機密性・完全性を保証するファイルシステムにおけるアクセス制御方法の提案

3U−06

藤田 智成　小河原 成哲

NTT 未来ねっと研究所

1 はじめに

我々は，携帯電話，PDA のような移動を伴う計算機を含めた全ての計算機環境に対し，連続的なストレージリソースを提供できるネットワークファイルシステムの研究を進めている．

広域環境における性能，可用性などを考慮すると，そのアーキテクチャは一つのストレージリソースから構成される集中型ではなく，複数のものから構成される分散型が望ましい．しかし，単一の組織が世界中にストレージリソースを配置し，管理することは困難であるため，お互いに信頼関係を持っていない複数の組織が管理するストレージリソースを統合し，協調動作させることが必要となる．

このように他組織のストレージリソースを利用する環境では，その管理者を含めた第三者によるファイルの不正な閲覧，改竄を防ぐために，データの機密性と完全性を保証する機能をファイルシステムは備えていなければならない．更に，複数の組織を横断した環境でも機能するような，組織内に制限されないファイルのアクセス制御方法が必要となる．

本稿では，機密性・完全性を保証するネットワークファイルシステムにおける，組織に依存しないファイルのアクセス制御方法を提案する．

2 既存方法の問題点

ファイルのアクセス制御は，ユーザの識別，その情報に基づいたアクセス制御の二段階に分けることができる．

従来のネットワークファイルシステムは，第一段階のユーザの識別が，各々の組織が管理するユーザに関するデータベースに依存しているため，アクセス制御の機能範囲が組織内に制限される．更に，そのデータベースへのアクセスが許されているのは特別な権限を持つ管理者のみであるため，一般のユーザは自分が所有するファイルであっても，そのファイルへの他のユーザのアクセスを許可することができないなど，ユーザが持つアクセス制御機能は限定されている．

本研究と同様に，お互いに信頼関係を持たない複数組織のストレージリソースの協調動作を目的とする OceanStore[1] は，上述のような中央集権的モデルのアクセス制御方法を提案している．しかし，中央集権的モデルを多様なユーザ，組織が共存するインターネット全体に拡大することは現実的でない．そのような環境下では，PGP(Pretty Good Privacy) 等で用いられる，各ユーザが自由に用いることのできる権限を持つ「ユーザ中心信頼」と呼ばれるモデルが適している．

組織に依存しない「ユーザ中心信頼モデル」のアクセス制御方法として，CapaFS[3] はファイルの所有者がアクセス権を付与する相手に対してトークンを発行し，サーバはそのトークンの正当性を確認することでユーザのアクセス権を判断する方法を提案している．しかし，そのアーキテクチャは，ファイルとサーバの関係が一対一に制限されており，広域環境で不可欠となるファイルを他のサーバにキャッシングするような動作を実現することができない．

Swallow[2] は，サーバがユーザの正当性を判断せずに，全てのユーザに対してファイルを提供する方法を提案している．これは，ユーザが暗号化したファイルをサーバに保存しているため，ファイルを復号化し，その内容を読むことができるのは，正当な鍵を保持しているユーザのみであるという性質を利用したものである．つまり，暗号鍵自身にユーザ識別を必要としないアクセス制御機能を持たせたものと言える．しかし，この方法はファイルの読み込み権の制御のみを実現するものであり，ファイルの書き込み権の制御は他の方法が必要となる．また，ファイルを復号化するための鍵を第三者に渡すことで，その第三者もファイルを復号化することができるようになるため，ファイルを複数のユーザで共有した場合，共有グループに所属している権利 (グループ所属権) と共有グループのユーザ構成を変更することができる権利 (グループ構成変更権)，という二種類の権利を個別に制御することができないという問題がある．

3 提案するアクセス制御方法

本研究のファイルシステムは，ファイルを暗号化することで機密性，デジタル署名を用いて完全性を保証している．メッセージ認証コード (MAC) を用いて完全性を保証する方法も考えられるが，MAC はコードの生成と正当性の確認に同じ鍵を用いるため，ファイルのアクセス制御に適用すると，読み込み権，書き込み権という二種類の権利を個別に管理することができない．

提案する方法は，上述の Swallow と同様に，ファイルが暗号化されているという特性を利用した，鍵自身によってアクセス制御を実現する方法である．デジタル署名に用いる一組の非対称鍵暗号の公開鍵と秘密鍵の鍵ペアを用いることで，読み込み権と書き込み権の各々を制御する．本手法は，一方向の衝突困難ハッシュ関数を用いて，デジタル署名に用いる鍵ペアから導出した値を機密性を保証するための対称鍵暗号用の鍵として利用することで，必要な鍵数を減らした，という特徴がある．

An access control mechanism under network file system providing confidentiality and data integrity

Tomonori FUJITA, Masanori OGAWARA

NTT Network Innovation Laboratories

1-1 Hikarinooka, Yokosuka-Shi, Kanagawa, 239-0847, Japan

{fujita, ogawara}@exa.onlab.ntt.co.jp

また，本手法では鍵自身のアクセス制御にトークンの概念を用いたサーバでのユーザ認証を加えることで，ファイルを共有した場合のグループ所属権と構成変更権を個別に制御することを可能とした．本手法のトークンの実装方法は，上述の CapaFS と異なり，ファイルの保存されるサーバが制限されない，という特徴がある．

3.1 ユーザ識別とアクセス制御

各ユーザは鍵ペア (K_u, K_u^{-1}) を所有し，世界中でユニークな識別子として使用する．また，サーバとファイル自身も，それぞれ固有の鍵ペア (K_s, K_s^{-1})，(K_f, K_f^{-1}) を持つ．なお，ディレクトリのアクセス制御もファイルと同様に行う．

以下では，ユーザ識別用鍵ペアとして (K_{u1}, K_{u1}^{-1}) を用いるユーザ u1 がファイルへの書き込み・読み込み権の両方を持ち，(K_{u2}, K_{u2}^{-1}) を用いる u2 がファイルへの読み込み権のみを持つ例 (図 1) を用いて，提案するアクセス制御方法について説明する．

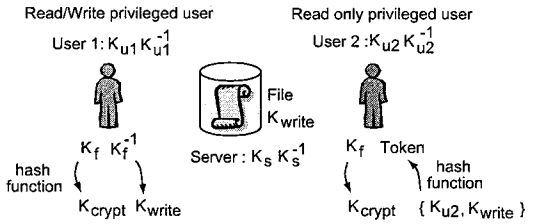

図 1: 各要素が保有する鍵

3.1.1 ファイル書き込み

ファイル書き込みは，新規作成と既存ファイルの変更の二種類に分けられる．これらの操作は，一般のファイルシステムと同様に，対象となるディレクトリに対する書き込み権を持っている場合に許可される．

ユーザは新たなファイルを作成した際，そのファイルに対する鍵ペア (K_f, K_f^{-1}) を既存ファイルの鍵ペアから選択するか，又は新たに生成する．次に一方向のハッシュ関数を用いて，K_f^{-1} からファイル書き込み認証鍵 K_{write} を導出し，サーバに保存する．K_f^{-1} を用いてファイルに署名し，K_{write} と同様にハッシュ関数を用いて K_f から導出したファイル暗号化鍵 K_{crypt} を対称鍵暗号の鍵として，ファイルを暗号化し，暗号化したファイルをサーバに保存する．

サーバにおける書き込みに関するアクセス制御は，書き込みを要求するユーザが対象ファイルの K_{write} を所有しているかどうかによって行われる．つまり，K_{write} を所有することが，そのファイルに対する書き込み権を持つことを意味する．

3.1.2 ファイル読み込み

ファイルの読み込み権を持つユーザは K_f とトークンを持つ．本研究でのトークンは K_{write} と自分の公開鍵を組み合せた値からハッシュ関数を用いて導出した値である．ファイル読み込みの際，ユーザはサーバから受け取ったファイルをハッシュ関数を用いて K_f から導出した K_{crypt} で復号化し，更に K_f を用いて署名の正当性を確認する．

サーバにおける読み込みに関するアクセス制御はトークンを用いて行われる．次にその具体的な手順を以下に示す．U は u2，S はサーバでの処理を表す．

U → S　ファイルパス，K_{u2}，トークンを送信する．

S　サーバに保存されているファイルパスに対する K_{write} と送られてきた K_{u2} を組み合せた値のハッシュ値を求めて，トークンと比較し，トークンの正当性を確認する．

S → U　通信を暗号化するためのセッション鍵 $K_{session}$ を生成し，K_{u2} を使って暗号化したメッセージを送信する．

U　K_{u2}^{-1} を用いてメッセージを復号化し，$K_{session}$ を入手する．

U → S　$K_{session}$ のハッシュ値を送信する．

S　$K_{session}$ のハッシュ値と，送られてきたハッシュ値と比較し，ユーザが K_{u2} と組となる K_{u2}^{-1} を所有していることを確認する．

この手順により，サーバはトークンが u2 に対して作成された正当なものであることが確認できる．もし，トークンが第三者の手に渡った場合も，第三者が u2 の秘密鍵 K_{u2}^{-1} を入手しない限り，サーバから送られてくる K_{u2} を使って暗号化されているメッセージを復号化できない．

トークンは K_{write} とユーザの公開鍵を組み合せた値から導出した値であるため，K_{write} を知らない第三者は作成することができない．従って，グループ所属権はファイル読み込み権，グループ構成変更権はファイル書き込み権に対応し，それぞれの権利を個別に制御することができる．

また，ファイル読み込み，書き込みのどちらのアクセス制御方法もサーバ自身の鍵ペアを用いていないため，ファイルを保存するサーバが制限されない．

3.2 他のインフラストラクチャとの協調

本研究では，鍵やトークンを用いてアクセス制御を実現しているため，他のユーザとファイルを共有する際などには，それらを交換する必要がある．提案した方式は特別な機能に依存していないため，電子メールや Instant Messenger など，任意の方法と組み合せて利用することが可能である．

4 まとめ

本稿では，機密性・完全性を保証するファイルシステムにおいて，ファイルの暗号鍵自身によって権利を表現し，ハッシュ関数を利用してサーバがユーザの正当性を確認する，組織に依存しないファイルのアクセス制御方法を提案した．

参考文献

[1] John Kubiatowicz, et al. Oceanstore: An architecture for global-scale persistent storage. In *the Ninth international Conference on Architectural Support for Programming Languages and Operating Systems*, Nov 2000.

[2] D. Reed and L. Svobodova. Swallow: A distributed data storage system for a local network. In A. West and P. Janson, editors, *Local Networks for Computer Communications*, pp. 355–373. North-Holland Publishing Company, 1981.

[3] Jude T. Regan and Christian D. Jensen. Capability file names: Separating authorisation from user management in an internet file system. In *11th USENIX Security*, 2001.