L-014

P2P 型自律分散ストレージ上のデータへの秘密分散法の適用による耐検閲性の向上

Improvement of Resistance to Censorship by Applying Secret Sharing to Data on P2P Autonomous Distributed Storage

清水 耕太郎[1)]　松澤 智史[2)]　宮本 暢子[2)]
Kotaro Shimizu　Tomofumi Matsuzawa　Nobuko Miyamoto

1　序論

近年，多くの Web サイトが商用クラウドストレージを利用して運用されるようになっている．クラウドストレージは利便性が高いものの，商用サービス提供企業によるサービス内容変更や倒産，不正利用や検閲などのベンダーリスクを抱えている．そこでリスクの対策として商用ストレージに代わって IPFS[1] や StorJ[2] のような P2P 型の自律分散ストレージを利用することが考えられる．P2P 型の自律分散ストレージは有志による参加者がストレージをノードとして提供し，アルゴリズムに従ってアップロードされたデータを分割して各ノードに保管する．全体の管理をアルゴリズムが行うため，特定の管理者を持たず，ベンダーリスクが小さい．しかしながらこの上で Web サイトをホスティングしようとすると，通常暗号化せず平文のままアップロードするため，各ノードの管理者により容易に検閲，削除が可能であるという問題が残る．意図的な削除は可用性を低下させうるため避けるべきである．本研究では耐検閲性を高め，ファイルの可用性を向上させるため，P2P 型分散ストレージ上のファイルについて秘密分散法を利用することを提案する．秘密分散法を利用することにより，各ノードに保管されるファイルの断片が無意味化され，スクレイピングによる検閲を防ぐことができる．また秘密分散法による暗号化器，復号器を作成しファイルの暗号化，復号にかかった時間を測定した．

2　関連研究

先行研究では P2P 型自律分散ストレージ上で不特定多数にファイルを公開するユースケースにおける耐検閲性は，冗長化による削除耐性で担保しており [1]，精査可能性に言及するものは無かった．福光ら [4] の研究 (2016) では P2P 型の秘密分散ストレージ上へ秘密分散法を用いてデータを分割し，さらに匿名通信を援用したことで秘匿性に優れたストレージ技術を提案している．これは個人の秘匿しておきたいデータを対象としたものであり，復号にもパスワードを要する．Web サイトのデータのように不特定多数に公開するデータを対象としていなかった．また暗号化，復号の速度についての評価は無かった．そこで本研究では不特定多数に公開するデータを対象とし，暗号化，復号の速度の評価を行う．

3　用語

3.1　検閲

ここではノードの管理者により，保管しているファイルを精査し，不適当であると判断したファイルを管理者が管理するノードから削除することをいう．耐検閲性を高めるためには暗号化などにより精査可能性を低下させるか冗長化により削除耐性を向上する 2 つのアプローチが考えられる．

3.2　秘密分散法

秘密にしておきたいデータ「シークレット」をシークレットから生成される複数のデータ「シェア」に分散し，シェアのいくつかが揃わなければシークレットが復元できない仕組みを秘密分散法という．特に n 個に分割したうち k 個が揃わなければシークレットを復元できないようなシェアの生成方法を (k,n) －閾値秘密分散法という．(k,n)-閾値秘密分散法の構成法はいくつかあるが，今回はこの中でも Shamir の秘密分散法 [3] を使用する．

4　既存の自律分散ストレージの耐検閲性

まず精査可能性であるが，平文のままであるので単純なスクレイピング等により精査は可能である．次に削除耐性であるが，削除耐性は採用されているデータ管理モデルに影響を受けるため，代表的な以下の 2 のモデルについて考える．1 つ目は IPFS が採用している，単一のノードにファイルをアップロードするモデル (以下単一型という)．2 つ目は StorJ が採用している，複数のノードにファイルをアップロードするモデル (以下複数型という) である．単一型は 1 つのノードにファイルをアップロードした後，他のノードからファイルにアクセスがあった際にアクセスしたノードにキャッシュを生成する．そのため，他のノードからファイルにアクセスがありキャッシュが作成されるまでは，アップロードしたファイルに冗長性はない．最初にアップロードした先のノード管理者がファイルを検閲し削除した場合，ファイルは永久に失われる．複数型はファイルのコピーや冗長性をもたせた断片を複数のノードにアップロードするため単一型よりも削除耐性が高い．またノード管理者がファイルの所有者の許可なくファイルを削除した場合，ペナルティを受ける (ノード参加の権限停止など) よう対策されているサービスもある．複数のノードが同一の主体により運営されている場合や，ペナルティを受けた管理者が別のアカウントで参加する場合があるため確実とはいえない．

5　提案手法

本研究では，秘密分散法を用いることで P2P 型自律分散ストレージ上で公開される web サイトの耐検閲性を高める手法を提案する．

5.1　公開

公開時の手順は以下のようになる．

1. 公開したいファイル s に Shamir の秘密分散法を適用し，n 個のシェアファイルに分割する．
2. 保存先のノードを n 個選定する．選定法は任意である．

1）東京理科大学 創域理工研究科 情報計算科学専攻
2）東京理科大学 創域理工学部 情報計算科学科

3. ノード 1 個に対し，シェアファイルを 1 個アップロードする．これを n 個のノードに対して行う．
4. 公開したいファイルのハッシュ値，各シェアファイルのハッシュ値，保存先のノードをメタデータを管理するノード (ストレージノードではない) に登録する．
   メタデータを管理するノードは，アクセスしたいファイルのハッシュを渡すとそのシェアファイルのハッシュと保存先ノードのアドレスを返す役割をもつ．

5.2 閲覧

ファイルにアクセスする流れは以下のようになる．

1. アクセスしたいファイルのハッシュ値をメタデータを管理するノードに問い合わせ，シェアファイルを保存している各ノードのアドレスと各シェアファイルのハッシュ値を受け取る．
2. 各ノードにシェアのハッシュ値を送り，対応するシェアファイルをダウンロードする．
3. Shamir の秘密分散法の復号を行い，元のファイル s が復元される．

5.3 検閲耐性

検閲耐性は，特定のシェアファイルから残りのシェアファイルと，その保存先を特定できるかによって変化する．特定できない場合，k 個のシェアファイルは集まらず，復号できない．よって精査の時点で検閲は不可能である．次に特定できる場合，これは具体的にはメタデータを管理するノードがアクセスしたいファイルのハッシュだけでなくシェアファイルのハッシュに対しても残りのシェアファイルのハッシュと保存先のノードを開示してしまう仕様になっている場合などが挙げられる．この場合，検閲者の保有する以外の残る $k-1$ 個のシェアファイルをダウンロードすることで復号できてしまう．しかし，一切暗号化を行っていなかったときの検閲は自ノード内のスクレイピングで完結していたのに対し，シェアファイルのダウンロードと復号というネットワークコスト，計算コストを検閲者に対し課すことができる．そのために特定のシェアファイルから残りのシェアファイルが特定できる場合においても一定の検閲抑止効果がある．

6 実験と考察

与えられたファイルを Shamir の秘密分散法により暗号化及び復号するプログラムを作成し，サイズの違ういくつかのファイルの暗号化，復号にかかる時間を測定した．実装言語は高速性とメモリ安全性から Rust を採用した．与えられたファイルの先頭から 1byte ずつ暗号化，復号する．計算量の観点から，復号にはラグランジュ補間を利用している．実数上で計算するとラグランジュ補間の際に浮動小数点演算の特性上丸め誤差により誤った値を補完しまう (ルンゲ現象) ため，ガロア体 GF(2) の 8 次拡大体 GF(2^8) 上で計算している．秘密分散法のパラメータは分割するシェアの数が $n=3$，復元に必要なシェアの数が $k=2$ となる $(2,3)$-閾値秘密分散である．

実行環境は表 1 の通りである．実行結果は表 2 のように，2 種類のファイルサイズ両方の場合において暗号化処理よりも復号処理にかかる時間が長かった．ファイル 1 では 32 倍，ファイル 2 では 72 倍である．

表 1 実行環境

CPU	AMD ryzen5 3600 4.2[GHz]
OS	Ubuntu 20.04.5 LTS
メモリ	16[GB] DDR4 2667[MHz]
コンパイラ	Cargo 1.65.0
コンパイルオプション	–release

表 2 実行結果

	暗号化	復号
ファイル 1(34.3KB)	427[ms]	13,762[ms]
ファイル 2(106KB)	583[ms]	42,033[ms]

100KB ですら 42 秒復号に掛かっていることから，実際の web ページの運用には耐えられない．復号に比べて暗号化が早く終わっていることから，復号時の処理の多くはファイル IO でなく，ラグランジュ補間の計算に費やされている．

7 今後の予定

利便性向上のためにハッシュ以外でのファイル検索機能を取り入れたいが，検索可能性はそのまま検閲の精査可能性に繋がってしまう．検索性と耐検閲性の両立を模索したい．また各シェアの暗号化，復号は byte 単位で独立して行われているため byte 単位での並列化が可能である．今回のプログラムは並列処理を行っていなかったため，並列化を用いれば実行時間の短縮が期待できる．今後は並列化により実用的な復号速度を実現できるかを調査したい．

8 結論

本研究では，秘密分散法を用いることで P2P 型自律分散ストレージ上で公開される web サイトの耐検閲性を高める手法を提案した．公開したいファイルに秘密分散法を適用し，シェアファイルをそれぞれ別のノードにアップロードすることで，検閲者が他のシェアを特定できない状況では検閲が不可能になる．入手できる場合でも検閲の抑止効果が望める．

参考文献

[1] Juan Benet, ”IPFS - Content Addressed, Versioned, P2P File System”, arXiv, cs.NI, 1407.3561, (2014)
[2] S. de Figueiredo, A. Madhusudan, V. Reniers, S. Nikova and B. Preneel, ”Exploring the storj network: A security analysis”, ACM, SAC ’21: Proceedings of the 36th Annual ACM Symposium on Applied Computing, pp. 257 - 264, (2021)
[3] A. Shamir: How to Share a Secret, Communications of the ACM, Vol.22, No. 11, pp. 612 – 613, (1979)
[4] 福光 正幸, 長谷川 真吾, 岩崎 淳也, 酒井 正夫, 高橋 大樹, ”秘密分散法と匿名通信による秘匿性に優れた P2P 型ストレージ技術の提案”, 情報処理学会研究報告マルチメディア通信と分散処理（DPS）, 2016-DPS-166, vol. 6, pp. 1-8, (2016).