秘密分散技術を用いた非集中化ストレージサービスの提案

張 一凡† 尾形 徹† 小池 幸生†

概要：当研究ではインターネットストレージのように利用時のデータの機密性や可用性を特定企業に依存させず，コストも削減できるストレージの構成方法を提案する．その構成方法とは，利用者環境に存在する PC などの端末のストレージを秘密分散技術によって一つのストレージサービスとして構成するものである．秘密分散技術により機密性を向上し，既存の機器を利用することでコストを抑えたストレージを構築できるが，既存の機器を利用するためには機器に応じた容量や可用性の偏りを考慮した分散方法が必要である．当研究では可用性とストレージ利用効率を数式化する事によって，機器毎の特性を考慮し利用者が自身の求める任意のコストパフォマンスを満たすストレージサービスの構築方法を提案する．

A Study of Decentralized Storage Service with Secret Sharing Scheme

IIFAN TYOU TORU OGATA YUKIO KOIKE

1. はじめに

近年，生成される情報の増加に伴い，それらを利用者環境内に保管する大容量 HDD(Hard Disk Drive)の開発が進むと同時に，インターネットストレージなど外部環境に保管するストレージサービスも多数提供されている．安価に大容量の情報を安全に保管するため，利用者は各種ストレージ製品・サービスの比較選定を行う必要がある．a

ストレージとして HDD や NAS(Network Attached Storage)を利用し利用者環境内にデータを保管する場合，機器故障などによる可用性のリスクが生じる．この可用性のリスクを回避するために対策として冗長度を高める方法があるが，対策によりコストが増加する．

ストレージとしてインターネット上のストレージサービスを利用する場合，サービスでの対策により可用性の確保が容易になる．しかし，この場合もストレージサービスの管理者からデータを見られる機密性の問題，ネットワーク切断時にデータアクセスができなくなる可用性の問題が生じる．

本研究では，「機密性」，「可用性」，「低コスト化」を共に満たすサービスの検討を行い，そこでの課題と解決に向けた提案を行う．

2. 非集中化ストレージサービス

「機密性」，「可用性」，「低コスト化」を満たすサービスとして，

- 機密性：秘密分散により情報を断片化することで，断片を保管するストレージでの機密性を確保
- 可用性：冗長化を高め，単一故障点を持たないシステム構成により可用性を向上
- 低コスト化：既存利用者ストレージを再利用することによりコストを削減

が可能となる「非集中化ストレージサービス」を提案する．

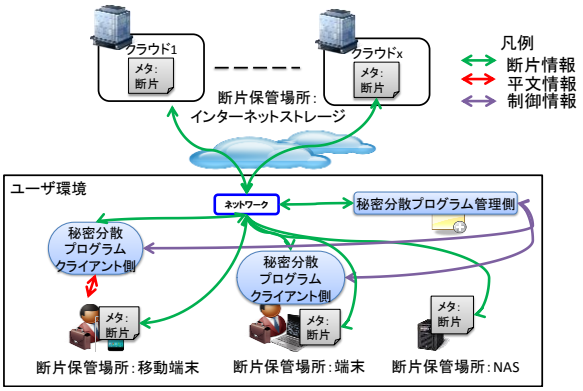

図 2-1 非集中化ストレージサービス

「非集中化ストレージサービス」は図 2-1 の後述する「断片保管場所」と「秘密分散プログラム管理側」および「秘密分散プログラムクライアント側」から成り立つ．利用者は「秘密分散プログラム」を経由して，NAS のような一つのストレージを操作する．当ストレージに配置されるファイルは秘密分散により断片化され，その断片をシステムに予め指定したインターネットストレージサービスやローカルストレージからなる「断片保管場所」で保管する．

当サービスの実現方法とシステム構成については 2.1 節で紹介し，類似サービスと比較した際のメリット整理をエ

† 西日本電信電話株式会社
技術革新部 研究開発センタ

ラー! 参照元が見つかりません。節で実施する．その後課題を 2.3 節で述べる．

2.1 システム構成の提案

提案する「非集中化ストレージサービス」を実現するためのシステム構成を下記のように検討した．

- 断片保管場所 : 利用者端末やインターネットストレージに断片を保管する場所を「断片保管場所」と呼ぶ．平文ファイル M を秘密分散処理により生成される「断片」と，M に対する全ての断片の保管箇所を示す「メタ情報」をセットにして保管する．このような構成により，単一故障点となりやすい，メタデータ管理サーバの利用せずにシステムを構築できる．複数の端末からの断片保管場所にアクセスするため，断片保管端末では CIFS や WebDAV などのインターフェイスによって断片を利用者に対して公開する．
- 秘密分散プログラム
  - クライアント側 : 利用者の端末で動作し，管理側から「断片保管場所」とアクセス権限の情報を取得し，それを元に断片とメタ情報にアクセスする．メタ情報から利用者に見せるフォルダ構造を構築し，利用者がアクセスできるフォルダや仮想ドライブ等の形式での操作を可能にする．利用者がファイル読み出しを行う際は各「断片保管場所」から断片を収集，復号し平文を利用者に提供し，ファイル書き込みを行う際は投入された平文を断片に分散し，メタ情報とともに各「断片保管場所」に保存する．操作時のファイルのロック情報はメタ情報を経由して複数端末間で共有する．
  - 管理側 : 利用者の端末とは別に NAS サーバなどの専用機器上で動作し，利用者の設定 UI，設定の管理インターフェイスおよびクライアント側プログラムと同等のファイルアクセスインターフェイスを提供する．

クライアント分散プログラムを実行できる利用者は自環境，できない利用者は管理側環境を経由して通常のファイル操作を行うことにより，秘密分散処理のメリットを享受したストレージサービスを利用できる．

2.2 提案構成による目標達成確認

提案するシステム構成では秘密分散を用いることでサービスとして目指した「機密性」,「可用性」,「低コスト化」を実現している．また，非集中化構成を取ることで，更なる「可用性」の向上と秘密分散を用いることにより発生する「性能」課題への対策を行っている．

2.2.1 機密性

可用性向上とコスト削減を可能にするストレージサービスとして，HDFS などの分散ストレージ構成がある．分散ストレージでも提案手法と同様に，データを複数の断片に分散し，別々のストレージに保管する機能を有する．しかし，多くの分散ストレージでは断片の機密性は保証されていない．本施策では秘密分散技術を用いることで断片の機密性を確保する．提案構成では各ストレージに秘匿化した断片しか保管しないため，断片保管時に発生する「機密性」に関する問題を解決できる．

2.2.2 可用性

秘密分散データの復号には分散ストレージで用いられているように「総断片数 n≧復号に必要な断片数 k」という特性(3.1 節参照)があり，すべての断片を集めずとも復号しファイルアクセスを提供できる．この特性を用いることで，故障・紛失・ネットワークから切断によって一部断片保管端末にアクセスできない場合でもファイル操作が可能であり，分散ストレージと同等の可用性を提供できる．

更に，提案構成では非集中化として秘密分散の処理をクライアント，専用機器の双方での実行可能にしており，処理端末依存の単一故障点の発生も回避できる．

このため，提案構成を用いることでインターネットストレージの切断を含む断片保管端末の切断や処理端末故障に対して「可用性」を確保することができる．

2.2.3 コスト

断片の機密性確保により，利用者環境で普及している高度なセキュリティ対策を有しない PC などの一般端末のストレージを断片保管先として利用できる．利用者環境にすでにある余剰ストレージリソースを提供し合うことで大きなストレージ容量を確保するため，ストレージの購入，外部契約におけるコストを削減できる．

2.2.4 性能

提案するシステム構成では秘密分散を導入したことにより，分散ストレージ製品で用いられる手法より，データの分散にかかる演算コストが大幅に増加する．既存の秘密分散を用いたサービスでは分散処理をインターネット上のサーバで代行するものや専用機器で行うものがあるが，負荷が集中しやすかった．

これに対して，提案構成では非集中化して，クライアント側での分散・復号処理を実施することにより処理の分散ができるため，単一端末にかかる性能負荷を低減できる．

2.3 サービス実現上の課題

提案する構成により目的としていたメリットが確保できるが，同時に幾つかサービス実現に向けた課題も残っている．まず，提案する構成でシステムを構築する場合，非集中化システムに対する管理の難しさ，同期タイミング等を考慮した実装の複雑さなどが考えられる．また，クライアントのストレージリソースを利用するため，リソース毎の容量や可用性の差を考慮した処理ロジックがないと，効率的な利用や高い可用性の確保が困難となる．

特に提案サービスの有効性への影響が強いため，クライアントストレージの利用効率化と可用性確保に関する課題に対して優先的に取り組む必要がある．

3. 関連技術

本章では取組課題やその対策の検討を行う前提となる技術の説明を行う．

3.1 秘密分散技術

秘密分散とは, 入力される平文データ M を複数の分散情報（断片）に分散させ，一定量の断片が集まった場合にのみ平文データ M を復号できるようにする暗号化手法である[1]．鍵の管理によって情報の機密性を確保する公開鍵，共通鍵暗号系と異なり，断片データの保管箇所やアクセス管理によって，平文データ M の機密性・可用性・完全性を確保できる．その実現方法として Shamir[2]や Blakley[3], Krawczyk[4]など複数の方式が提案されている．本研究では, データの暗号化に利用しやすく，断片の情報量を削減できる「(k, L, n) しきい値ランプ型秘密分散」[1]の利用を前提として検討を行う．

(k, L, n) しきい値ランプ型秘密分散では，入力される平文データ M を n 個の断片に分散(秘匿化)し．k 個の断片を持ち寄ることで平文 M の復元(復号)を可能にする．n-k 個の断片については断片保管の冗長化として利用できる．ここで，k-L 個以下の断片からは M を推測できず，k-L 個以上かつ k 個未満の断片を揃えることで, 平文 M の推測が可能になる．L の値は大きいほど推測が容易になるが，断片 1 つのサイズが小さくなるため，利用者のニーズに合わせて 1≦L≦k の間で設定する必要がある．この際，一つの断片のサイズは平文 M の情報量 m[bit]を用いて, $\frac{1}{k-L+1}\times$ mとなり，秘密分散により生成される断片の総情報量は n 個の断片の和$\sum_1^n(\frac{1}{k-L+1}\times m)=\frac{1}{k-L+1}\times m\times n$となる．つまり，秘密分散を用いることで情報量は入力される平文データの$\frac{n}{k}$からn倍に増加する．

秘密分散による機密性向上効果を確保し，可用性も同時に高めるため，一般的に$n>k>L\geq 1$, $n\geq 3$, $k\geq 2$が用いられる．n = kの場合，断片が 1 つでも紛失すると M が復号できなくなり可用性が低下し，k = Lの場合，秘密分散による総断片サイズがm × nと平文保管時の n 倍に増加するためストレージ利用の効率が低下するため$n>k>L\geq 1$となる．更に，$n\leq 2$の場合，前述条件$n>k$を考慮すると k の取りうる値は k=1 に制限されるが，この場合単一の断片からの復号が可能となり機密性の確保ができない．

3.2 稼動率計算

システムの稼動率は，システムに搭載される各機能(処理)の稼動率を元に計算することができる．システムが直列搭載される機能 A，B から成り立つ場合，それぞれの機能の稼動率K(A)，K(B)とすると，システム全体の稼動率はK(A) × K(B)となる．システムが直列搭載される機能 C，D から成り立つ場合，それぞれの機能の稼動率K(C)，K(D)とすると，システム全体の稼動率は$1-(1-K(C))\times(1-K(D))$となる．

秘密分散の断片保管では複数の k 個以下の断片が揃わない場合，復号ができないため，k 個以下の断片については直列システムとみなし，各断片の必要時のアクセス可能性＝稼動率の積によって評価する．

3.3 非集中化

本研究における「非集中化」は処理や負荷の集中箇所を持たずに一つの機能を実現するシステムと定義する[5]．一般的に実行されるアプリケーションは単一端末で処理されるため，端末の停止がアプリケーションの停止に直結される．これを複数端末に分散させた物に分散システムがあるが, 分散システムでも処理種別毎に実行端末を変えるなど，単一故障点が存在する場合がある．これに対して，非集中化では端末停止による影響はサービスに波及せず，停止した端末に閉じることを前提に構成を考える．

非集中化ストレージサービスでは，複数端末のストレージリソースを利用するが，冗長化により単一端末が停止してもサービスは継続でき，負荷の集中も発生しないため，非集中化サービスと呼ぶものとする．

4. 非集中化ストレージサービスの課題

秘密分散を用いた「非集中化ストレージサービス」を実現する際の課題として，大きく 2 点技術課題を定義する：

課題 1：分散先ストレージの容量差を考慮した構成

課題 2：分散先ストレージの可用性を考慮した構成

4.1 課題 1：分散先ストレージの容量差を考慮した構成

本課題は多様な機器に断片を保管する際に生じる．例として，PC，NAS，インターネットストレージを断片の保管先として利用する環境（環境 1）を考えると，それぞれの端末で保管できる容量に当然差が生じる．容量差がある場合，断片の配置方法を工夫しないと，各ストレージリソースを全て使い切れず, リソース利用効率の低下が発生する．

分散ストレージは断片保管先端末数 T が復号に必要な断片数 k に比べて，十分に多いこと(T > n > k)を前提とし，端末に保管する断片数の調整によりストレージサイズの差を吸収する構成(**図 4-1**)を取っている．

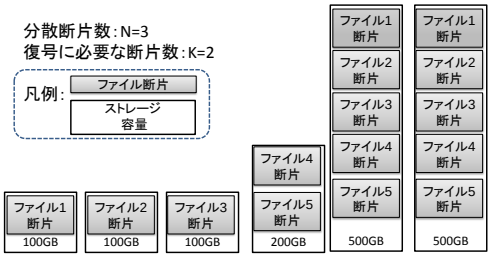

図 4-1：HDFS におけるストレージサイズ差の活用

しかし，家庭や小規模オフィス等のようにコストの観点で多数のストレージを保持・利用しづらい環境では，この手法を用いることができない．秘密分散で利用する n,k を考慮すると，n ≥ 3台のデータ保管端末が望ましいが，前述環境 1 を利用する場合，保管端末数 n=3 となり全てのストレージに平文ファイルの断片を保管する必要がある．断片サイズは全て同じであるため，各ストレージに断片を保管すると，保管できる断片量の和 total(S)は最小ストレージのサイズmin(S)を用いて，total(S) = n × min(S)となる．ここで**図 4-2** 上のようにmin(S)と他のストレージのサイズ差が大きい場合，それらのストレージリソースは効率的に利用できない．

また，容量が最大のストレージの容量が他のストレージ容量の和より大きいなど容量差が大きい場合(**図 4-2** 下)も最大ストレージの容量を断片数の配置調整によって使い切れず，ストレージリソースを全て効率的に利用できない．

異なるサイズの断片保管先ストレージを利用する場合，それぞれのストレージリソースを最大限活用するため，最適な断片保管方法を検討する必要がある．

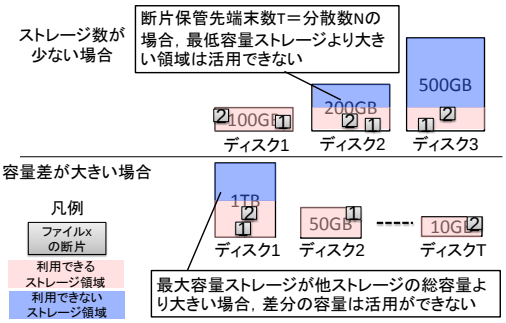

図 4-2：ストレージ容量差を活用できないケース

4.2 課題 2：分散先ストレージの可用性を考慮した構成

本課題は断片保管先として端末の接続，切断が頻繁に発生する場合に生じる．効率よく余剰リソースを集めて非集中化ストレージサービスを実現するため，移動端末を含む多種の端末リソースを利用できることが望ましい．しかし，移動端末などを用いた場合，持ち運び時にネットワークから切断される，電源を切断されるといった事象が発生しやすく，断片の稼動率低下を考慮する必要がある．

秘密分散システムでは，n-k 個の断片保管ストレージが切断されても，継続して読み出しを行うことができる．しかし，ストレージ毎の可用性が全体システムに与える影響はわからず，断片保管先を選定した際に利用者が求める可用性要件を達成できるかどうか判断する方法がない．そこで，端末毎の可用性が全体システムに与える影響を導出する方法を検討する必要がある．

5. 提案手法

5.1 課題 1：ストレージ利用効率の向上

容量差のあるストレージを効率的に使うため，それぞれのストレージサイズに応じた情報量を保管する方法を考える．その際，利用効率を確保するため，可用性を低下させる分散手法も検討対象とし，その可用性を確保する方式を 5.2 節で検討するものとする．

各断片保管先ストレージに対し，同サイズの断片を同数保管する際に課題 1 のストレージ利用効率低下が発生する．これを回避するため，各ストレージに保管する断片の情報量(＝断片のサイズ*断片数)をストレージ容量にもとづき，変化させる方法が有効と考えられる．

ストレージの容量に応じてそこに保管する断片の情報量を上下させるため，断片のサイズ*断片数を変化させる必要がある．しかし，一つの断片のサイズは秘密分散技術の特性から$\frac{1}{k-L+1} \times m$[bit]となり同一の秘密分散パラメータ n,k,L を利用する環境では全ての断片保管ストレージに同一となる．

そこで，本研究では断片のサイズではなく，断片数に対する工夫によってストレージ利用効率の向上を目指す．一般的には全ての断片保管ストレージに同数の断片を保管するが，ストレージ利用の効率化のため，ストレージサイズに応じて保管する断片数を変更する方法を提案する．

5.1.1 分散断片数の最適化

ストレージサイズに応じて保管する断片数を調整する方法として，接続される各ストレージ容量の最大公約数等を用いて n を求める方式を提案する．

事前条件としてストレージの容量などを下記のように定義する．

断片保管ストレージ数=T,

i 番目のストレージサイズ=S_i,

ストレージ配列$S = \{S_1, \ldots, S_T\}$,

ここで n を下記のように定義ことで，断片数を増やす．

$$n = \frac{\sum_{i=1}^{T} S_i}{\gcd(S)}$$

$\gcd(S)$では各断片保管ストレージに共通して保管できる断片データの最大サイズを求める．断片保管ストレージの全体容量$\sum_{i=1}^{T} S_i$を$\gcd(S)$で割ることで，全体の容量 100%利用するための断片数を導出できる．断片保管ではストレージ i には$\frac{S_i}{\gcd(S)}$個の断片を置くものとする．

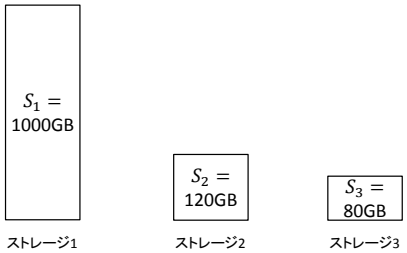

図 5-1 ストレージ構成例

n 導出の例として**図 5-1** のストレージ条件を考える．n をストレージ数とした場合，各ストレージに同サイズのデータを保管することが必要となり，3 番目のストレージに保管する断片サイズ 80[GB]が頭打ちとなり，システムに保管できる情報量は$80[\mathrm{GB}] \times 3 = 240[\mathrm{GB}]$が上限であった．一方，システムで保持している総ストレージサイズは$\sum_{i=1}^{T} S_i = 1000[\mathrm{GB}] + 120[\mathrm{GB}] + 80[\mathrm{GB}] = 1200[\mathrm{GB}]$あるため，ストレージの利用効率は$\frac{240}{1200} = 0.2$と 2 割にとどまる．

しかし，提案手法の n を用いると，$\gcd(S) = 40$が共通する最大断片サイズとなり，全体容量$1200[\mathrm{GB}]$を割ると$n = 30$が導出される．この n に従い断片を 30 個生成し，ストレージ 1 には 25 個，ストレージ 2 には 3 個，ストレージ 3 には 2 個の断片を配置することになる．平文情報 M を上記の割合で分散保管することで，断片保管ストレージを同じ割合で消費することができ，ストレージ利用率は同時に 100%に達することができる．

5.1.2 復号に必要な断片数の最適化

5.1.1 節では断片数の最適化により，断片保管ストレージの数や容量に依存せず，そのストレージを 100%利用可能であることを示した．しかし，秘密分散による秘密分散による情報量の増加を抑え，機密性を確保するため，復号に必要な断片数 k についても考慮が必要となる．

5.1.1 節で定義する n と 3.1 節から k の値域は$\mathrm{n} = \frac{\sum_{i=1}^{T} S_i}{\gcd(S)} > \mathrm{k} \geq 2$となる．この際，k は大きいほど秘密分散による情報量の増加を制限できる．本研究では最大容量のストレージの容量$\max(S_i)$を用いて，$\mathrm{k} > \frac{\max(S_i)}{\gcd(S)}$とすることにより，最大容量のストレージに保管した断片数$\frac{\max(S_i)}{\gcd(S)}$のみで復号できない構成を機密性に対する要件と定義する．

具体的な k の数値は課題 2 の可用性に対しても影響をおよぼすため，その考慮を次節で紹介する．

5.2 課題 2：ストレージ可用性の向上

非集中化ストレージサービスの可用性を検討するためにはまず，断片を保管するストレージの稼動率を確認する必要がある．各断片保管ストレージの稼動率を踏まえて，冗長化が必要な箇所と不要な箇所を選定し，利用者が求めるサービスの稼動率を確保する．

ストレージ利用効率の向上のため，5.1.1 節では同一断片保管先ストレージに複数の断片を保管する方式を検討した．当方式では多数の断片を保管するストレージが存在しうる．このようなストレージがアクセス不能になると，非集中化ストレージサービス全体の稼動率に大きな影響をおよぼす．

例えば，**図 5-1** のストレージ 1 の稼動率が低い場合，ストレージ 1,2,3 の 3 台のストレージに断片を保管していたとしても．多数の断片を保管するストレージ 1 が停止すると，残りの端末保管される断片は$\frac{S_2+S_3}{\gcd(S)} = 5$個しかアクセスできなくなる．一方，5.1.2 節で定義する機密性を確保した復号に必要な断片数は$\mathrm{k} > \frac{S_1}{\gcd(S)} = 25$個となるので，アクセスできる断片だけでは非集中化ストレージサービスを継続提供できない．

この状況から，ストレージ利用効率と可用性の確保を両立する方式には復号に必要な断片数 k について矛盾が生じることがわかる．このため，両立ではなく，可用性・利用効率の低下を容認できるかによって，効果的にストレージ字を配置できる方法を検討する．

利用者が求める可用性と利用効率を満たす k の導出に向け，まず断片保管ストレージを「切断を想定しないもの」と「切断を想定するもの」に分類し，「切断を想定するもの」の内，「同時に切断しうる数」を考えたい．これらの前提条件を踏まえて，利用者が管理するストレージを**図 5-2** のように整理する．

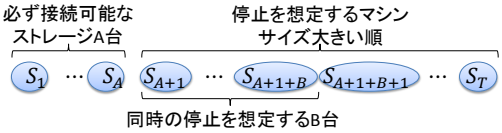

図 5-2 ストレージの構成

図では左側$S_1, \cdots, S_A$には NAS や操作端末など，ストレージの停止=サービスの停止利用者了承できるストレージ「切断を想定しないもの」の容量を入れる．$S_{A+1}, \cdots, S_T$には PC やモバイル端末等のストレージサイズ「切断を想定するもの」を入れ，サイズの順に並べる．その時ストレージ配列は下記のように記載できる．

ストレージ配列$S = \{S_1, \ldots, S_A, S_{A+1} \ldots, S_T\}$

$$\therefore for\ \forall S_i\ where\ i > A\ \ S_i \geq S_{i+1}$$

このストレージ配列の採用より，サービスとしての可用性を考える際に同時に切断しうる数 B を用いて$S_{A+1}, \cdots, S_{A+1+B}$の端末に保管される断片が利用できなくても継続利用できることが，k を求める条件と整理できる．そこで，$S_{A+1}, \cdots, S_{A+1+B}$に保管される断片が無くても復号ができる k は

$$\mathrm{k} = \mathrm{n} - \frac{\sum_{i=A+1}^{A+1+B} S_i}{\gcd(S)} = \frac{\sum_{i=1}^{A} S_i + \sum_{i=A+B+2}^{T} S_i}{\gcd(S)}$$

となる．ただし$\max(S_i) > \sum_{i=1}^{A} S_i + \sum_{i=A+B+2}^{T} S_i$となる場合は，5.1.2 節の k に対する条件を満たせず，秘密性を担保するため，$\max(S_i)$のストレージを「切断を想定しないもの」とするか，条件を満たすように小規模ストレージを追加する必要がある．

このサービスでは総ストレージサイズ$\sum_{i=1}^{T} S_i$に対して保管できる平文のサイズ（=サービスとして提供できるスト

レージ容量）は$\sum_{i=1}^{T} S_i = \frac{1}{k-L+1} \times \mathrm{m} \times \mathrm{n}$から

$$\mathrm{m} = \sum_{i=1}^{T} S_i \frac{k-L+1}{\mathrm{n}} = \sum_{i=1}^{T} S_i \frac{\frac{\sum_{i=1}^{A} S_i + \sum_{i=A+B+2}^{T} S_i}{\gcd(S)} - L + 1}{\frac{\sum_{i=1}^{T} S_i}{\gcd(S)}}$$

$$= \sum_{i=1}^{T} S_i \frac{\sum_{i=1}^{A} S_i + \sum_{i=A+B+2}^{T} S_i + (1-L)\gcd(S)}{\sum_{i=1}^{T} S_i}$$

$$= \sum_{i=1}^{A} S_i + \sum_{i=A+B+2}^{T} S_i + (1-L)\gcd(S)$$

となる．また，各断片保管ストレージの容量は 100%利用可能であるが，平文保管容量に対する利用効率は

$$\frac{\sum_{\mathrm{i=1}}^{\mathrm{A}} \mathrm{S_i} + \sum_{\mathrm{i=A+B+2}}^{\mathrm{T}} \mathrm{S_i} + (1-\mathrm{L})\mathrm{gcd(S)}}{\sum_{\mathrm{i=1}}^{\mathrm{T}} \mathrm{S_i}}$$

となる．

ここまでで，利用者が停止を許容する条件を設定できた場合，システムが提供できるストレージ容量と，ディスク利用効率を導出できるようになった．これに加え，サービス全体の稼動率も導出することによって，利用者が許容すべき条件を決める補助をしたい．

3.2 節稼動率の計算方法から，断片保管先ストレージ i の稼動率をθ_iとすると，サービス全体の稼動率θは

$$\theta = \prod_{i=1}^{\mathrm{A}} \theta_i \times \partial$$

となり，ここで$\partial = 1 - \max(\frac{\sum_{i=A+1}^{A+1+B} S_i}{\gcd(S)}$ 以上の断片を持つ端末が故障する確率)となる．∂は$\sum_{i=1}^{T-B} C_{(T-B,i)}$通りの故障組合せから$1-(1-\theta_x)\times(1-\theta_y)\times\cdots$が最大になる組を抽出することで導出する．抽出自体は最適化問題などの手法を利用することができる．

提案した手法により，利用者が自環境にあるストレージをどのように組み合わせるか，可用性(稼動率)と利用効率をどの様に分配するかを考える際の参考情報を提供できるようになった．

6. 性能評価

6.1 提案手法を用いない周辺機能の性能検証

2.1 節にて提案する秘密分散の処理機能として，秘密分散専用機器上で処理する部分をメインに開発し，提案手法による断片数の拡張を行わない状態で実機による性能検証を行った．開発機能とは CIFS により受け取ったファイルを秘密分散し, PC の共有フォルダに書き込むプログラムである．当プログラムではシーケンシャルアクセスのみならず，ランダムアクセスにも対応するため，データのブロック化など実運用に適した機能を追加している．

当プログラムを汎用 CPU(**表 6-1**)で実行した際の処理性能は**図 6-1**,**図 6-2** のとおりであった．評価では開発プログラムに対して各ファイルサイズのデータを連続書き込みし，平均性能を確認している．

表 6-1 評価端末性能

	CPU	Mem	NIC
マイクロサーバ	2core 2.2GHz	4GB	1Gbps
シングルボード PC	4core 1GHz	1GB	100Mbps

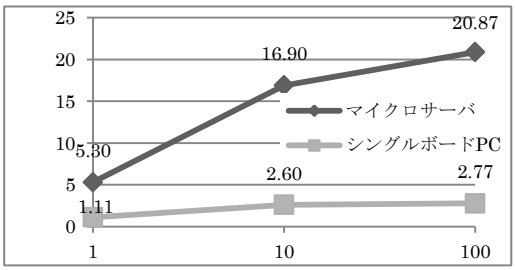

図 6-1 秘密分散書き込み性能(縦:Mbps,横:データ MB)

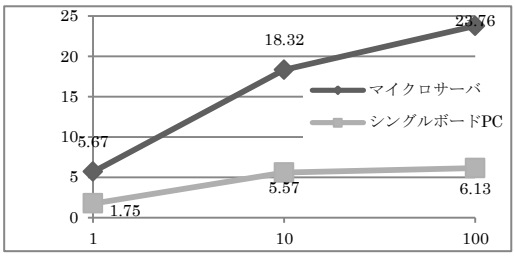

図 6-2 秘密分散読み出し性能(縦:Mbps,横: データ MB)

評価結果の傾きはファイルサイズが小さい場合，秘密分散処理のイニシャルコストが高く，性能が低下するためと考えられる．またシングルボード PC とマイクロサーバとの性能差は，シングルボード PC の処理バスの狭さなどが影響しているためと考えられる．

この性能結果から，全ての処理負荷を専用機器で行う場合，企業で使うストレージの要件を**表 6-2** のように定義した場合, 小容量ファイルを処理すると性能要件を達成せず，その他の場合も性能マージンの確保が難しいことがわかる．しかし，単一利用者に対しては十分な性能を確保できると考えられるため，本サービスの実現には検討システム構成の通り，性能を確保するためファイル操作を行うクライアント上で秘密分散処理を実行し，負荷を分散することが必要であることがわかる．

表 6-2 企業ストレージ要件想定

項目	要件
1 利用者が 1 日に行うファイル操作	100 回/（利用者・日）
平均ファイルサイズ	5MB
業務時間	8 時間
1 利用者利用時の性能要件	0.14Mbps
100 利用者利用時性能要件	13.88Mbps

6.2 提案手法を用いた場合の性能予測

本研究で提案する断片数の変更を行った状態での性能は机上で評価した．提案手法ではファイルの断片数を端末数より大幅に増加させる場合がある．断片数 n が大きい場合，秘密分散の処理エンジンに応じて性能が大きく変化する．再検証・変更が容易なよう 6.1 節の実機検証で利用したエンジンではなく，市中のオープンソースの秘密分散プログラムとそこに搭載されるエンジン[6]を元に性能を考える．

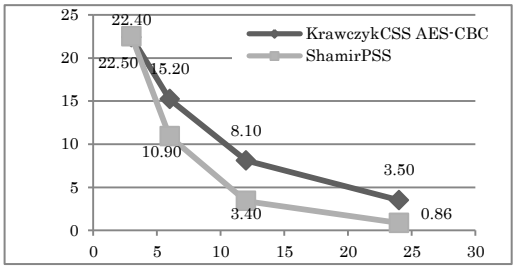

図 6-3 分散断片数と秘密分散書き出し性能
（縦：MB/s 横：断片数）

市中の秘密分散エンジンを利用して，分散断片数を増やした際の秘密分散性能を評価した結果を**図 6-3** に示す．評価は CPU:Core i7 4600U，Mem:8GB，HDD：5400rpm の環境で 10MB のファイル分散を 10 回実施し，他プロセスの割り込み影響が小さいと考えられる最速値を用いている．

図 6-3 から断片数が増えると性能が大幅に悪化することがわかる．提案手法をサービス化するためには断片数を増加させた際にも高速で処理が可能となる秘密分散処理の検討が必要であることがわかった．

また，秘密分散では書き出す複数の断片をメモリに保持した状態で分散処理を行うため，n を大きくするに従い保持残片数が増加し，多くのメモリを消費する．この問題に対応するためファイルをブロック化して分散することでメモリ仕様の効率化が考えられる．

7. まとめ

秘密分散技術により情報を断片化することにより機密性，可用性を向上し，余剰ストレージに断片を保管することによりコストを削減する非集中化ストレージサービスとその構成方法を提案した．当システムの利用者環境では断片保管先に利用するストレージ間で容量・可用性の差が想定される．この差によってリソースの利用効率や，可用性について発生しうる問題への対策を検討・提案した．提案手法により，利用者が求めるリソース利用効率と可用性を提供するための要件が導出できる．

また，非集中化構成によるサービス機能のアーキテクチャを考察し，性能的な実現性を確認した．提案手法を模擬した性能の評価から，既存の秘密分散エンジンを用いた場合，性能課題があることがわかった．非集中化システムの開発などでも，プログラム間の設定同期などの実装上の課題が想定される．これらの課題については，継続して検討していきたい．

参考文献

[1] 山本博資 "秘密分散法とそのバリエーション" 数理解析研究所講究録 1361 巻 2004 年 19-31，入手先 <http://www.kurims.kyoto-u.ac.jp/~kyodo/kokyuroku/contents/pdf/1361-3.pdf> (参照 2016-04-28).
[2] Shamir, Adi "How to share a secret", Communications of the ACM 22 (11): 612–613 入手先 < http://cs.jhu.edu/~sdoshi/crypto/papers/shamirturing.pdf > (参照 2016-04-28).
[3] Blakley, G. R. "Safeguarding cryptographic keys". Proceedings of the National Computer Conference 48: 313–317.
[4] Krawczyk, Hugo (1993). Secret Sharing Made Short . CRYPTO '93.
[5] Decentralized system https://en.wikipedia.org/wiki/Decentralised_system (参照 2016-05-11).
[6] Archistar-smc https://github.com/Archistar/archistar-smc (参照 2016-05-11).
[7] Hadoop http://hadoop.apache.org/docs/r2.7.2/ (参照 2016-05-11).