \neq 0$ とする（$\gamma_t - A_{0,1}^{-1} \neq 0$ の場合，時刻 ts_t を変更し γ_t を再計算する）．

$$r_t = A_{(0,0)}^{-1}(f_t(\alpha_0) - m_t) + A_{(0,1)}^{-1}(-m_t) + A_{(0,2)}^{-1}(f_t(\alpha_2) - m_t)/\gamma_t - A_{(0,1)}^{-1} \quad (8)$$

(2) P_i は手順 (3) から (5) を実行し，以下をドローンネットワークに送信する．

$$(addr_{i,t}, f_t(\alpha_3))$$

プロトコル 5.2: 復元処理

(1) $\mathcal{C}$ はドローンネットワークより以下の情報を収集し，チェックサムの検証を行う．

$$(addr_{i,t}, f_t(\alpha_3))$$

(2) $\mathcal{C}$ は $id[i]$ に一致する鍵のペア $(key_i, key_i^{\mathcal{C}})$ を検索し，以下を計算する．

$$f_t(\alpha_0) = PRNG(key_i, ts_t)$$
$$f_t(\alpha_2) = PRNG(key_i^{\mathcal{C}}, ts_t)$$
$$\gamma_t = PRNG(key_i, key_i^{\mathcal{C}}, ts_t)$$

(3) $\mathcal{C}$ は以下の式より分散値 $f(\alpha_1)^{'} = r_t^{'}$ を計算する．

$$f_t(\alpha_1)^{'} = r_t^{'} = (X_{(1,0)}^{-1} f_t(\alpha_0) + X_{(1,2)}^{-1} f_t(\alpha_2) + X_{(1,3)}^{-1} f_t(\alpha_3))/\gamma_t \quad (9)$$

(4) $\mathcal{C}$ は P_i の秘密情報 $m_t^{'}$ を以下のように復元する．

$$m_t^{'} = X_{(0,0)}^{-1} f_t(\alpha_0) + X_{(0,1)}^{-1} f_t(\alpha_1) + X_{(0,2)}^{-1} f_t(\alpha_2) + X_{(0,3)}^{-1} f_t(\alpha_3) \quad (10)$$

6. 拡張法２：通信量と計算量の削減

4 章と 5 章で紹介した手法をさらに効率化し，通信量と計算量を削減できる方法を検討する．提案法１と拡張法１における 1 回の情報共有処理の実行において 2 つの情報を送信するに対して，拡張法２では，同じ $(4, 2, 4)$ ランプ型秘密分散法を用いて，1 つの秘密情報のみを安全かつ効率的に共有する方法を検討する．拡張法２は，以下の分散式を使用する．

$$f_t(x) = r_t + m_t x + a_{t,1} x^2 + a_{t,2} x^3 \quad (11)$$

r_t は真正乱数とし，定数項として設定する．また，安否情報 m_t を分散式の第 2 項の係数として設定することによって，情報共有処理の計算量を大幅に削減する．なお，ここの乱数 r_t は，分散式をランダム化にする役割を持ち，復元する必要はない情報とする．そのため，γ_t などのような補助情報を送信する必要はない．以下に，提案する拡張法２の詳細を説明する．

プロトコル 6.0: 事前処理

(1) $\mathcal{C}$ はプロトコル 4.0 の手順 (1), (2) を実行し，$\alpha_j = \alpha_1$, $\alpha_{j-1} = \alpha_0$, $\alpha_{j+1} = \alpha_2$, $\alpha_{j+2} = \alpha_3$ とする．

(2) $\mathcal{C}$ は以下を計算する．

$$\mathbf{B}^{-1} = \begin{bmatrix} \alpha_0^2 & \alpha_0^3 \\ \alpha_2^2 & \alpha_2^3 \end{bmatrix}^{-1} = \begin{bmatrix} B_{(0,0)}^{-1} & B_{(0,1)}^{-1} \\ B_{(1,0)}^{-1} & B_{(1,1)}^{-1} \end{bmatrix}^{-1}$$

(3) $\mathcal{C}$ は，各 P_i に対して次の鍵のペア $(key_i, key_i^{\mathcal{C}})$ と識別子 $id[i]$ を生成し，以下の情報を各 P_i と共有する．

$$\left(\alpha_0, \alpha_1(=\alpha_j), \alpha_2, \alpha_3, \alpha_3^2, \alpha_3^3, key_i, key_i^{\mathcal{C}}, id[i], \mathbf{B}^{-1}\right)$$

プロトコル 6.1: 情報共有処理

(1) P_i は以下の式より分散値を生成する．

$$f_t(\alpha_0) = PRNG(key_i, ts_t)$$
$$f_t(\alpha_2) = PRNG(key_i^{\mathcal{C}}, ts_t)$$

(2) P_i は秘密情報 m_t を生成し，乱数 $r_t \in Z_p$ を選択し，以下の式より分散式 $f_t(x)$ における係数 $a_{t,2}, a_{t,3}$ を計算する．

$$\begin{bmatrix} a_{t,2} \\ a_{t,3} \end{bmatrix} = \mathbf{B}^{-1} \cdot \begin{bmatrix} f_t(\alpha_0) - r_t - m_t\alpha_0 \\ f_t(\alpha_2) - r_t - m_t\alpha_2 \end{bmatrix}$$

(3) P_i は手順 (2) で計算した $a_{t,2}a_{t,3}$ を用いて，分散式 $f_t(x)$ を生成し，分散値 $f_t(\alpha_3)$ を計算する．

$$f_t(x) = r_t + m_t x + a_{t,2}x^2 + a_{t,3}x^3$$

(4) P_i はプロトコル 4.1 の手順 (5) を実行し，以下をドローンに送信する．

$$(addr_{i,t}, f_t(\alpha_3))$$

プロトコル 6.2: 復元処理

(1) $\mathcal{C}$ はドローンネットワークより以下の情報を収集し，チェックサムの検証を行う．

$$(addr_{i,t}, f_t(\alpha_3))$$

(2) $\mathcal{C}$ はプロトコル 4.2 の手順 (2), (3) を実行し，以下より秘密情報 m_t を復元する．

$$m_t = X_{(1,0)}^{-1} f_t(\alpha_0) + X_{(1,2)}^{-1} f_t(\alpha_2) + X_{(1,3)}^{-1} f_t(\alpha_3)$$

7. 考察及び評価

7.1 提案法の安全性について

提案した 3 つの手法の安全性を考察する．しかし，ページ数の制約上，定理やシミュレーションによる詳細な安全性評価は省略する．提案法では，ユーザ P_i と災害センター $\mathcal{C}$ が両方とも信頼できるとする．事前処理における鍵のペア $(key_i, key_i^{\mathcal{C}})$ が安全な方法で事前に共有され，使用する乱数が等確率 $1/Z_p^*$ で選択されるとする．また，簡単のために，秘密情報 $\mathcal{S} = \{m_t, r_t\}$ に対して，$(4, 2, 4)$ ランプ型秘密分散において生成される分散値集合を $\mathcal{F} = \{f_t(\alpha_0), f_t(\alpha_1), f_t(\alpha_2), f_t(\alpha_3)\}$ とし，そのプライバシー集合を $\mathcal{B}$ とする．ここで，$\mathcal{B} \subset \mathcal{F}$, $0 \leq |B| \leq 2$ (for $k - L = 2$) と仮定する．$\mathcal{B} \neq \emptyset$ ならば，$\mathcal{B}$ は秘密 $\mathcal{S}$ のプライバシー集合となり，$H(\mathcal{S}|\mathcal{B}) = H(\mathcal{S})$ となる．つまり，$\mathcal{B}$ に含まれる分散値は，秘密情報 $\mathcal{S}$ とは独立であり，攻撃者は通信路よりの情報 $\mathcal{B}$ を集めたとしても，秘密情報 $\mathcal{S}$ に関する情報を一切得ることができない．

攻撃者は，提案法 1 , 拡張法 1 と拡張法 2 の情報共有処理において，以下の情報を知ることができる．$\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3$ はそれぞれ提案法 1 , 拡張法 1 , 拡張法 2 を実行した際に得た公開情報の集合とする．

$$\mathbf{X}_1 = \{addr_{i,t}, \gamma_t, f_t(\alpha_3)\}$$
$$\mathbf{X}_2 = \{addr_{i,t}, f_t(\alpha_3)\}$$
$$\mathbf{X}_3 = \{addr_{i,t}, f_t(\alpha_3)\}$$

まず，提案法 1 の γ_t について考える．γ_t は提案法 1 で生成した r_t を保護するための補助乱数であり，秘密 r_t を得るには，$\gamma_t r_t$ を先に求める必要がある．式 (6) より $\gamma_t r_t$ を求めるには，分散値 $f_t(\alpha_0)$ と $f_t(\alpha_2)$ を得る必要がある．しかし，鍵のペア $(key_i, key_i^{\mathcal{C}})$ は漏洩しない限り，攻撃者は分散値 $f_t(\alpha_0)$ と $f_t(\alpha_2)$ を知ることができない．よって，$\gamma_t r_t$ も漏洩しないと言える．さらに，提案法と拡張法では，$(4, 2, 4)$ ランプ型秘密分散法に基づく分散式 $f_t(x)$ を用いて秘密情報を分散する．ランプ型の安全性定義によると，攻撃者は任意の $k - L$ 個以下の分散値を集めても，秘密情報 $\mathcal{S}$ が全く漏れない．すなわち，提案した 3 つの手法より攻撃者が収集できる上記の公開情報からは，1 つの分散値 $f(\alpha_3) \in \mathcal{B}$ は漏洩する．しかし，$(4, 2, 4)$ ランプ型秘密分散のプライバシー定義より，攻撃者はこの分散値から秘密情報 $\mathcal{S}$ を一切得ることができず，秘密情報の秘密性が保証され，以下が言える．

$$H(\mathcal{S}|\mathbf{X}_1) = H(\mathcal{S}),\ H(\mathcal{S}|\mathbf{X}_2) = H(\mathcal{S}),\ H(\mathcal{S}|\mathbf{X}_3) = H(\mathcal{S})$$

7.2 関連研究との定性的な評価

ここでは，提案法 1 ，拡張法 1 と拡張法 2 を，従来の秘密分散を用いた情報共有法 [8], [9]，秘密計算を利用した安全な情報通信法 [6] と比較を行う．各方式の特徴，通信量，ラウンド数，計算量を比較したものを表 1 にまとめる．なお，比較に使用する記号を以下に定義する．

記号定義:

- $|m|$: 1 つの秘密情報・シェアのサイズ
- $\mathcal{L}$ ：$k = 2$ におけるラグランジュ補間法の計算量
- $\mathcal{H}$ ：PRNG・ハッシュ関数の計算量
- $\mathcal{C}$ ：巡回冗長検査 (CRC-32) の計算量
- $\mathcal{B}$ ：行列置換・ビットシフトの計算量
- $\mathcal{A}$ ：加算・減算の計算量
- $\mathcal{M}$: 乗算の計算量
- $\mathcal{D}$ ：除算（逆元を求めて，乗算）の計算量

秘密分散によって生成される各分散値はのサイズは通常，秘密情報のサイズとほぼ同程度であるため，秘密情報と分散値のサイズを $|m|$ とする．また，加減算と XOR 演算は，乗除算よりも計算量が軽いため，表 1 における各方式の計算量には，乗除算の計算量が含まれる場合，加算と XOR 演算の計算量を省略する．また，[8], [9] で提案された情報共有法は，$\mathcal{C}$ に加えて，P_i が許可する一人のユーザ $P_{i,1}$ に情報を共有する処理も議論されているが，公平のために，表 1 では，**ユーザ P_i が $\mathcal{C}$ に，1 回の秘密情報を送信**

する場合を想定し，他のユーザとの情報共有を考慮しない．

7.3 提案方式の実装評価

有効性を評価するために，提案法１，拡張法 1 と拡張法２を実装し，P_i による情報共有処理の実行時間と $\mathcal{C}$ による復元処理の実行時間を記録した．また，異なる計算能力を持つ計算機による実行時間の変化を評価するために，3 つの異なる実行環境を用いて，C++言語を使用して実装を行った（表 2 参照）．なお，実装では，複数スレッドを利用した並列計算などの最適化処理は考慮せず，実行時間は，C++11 の std::chrono ライブラリを使用し測定した．また，簡単のため，[8] の情報共有処理における手順 (2) と (4) の 2 進数の XOR 演算を $GF(p)$ 上の加算演算で実装した．2 進数の XOR 演算よりも実行時間が少し増えると予想されるが，ブール回路における XOR ゲートによって保証されている安全性は変わらない．また，提案法１とその拡張法におけるタイムスタンプ情報は，gen_rand 関数を利用し，ランダムに生成されるとする．さらに，異なるサイズの素数 p による影響を調べるために，256bit と 128bit で表現する異なる素数 p を選択し，かつ，ログ機能の有無による実行時間の変化も測定した．各手法を $100,000$ 回実行し，その平均実行時間を表 3 と表 4 にまとめた．

提案法１と拡張法１は，複数回のハッシュ関数と乗算が必要のため，表 3 より，情報共有処理を実行するのにかかる時間が，川原ら方式と岩村ら方式よりも長いことが分かった．特に，提案法１と拡張法１は，川原ら方式に比べて 3 倍，岩村ら方式に比べて 2 倍の実行時間が必要になった．その理由の一つは，提案法で使用される分散式 $f(x)$ の次数 $\deg(f) = 3$ であるため，分散値を計算する際に多くの時間がかかると考えられる．しかし，提案法１と拡張法１は，1 回の通信で 2 つの情報を送信できるに対して，川原らと岩村ら方式では 1 つの情報のみ送信できる．

復元処理の時間を比較すると，使用する分散式の次数 $\deg(f) = 3$ であっても，提案法１と拡張法１は川原らと岩村ら方式とほぼ同じであることが分かった．また，拡張法２では，1 つの情報を復元するため，どの手法よりも高速に実現できることが分かった．提案法では，分散値の識別子 α_0，α_1，α_2 と α_3 の値は事前の決められるため，復元に必要となるラグランジュ係数も事前に計算でき，復元処理の効率化を実現する．対して，川原ら方式では，復元処理ごとに以下のラグランジュ補間の演算が必要になるため，余計な計算時間が必要となる（各記号の意味は [8] を参照）．

$$f_t(0) = f_t(r_2)\frac{-d_C}{r_2 - d_C} + K_{C,i}^{(2)}\frac{-r_2}{d_C - r_2}$$

最後に，提案法１，拡張法１と拡張法２の実行時間は，1ms 未満に抑えられていて，かつ，送信する情報量を他の秘密分散を利用した従来方式よりも低いため，本研究で想定する小型通信端末，または，IoT デバイスのバッテリ長寿命化と通信のオーバーヘッドを最小限にすることができ，様々な IoT アプリケーションに適すると言える．

8. おわりに

本研究では，非対称と $(4, 2, 4)$ ランプ型秘密分散の部分的な情報漏洩の特性を用いて，秘密分散の問題であった送信する分散値の数を最小にしながら，低計算量で実現できる情報共有法を 3 つ提案した．今後は，提案法における乗算回数（計算量）をさらに削減し，分散値の改ざんができる malicious な攻撃に対する安全性を検討する予定である．

謝辞　本研究は JSPS 科研費 JP23K16884 の助成を受けたものである．

参考文献

[1] L. Gupta, R. Jain, G. Vaszkun, “Survey of important issues in UAV communication networks,” IEEE Communications Surveys & Tutorials, vol. 18, no. 2, pp. 1123–1152, 2016.

[2] E. W. Frew, T. X. Brown, “Airborne communication networks for small unmanned aircraft systems,” in Proceedings of the IEEE, vol. 96, no. 12, pp. 2008–2027, 2008.

[3] R. Altawy, A. M. Youssef, “Security, Privacy, and Safety Aspects of Civilian Drones: A Survey,” ACM Transactions on Cyber-Physical Systems, vol. 1, no. 2, pp. 1–25, 2016.

[4] C. Liu, T. Quek, J. Lee, “Secure UAV communication in the presence of active eavesdropper (invited paper),” 2017 WCSP, pp. 1–6, 2017.

[5] A. Abdallah, M. Ali, J. Mišić, V. Mišić, “Efficient security scheme for disaster surveillance UAV communication networks.” Information 2019, vol. 10, no. 43, 2019.

[6] K. Iwamura, A. Kamal, “Secure user authentication with information theoretic security using secret sharing-based secure computation,” IEEE Access, vol. 13, pp. 9015–9031, 2025.

[7] A. Shamir, “How to share a secret.” Communications of the ACM, vol. 22, no. 11, pp. 612–613, 1979.

[8] Y. Kawahara, A. Kamal, M. Fujisawa, “Lightweight information-sharing system with access control in disaster management with security against dishonest users,” Journal of Signal Processing, vol. 28, no. 4, pp. 161–164, 2024.

[9] Y. Kawahara, M. Fujisawa, A. Kamal, “Secure information Sharing with access control and tamper detection for drone-based disaster management,” in Proceedings of 2024 IEEE 13th Global Conference on Consumer Electronics (GCCE), pp. 178–180, 2024.

[10] 川原 悠暉，藤沢 匡哉，アフマド アクマル アミヌディン，“Ramp 型秘密分散を用いて軽量で改ざん検知可能な暗号方式”，電子情報通信学会ソサイエティ大会，2024.

[11] M. Yahuza, M. Idris, I. Ahmedy, A. Wahab, T. Nandy, N. Noor, A. Bala, “Internet of drones security and privacy issues: taxonomy and open challenges,” IEEE Access, vol. 9, pp. 57243–57270, 2021.

[12] H. Yamamoto, “Secret sharing system using (k, L, n) threshold scheme,” Electronics and Communications in Japan, vol. 69, no. 9, pp. 46–54, 1986.

[13] K. Iwamura, A. Kamal, “Simple Approach to Realizing Verifiable Secret Sharing for Secure Cloud System,’’ in IEEE Access, vol. 10, pp. 76794–76804, 2022.

表 1 従来の秘密分散を用いた手法との比較

	提案法 1	拡張法 1	拡張法 2	[8]	[9]	[6]
k, n の設定	$k=4, n=4$	$k=4, n=4$	$k=4, n=4$	$k=2, n\geq 2$	$k=2, n\geq 2$	$k=2, n=2$
共有する情報数, L	$\mathcal{S}=\{m_t, r_t\}$	$\mathcal{S}=\{m_t, r_t\}$	$\mathcal{S}=\{m_t\}$	$\mathcal{S}=\{m_t\}$	$\mathcal{S}=\{m_t\}$	$\mathcal{S}=\{m_t\}$
分散値のランダム性	✓	✓	✓	✗	✓	✓
複数ユーザと共有機能	✗	✗	✗	✓	✓	✗
鍵更新機能 （複数秘密 m に拡張可）	✓	✓	✗	✗	✗	✗
チェックサム機能	✓	✓	✓	✗	✓	✗
事前処理	✓	✓	✓	✓	✓	✓
通信量（情報共有） 通信量（復元）	$\lvert addr_t\rvert + 2\lvert m\rvert$	$\lvert addr_t\rvert + \lvert m\rvert$	$\lvert addr_t\rvert + \lvert m\rvert$	$4\lvert m\rvert$	$\lvert id[i]\rvert + 3\lvert m\rvert$	$3\lvert m\rvert$
ラウンド数（情報共有） ラウンド数（復元）	1	1	1	1	1	1
計算量（情報共有）	$13\mathcal{M}+\mathcal{D}+5\mathcal{H}$	$13\mathcal{M}+\mathcal{D}+5\mathcal{H}$	$11\mathcal{M}+4\mathcal{H}$	$\mathcal{M}+\mathcal{D}$	$\mathcal{M}+\mathcal{B}+\mathcal{C}$	$4\mathcal{M}+2\mathcal{D}$
計算量（復元）	$7\mathcal{M}+\mathcal{D}+5\mathcal{H}$	$7\mathcal{M}+\mathcal{D}+4\mathcal{H}$	$3\mathcal{M}+4\mathcal{H}$	$\mathcal{L}$	$\mathcal{B}+\mathcal{C}+\mathcal{L}$	$5\mathcal{M}+2\mathcal{D}$

表 2 実行環境

	(1) Dell PC	(2) Raspberry Pi 5	(3) Raspberry Pi 4
ホスト	PowerEdge T150	Pi 5 Model B Rev 1.1	Pi 4 Model B Rev 1.5
CPU	Intel Xeon E-2378G @ 2.80GHz	Arm Cortex-A76 @ 2.40GHz	Arm Cortex-A72 @ 1.80GHz
メモリ	15.3 GB	15.6 GB	7.8 GB
OS	Ubuntu 20.04.6 LTS	Ubuntu 24.04.2 LTS	Ubuntu 24.04.2 LTS
カーネル	5.15.0-139-generic	6.8.0-1030-raspi	6.8.0-1030-raspi
Openssl ver.	1.1.1f 31 Mar 2020	3.0.13 30 Jan 2024	3.0.13 30 Jan 2024
g++ ver.	9.4.0	13.3.0	13.3.0

表 3 平均実行時間（() 内の値は 128 ビット素数 p を使用した場合の時間） 単位：μs

処理	環境	提案法 1	拡張法 1	拡張法 2	川原ら方式 [8]	岩村ら方式 [6]
				(without logs)		
情報共有	(1)	67 (53)	67 (52)	43 (37)	22 (16)	32 (21)
	(2)	157 (126)	158 (124)	104 (88)	51 (36)	76 (50)
	(3)	360 (296)	356 (294)	236 (206)	124 (94)	166 (115)
復元	(1)	37 (27)	30 (21)	11 (9)	36 (21)	26 (16)
	(2)	87 (65)	71 (50)	27 (23)	83 (51)	77 (46)
	(3)	191 (147)	153 (112)	61 (51)	171 (108)	152 (92)

表 4 平均実行時間（() 内の値は 128 ビット素数 p を使用した場合の時間） 単位：μs

処理	環境	提案法 1	拡張法 1	拡張法 2	川原ら方式 [8]	岩村ら方式 [6]
				(with logs)		
情報共有	(1)	756 (699)	669 (607)	665 (474)	590 (531)	368 (274)
	(2)	755 (816)	677 (719)	677 (530)	722 (541)	371 (279)
	(3)	973 (894)	869 (821)	759 (678)	752 (557)	438 (331)
復元	(1)	149 (148)	148 (115)	73 (92)	207 (164)	215 (162)
	(2)	209 (167)	173 (170)	80 (81)	231 (173)	221 (166)
	(3)	316 (256)	276 (215)	116 (97)	326 (251)	314 (230)