情報処理学会論文誌　Vol. 51　No. 11　2066–2080 (Nov. 2010)

# アクセス制御統合による性能低下を改善する階層型アクセス制御方式

中村隆喜[†1,†2]　亀井仁志[†1]
山本彰[†3]　薦田憲久[†2]

アクセス制御を統合する場合の課題である，アクセス制御処理の性能低下を改善する，階層型アクセス制御方式を提案する．本論文では特に Windows ACL を Linux などの UNIX 系 OS に実装し，統合する場合の性能低下を取り扱う．提案方式は，ファイルの所有者，もしくはそのファイルへ高頻度のアクセスが予想されるユーザやグループのアクセス制御情報の一部をファイルの基本属性に格納することにより，ほとんどのファイルオペレーションを拡張属性にアクセスすることなく，実行可否の判定を可能とする．限られた基本属性領域にどのような情報を格納するのがよいのかを検討し評価した結果，代表的アクセスマスク設定パターンに基づく情報を格納するのが最も効果が高いことが分かった．

# A Hierarchical Access Control Method to Minimize Performance Degradation by Access Control Integration

Takaki Nakamura,[†1,†2] Hitoshi Kamei,[†1]
Akira Yamamoto[†3] and Norihisa Komoda[†2]

We propose a hierarchical access control method that minimizes the degradation of access control processing performance that occurs in cases when access controls are integrated. We mainly focus on a case in which Windows ACL is implemented on UNIX operating systems such as Linux. Our method makes it possible to process access control of almost all file operations without accessing the conventional ACL on extended attributes. It does so by storing part of the access control information about file owners or frequent-access users into basic attributes. We evaluate effective means of using access control information in basic attributes and conclude that storing the information about basic access mask setting patterns is the most effective of these.

## 1. はじめに

近年，ファイルを用いた情報共有の活性化にともない，ファイルサーバとストレージ装置を一体化した Network Attached Storage（NAS）の市場が拡大している．一般的な NAS がサポートする代表的なファイルプロトコルは Network File System（NFS）と Common Internet File System（CIFS）である．NFS は Linux をはじめとする UNIX 系 OS との親和性が高く，CIFS は Windows OS との親和性が高い．

それぞれの OS とファイルプロトコルは独立に発展してきたため，様々な仕様の違いが存在する．その中でも，ファイルのアクセス制御は両プロトコルを扱う NAS において統合が困難な仕様の 1 つである．その理由は，アクセス制御に関する仕様が NFS と CIFS で大きく異なることに加えて，同一ファイルに対して両プロトコルからアクセスされる場合を考慮して，セキュリティと従来との互換性の両面をふまえた統合仕様を策定する必要があるためである．

NAS における最も代表的なアクセス制御の仕様は，パーミッション，Windows ACL の 2 種類である．パーミッションは NFS で一般に利用されるアクセス制御仕様であり，そのアクセス制御情報は固定サイズである．Windows ACL は CIFS で一般的に利用されるアクセス制御仕様であり，そのアクセス制御情報は可変サイズである．Windows ACL は NTFS ACL，CIFS ACL とも呼ばれる．

パーミッションと Windows ACL を比較すると，基本的には Windows ACL の方がより情報量が多く高機能であるが，パーミッションに対して上位互換の仕様となっているわけではない．パーミッションと Windows ACL のアクセス制御情報と制御プログラムの統合は，Linux Samba [1]，NetApp FAS [2]，Microsoft Windows Storage Server [3]，Sun Solaris 10 [4] などで実現されている．

セキュリティを重視してアクセス制御の統合を行った場合，NFS 単体で運用していたファイルサービスのアクセス性能が統合により低下することがあった．これはアクセス制御処理

†1 株式会社日立製作所システム開発研究所
Systems Development Laboratory, Hitachi Ltd.
†2 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technologies, Osaka University
†3 株式会社日立製作所研究開発本部
Research & Development Group, Hitachi Ltd.



© 2010 Information Processing Society of Japan

が，NFS のアクセス制御処理から，より情報量が多く高機能な統合アクセス制御処理に置き換わるためである．既存の統合方式は，この統合によるアクセス性能低下を考慮できていなかった．本論文では，このアクセス性能低下を改善する方式について検討を行う．

Linux をはじめとする UNIX 系 OS のファイルは，基本属性，拡張属性，ユーザデータから構成される．基本属性の格納領域は固定サイズであり，拡張属性とユーザデータの格納領域は可変サイズである．パーミッションの制御情報は基本属性の 1 つであり，基本属性領域に格納されるのが一般的である．これに対し統合アクセス制御情報は Windows ACL の仕様により可変サイズとなるため，拡張属性領域に格納せざるをえない．したがって統合されたアクセス制御処理では，拡張属性領域への読み込みが発生し，またその読み込んだ情報を保持しておくためのメモリが必要となるため，処理性能が低下する．

そこで本論文では，拡張属性領域に格納される統合アクセス制御情報の一部情報を基本属性領域に格納し，その一部情報のみで先にアクセス制御処理を実行することにより，性能低下を改善する階層型アクセス制御方式を提案する．

本論文の以降の構成は次のとおりである．2 章では，アクセス制御の統合時の課題である，アクセス性能低下の改善について説明する．3 章では，提案方式である階層型アクセス制御方式を説明する．4 章では，基本属性領域に格納する情報に関する複数の格納方式について検討し，模擬実験に基づき最適な格納方式を評価する．5 章では，提案方式により見込まれる性能改善効果を検討し，提案方式を実装したシステムにおいて性能改善効果を確認する．6 章で関連研究について述べ，最後に 7 章で，本研究のまとめを述べる．

## 2. アクセス制御統合によるアクセス性能低下

NFS，CIFS 両プロトコルのアクセス制御に関する仕様の違いのうち，アクセス性能に関する項目を抜粋したものを**表 1** に示す．アクセス制御に関する仕様の違いは，文献 5)，6) にも詳しい．

両プロトコルの仕様を統合する場合，#1–#4 は情報がより豊富な仕様つまりこの場合 CIFS の仕様にあわせれば運用上の問題が少ない．CIFS 仕様にあわせた場合のアクセス制御情報の格納位置がどのように変化するかを**図 1** に示す．

統合前の NFS 単独システムにおいて，ファイルは固定長の基本属性領域と可変長のユーザデータ領域から構成される．アクセス制御情報は，ファイルの所有者識別子である UID と，所有グループ識別子である GID と，所有者，所有グループ，その他ユーザのアクセス権を規定するパーミッションからなり，これらはすべて基本属性に格納される．

**表 1**　NFS，CIFS でのアクセス制御仕様の違い
Table 1　Difference of access control specification between NFS and CIFS.

| # | 項目 | NFS (パーミッション) | CIFS (Windows ACL) |
|---|---|---|---|
| 1 | 各ACEマスク種類 | 3種類 (rwx) | 14種類 |
| 2 | ACL長 | 3エントリ固定<br>(owner, group, other) | 可変(最大64KByte, 典型的ACEサイズ36Byteでは最大1820エントリ) |
| 3 | 許可・拒否種別 | 許可のみ | 許可もしくは拒否 |
| 4 | ユーザID | UID, GID (それぞれ4Byte) | SID (28Byte) |
| 5 | アクセス制御情報の格納領域 | 一般に基本属性領域 | 一般に拡張属性領域 |

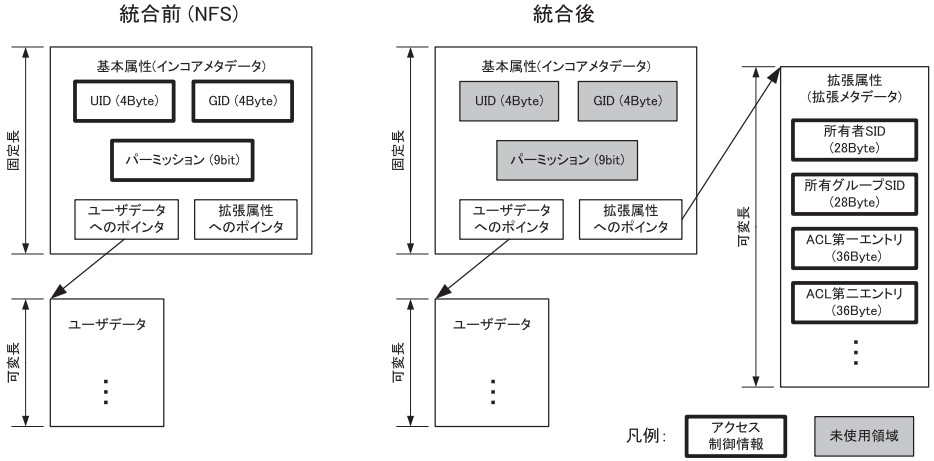


**図 1**　統合前後のアクセス制御情報の格納位置の変化
Fig. 1　Difference of storing area for access control information.

これに対し，統合後のシステムにおいて，ファイルは基本属性領域，ユーザデータ領域，に加えて可変長の拡張属性領域から構成される．これは CIFS のアクセス制御情報が可変であるため，統合後のアクセス制御情報も可変となるためである．統合後のアクセス制御情報はファイルの所有者識別子である所有者 SID，所有グループの識別子である所有グループ SID，各 SID に対するアクセス権を規定する ACL エントリからなる．ここで，ACL エントリ数は可変である．

統合前は基本属性の情報だけでアクセス制御処理が実行可能であるが，CIFS 仕様にあわ


© 2010 Information Processing Society of Japan

せて統合する場合，たとえ NFS であっても拡張属性から統合アクセス制御情報を読み込んだ後，アクセス制御処理をする必要がある．これにより，ファイルオペレーションで実行されるアクセス制御処理の性能が低下してしまうことがある．またパーミッション情報が，拡張属性上のアクセス制御情報に統合されてしまうため，パーミッション参照更新オペレーションの応答性能が低下してしまう．

次章以降では，主に前者のアクセス制御処理の性能低下を改善する方式について検討する．ただし，後者のパーミッション参照更新オペレーションの性能低下の改善も提案方式に関係するため，あわせて検討する．

## 3. 階層型アクセス制御方式

ファイルへのアクセスパターンを考えた際，あるファイルに対してアクセスするユーザは偏りがあり，またそのファイルに対して実行されるオペレーションも偏りがあることが予想される[7]．

たとえば，アクセスユーザに関しては，個人オンラインドライブとしての使用目的であれば所有者のアクセス頻度が最も高く，共有ドライブとしての使用目的であれば共有範囲の主グループに属するユーザのアクセス頻度が高いことが予想される．ファイルオペレーションに関しては，ディレクトリであればルックアップ要求が高く，ファイルであればデータ参照要求が高いことが予想される．

そこで本論文では，アクセスの確率の高いユーザ，また実行される確率の高いオペレーションに関するアクセス制御情報を基本属性に格納し，アクセス制御処理を基本属性の情報のみで先に実行する階層型アクセス制御方式を提案する．

**図 2** に提案方式のアクセス制御情報の格納位置と処理概要を示す．基本属性のみでアクセス制御処理を可能とするため，所有者，所有グループに関しては CIFS 仕様の SID は用いず，UID，GID とした．UID，GID と SID のマッピングは従来技術である winbind[8] を利用する．また，所有者や所有グループに関するアクセス制御情報の一部を基本属性に格納するために，統合前にパーミッション領域として使用していた 9 bit の領域を，部分アクセス制御情報格納領域として利用する．何の情報をこの部分アクセス制御情報格納領域に格納するかによって，性能改善効果に差が出る．

本方式により，一部のオペレーションは基本属性の情報を参照するだけでアクセス可否が判断でき，アクセス制御方式の統合による性能低下の改善が見込まれる．

本方式の適用によって，アクセス制御情報の更新処理が多少複雑となるデメリットがある

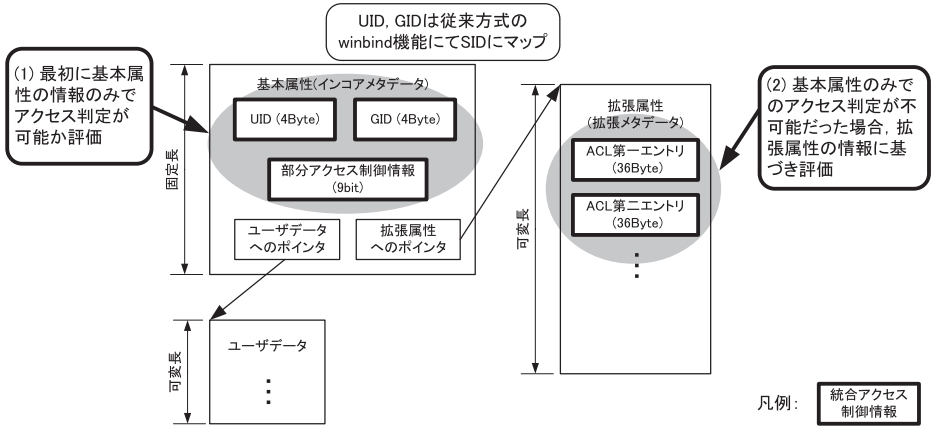


**図 2**　提案方式のアクセス制御情報の格納位置
Fig. 2　Storing area for access control information on proposed method.

が，更新処理の性能支配主要因はディスクアクセスであり，処理の複雑さ自体はほとんど性能に影響を及ぼさない．

またパーミッション参照更新オペレーションの応答性能低下についても，本提案の部分アクセス制御情報の格納方式を工夫すれば改善が可能である．ただし，格納方式の組合せによっては，本性能低下改善のために同オペレーションのユーザビリティを犠牲にしなければならないこともある．本性能低下の改善については，4.9 節で検討を行う．

なお，提案方式は，既存の UNIX 系 OS で，パーミッションをネイティブアクセス制御方式とするファイルシステムを統合元とした場合を主な検討対象としている．また，本論文ではパーミッションベースのファイルシステムを Windows ACL 対応とするための検討を中心としている．これに加えて，POSIX ACL[20] や NFSv4 ACL などのパーミッションより豊富な情報を持つアクセス制御仕様に対応する場合にも考え方を参考にできる．また，NFSv4 ACL[9] などの高機能 ACL をすでにネイティブサポートしている ZFS[10] のようなアクセス制御統合済みファイルシステムにも考え方は適用できる．さらに，本論文ではアクセス制御情報を階層化の対象としているが，これ以外のファイルの属性情報にも考え方を参考にできる．


© 2010 Information Processing Society of Japan

## 4. 部分アクセス制御情報の格納方式

本章では，従来のパーミッション領域である 9 bit にどのような部分アクセス制御情報を格納するのが効果的かを検討し，比較評価する．4.1 節で格納方式の概要を述べる．4.2，4.3 節で比較評価方法と比較する格納方式について述べる．4.4 節から 4.8 節で各条件における比較評価結果を述べる．4.9 節で格納方式がパーミッション参照更新オペレーションの性能低下に対して与える影響について述べる．4.10 節で比較評価結果をまとめる．

### 4.1 格納方式の概要

本節では，部分アクセス制御情報領域の格納方式の概要を述べる．Windows ACL のアクセス権の種類は 14 種類であり，つまりアクセスマスクは 14 bit であるため，仮に 1 ユーザもしくは 1 グループの情報に限定した場合にも，すべてを部分アクセス制御情報領域にはマップできない．

所有者からアクセスされる可能性が非常に高い場合は，所有者のアクセス制御情報の一部を部分アクセス制御情報領域の 9 bit すべてに割り当てる方式が考えられる．複数のユーザからアクセスされる可能性が高い場合には，9 bit を 2 つの領域に分割して所有者，所有グループのアクセス制御情報を格納する方式が考えられる．さらに，所有者，所有グループ，その他ユーザに対してそれぞれ 3 bit の部分アクセス制御情報領域を割り当てる方式も考えられる．

領域を分割して格納する場合，それぞれで異なる格納方式を用いてもかまわない．つまり，たとえば，所有者についてはある格納方式を用い，所有グループとその他ユーザについては別の格納方式を用いるなども可能である．どの領域にどの格納方式を用いるかは，ファイルシステムプログラムもしくはファイルシステムメタデータに記録させることにより実現する．

基本格納方式としては，以下の 3 種類が考えられる．

**(1) 高頻度使用アクセスマスクビット格納方式（most frequent USED access mask bit storing style）：以降，格納方式 U**

**図 3** に格納方式 U の概要を示す．格納方式 U は 14 種類のアクセスマスクビットの中から高頻度で使用されるアクセスマスクビットの上位を基本属性の部分アクセス制御情報領域に格納する．

たとえば，ファイルの参照要求の頻度が高い場合は，Windows ACL の `FILE_READ_DATA` ビットを優先的に部分アクセス制御情報領域に格納する．本方式はファイルオペレーション

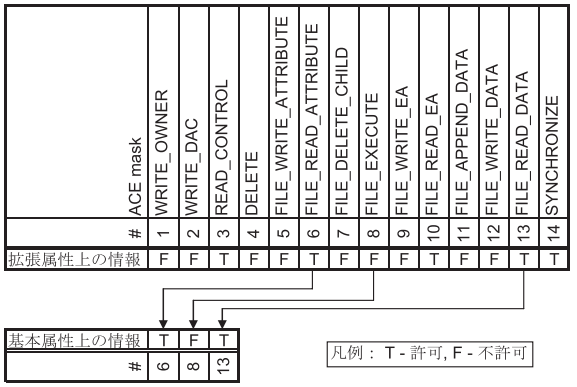


**図 3**　格納方式 U の概要
Fig. 3　Overview of storing style U.

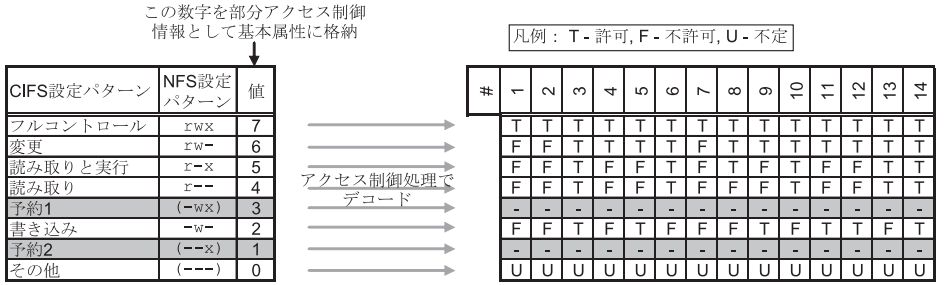


**図 4**　格納方式 S の概要
Fig. 4　Overview of storing style S.

に偏りがあり，使用されるアクセスマスクビットにも偏りがある場合は有効である．

**(2) 高頻度設定アクセスマスクパターン格納方式（most frequent SET access mask pattern storing style）：以降，格納方式 S**

**図 4** に格納方式 S の概要を示す．格納方式 S は高頻度で設定されるアクセスマスクパターン（代表的設定パターン）の情報を基本属性の部分アクセス制御情報領域に格納する．Windows ではアクセス許可の簡易設定パターンとして「フルコントロール」「変更」「読み取りと実行」「読み取り」「書き込み」の 5 種類がある．本格納方式では，たとえばこれら簡易設定パターンを代表的設定パターンとし，これらの設定パターンに合致している場合は


© 2010 Information Processing Society of Japan

その値を部分アクセス制御情報領域に格納する．上記簡易設定パターンのみを格納する場合は，5 種類 + それ以外を意味する 1 種類 = 合計 6 種類 $< 2^3$ であるため 3 bit で表現可能である．アクセス制御処理時には部分アクセス制御情報の値より図右のアクセスマスクパターンにデコードしアクセス可否を判断する．本方式はアクセスマスク設定パターンに偏りがある場合は有効である．

代表的設定パターンの数よりも部分アクセス制御情報格納領域が大きい場合，余剰ビットを格納方式 U として用いる方法も考えられる．つまり，格納方式 S と格納方式 U を組み合わせて用いる方式である．

**(3) アクセスマスクビット論理積格納方式（logical production [AND] of access mask bit storing style）：以降，格納方式 A**

**図 5** に格納方式 A の概要を示す．格納方式 A は複数のアクセスマスクビットの論理積を基本属性の部分アクセス制御情報に格納する．たとえば，`FILE_READ_ATTRIBUTE` ビットと `FILE_EXECUTE` ビットの論理積を部分アクセス制御情報の 1 bit に割り当てる．これにより，多くのアクセスマスクビット情報を部分アクセス制御情報領域に格納可能である．本方式は各アクセスマスクビットが真となる確率が高い場合に有効である．

本格納方式を用いる場合，以下のような工夫が重要となる．同時設定確率が高いアクセスマスクビット群を論理積対象とする．高使用確率のアクセスマスクビットは論理積対象とせずそのまま部分アクセス制御情報領域の 1 ビットを割り当てる．低使用確率のアクセスマスクビットは論理積対象から除外し，基本属性領域には格納しない．

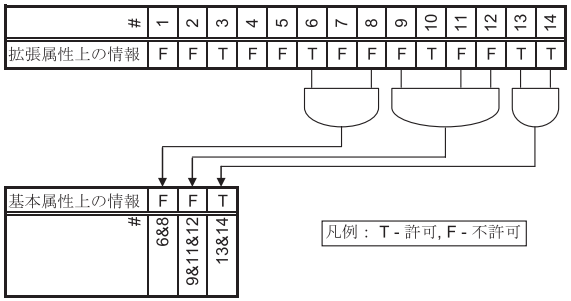


**図 5**　格納方式 A の概要
Fig. 5　Overview of storing style A.

**その他の格納方式**

以上述べた基本格納方式 U，S，A 以外にも，各システムコールやネットワークプロシージャの実行可否を各ビットにマップし格納する方式（格納方式 P）や，全アクセスマスクビットを格納する方式（格納方式 F）も考えられる．格納方式 P については，システムコール，ネットワークプロシージャの種類は数十以上あり，ユニークなものだけを抽出したとしても他の格納方式と比べて非効率であるため，以降の評価対象からは外した．また格納方式 F については，基本属性領域に十分な空き領域があれば最も優れた格納方式であることは間違いないが，すでに述べたとおり基本属性領域の空き領域は有限であり，以降では最も一般的な 9 bit 以下で格納可能な方式を議論するため，今回の評価対象からは外した．

### 4.2　格納方式の比較評価方法

あるファイルアクセスパターンを仮定した場合，ファイルアクセスパターンは複数のファイルオペレーションから構成される．それぞれのファイルオペレーションではアクセス権チェックに用いられるアクセスマスクが規定される．ファイルオペレーションごとに規定されたアクセスマスクの許可・非許可の状態は，格納方式に応じて，部分アクセス制御情報のみで決定可能な場合と，決定不可能な場合がある．

ファイルアクセスパターンにおける各ファイルオペレーションの実行割合が分かれば，部分アクセス制御情報のみで決定可能なファイルオペレーションの割合が算出できる．部分アクセス制御情報のみで決定可能なファイルオペレーションの種類は，格納方式ごとに異なるため，この部分アクセス制御情報のみで決定可能なファイルオペレーションの割合が評価指標となる．以降では，仮定するファイルオペレーションの割合として，NAS の代表的な性能ベンチマークソフトである SPECsfs 2008 を参考にする[11]．

NFS については所有者特権の関係で，所有者と非所有者とでは，各ファイルオペレーションで用いられるアクセスマスクが異なる．そこで以降では，(1) NFS 所有者，(2) NFS 非所有者，(3) CIFS，(4) NFS，CIFS 混在時所有者，(5) NFS，CIFS 混在時非所有者の 5 つのメニューを評価する．

### 4.3　比較評価する格納方式

4.1 節の基本格納方式に基づき，以下の格納方式の中から比較する．

**[格納方式 U$n$]** よく使用されるアクセスマスクビットの上位 $n$ bit を格納

**[格納方式 S3]** よく設定されるアクセスマスクパターンを 3 bit で表現

**[格納方式 SU$n$]** よく設定されるアクセスマスクパターンの 3 bit に加え，よく使用されるアクセスマスクビットの上位 $n-3$ bit を格納


© 2010 Information Processing Society of Japan

**表 2**　仮定した代表的アクセスマスクパターンとその割合
Table 2　Assumed typical access mask pattern and ratio.

| # | CIFS | 割合 | NFS | 割合 |
|---|---|---|---|---|
| 1 | フルコントロール | 30% | rwx | 40% |
| 2 | 変更 | 30% | rw- | 30% |
| 3 | 読み取りと実行(フォルダの内容の一覧表示) | 30% | r-x | 20% |
| 4 | 読み取り | 5% | r-- | 10% |
| 5 | 書き込み | 5% | -w- | — |

**[格納方式 A3]** なるべく多くのアクセスマスクビットの論理積を 3 bit にマップし格納

なお，これ以外にも $n \geq 4$ での格納方式 S$n$ や格納方式 A$n$ を評価することも考えられるが，本論文では以下の理由で行わなかった．格納方式 S$n$ については，基本的に格納方式 S3 と同様の傾向となるが，S3 と S$n$ との定量評価結果は，想定環境でのセキュリティレベルに大きく依存する．その中でも一般的な部門ファイル共有のセキュリティレベルにおいては 3 bit でも代表的なアクセスマスクパターンは十分カバーできるため，4 bit 以上にしても効果は小さいと判断した．格納方式 A$n$ については，ビット数が大きくなるにつれ，格納方式 U$n$ との違いがほとんどなくなる．

評価軸は，データセット全体に対する代表的アクセスマスク設定パターンの割合とした．この評価軸では格納方式 U$n$，S3，SU$n$ は，一意に評価指標の算出が可能であるが，格納方式 A3 については仮定を設定しなければ算出不能である．そこで A3 の評価指標の算出にあたり以下の 2 つの仮定を設定した．

1 つ目の仮定は，代表的設定パターンの具体例とその割合である．これは著者らの実験環境での割合を参考にし，**表 2** のように設定した．

2 つ目の仮定は，非代表的設定パターンの論理積が真となる確率である．これは代表的設定パターンの論理積が真となる確率の平均値の半分を期待値とした．また，論理積がすべて偽となる場合と，論理積がすべて真となる場合の最悪，最良ケースも合わせてグラフにプロットした．

なお，1 つ目の仮定である代表的設定パターンの割合を変えた場合でも，以降の比較グラフでの中心線は多少変動するが，最悪，最良ケースはほとんど変動しない．

### 4.4　所有者の NFS によるアクセスの評価

NFS の所有者は所有者特権により 5 bit あれば完全にすべてのオペレーションのアクセスマスクがカバーできる．つまり格納方式 U5 ならばどんな状況でも基本属性の情報のみでアクセス可否の判定が可能となる．具体的に，その 5 種類のアクセスマスクは使用割合の高い順に，FILE_EXECUTE，FILE_READ_DATA，FILE_WRITE_DATA，FILE_APPEND_DATA，DELETE である．

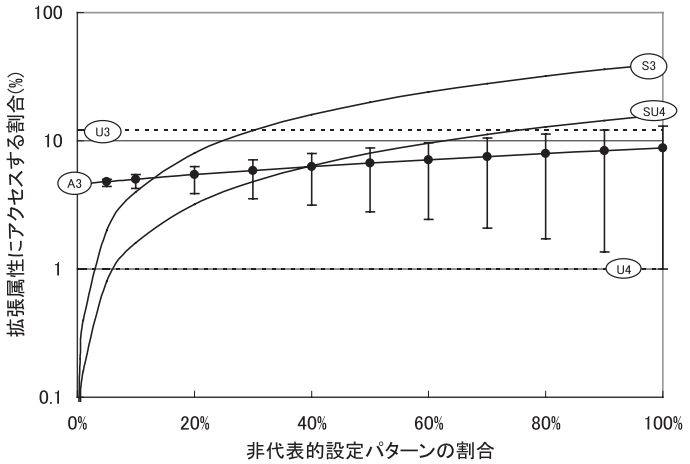


**図 6**　所有者の NFS アクセスの評価結果
Fig. 6　Evaluation result of NFS access by owner.

**図 6** に，格納方式 U3，U4，S3，SU4，A3 を比較した結果を示す．横軸は非代表的設定パターンの割合，縦軸はファイルオペレーションのアクセス制御処理に拡張属性の情報が必要な割合であり対数スケールである（図 7 以下のグラフも同様）．下であればあるほど望ましい結果となる．A3 のみ期待値に加えて最悪，最良ケースを合わせて示している．

格納領域が 3 bit である場合は，一般的な部門ファイル共有のユースケースでは非代表的設定パターンは数%未満であると考えられるため方式 S3 が良い．ただし，12%以上のファイルのセキュリティを細かく設定することが分かっている場合は方式 A3 が良い．

格納領域が 4 bit 以上である場合は，全体的に低く安定する方式 U4，U5 が良い．

### 4.5　非所有者の NFS によるアクセスの評価

NFS の非所有者は 8 bit あれば完全にすべてのオペレーションのアクセスマスクがカバーできる．つまり格納方式 U8 ならばどんな状況でも基本属性の情報のみでアクセス可否の判定が可能となる．

**図 7** に，格納方式 U3，U4，U5，U6，U7，S3，SU4，SU5，SU6，SU7，A3 を比較し


© 2010 Information Processing Society of Japan

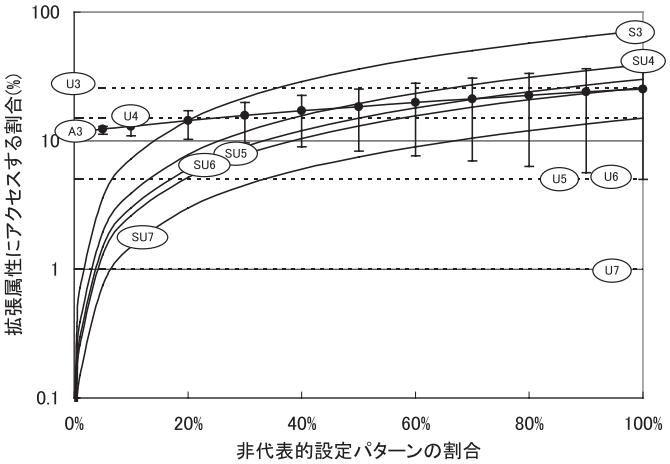


**図 7**　非所有者の NFS アクセスの評価結果
Fig. 7　Evaluation result of NFS access by nonowner.

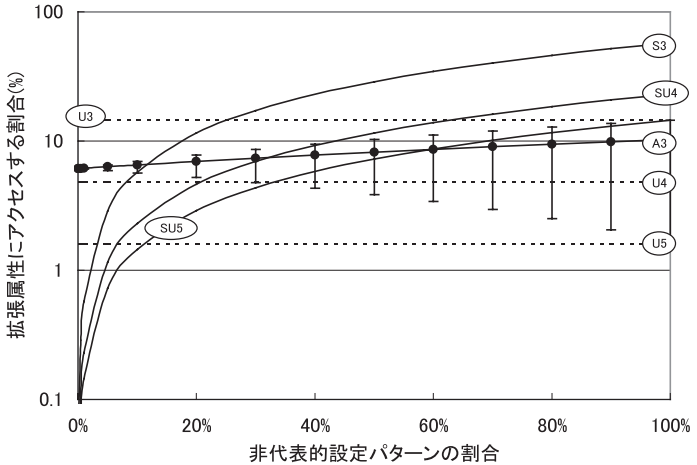


**図 8**　CIFS アクセスの評価結果
Fig. 8　Evaluation result of CIFS access by user.

た結果を示す．

格納領域が 3 bit である場合は，方式 S3 が良い．ただし，20%以上のファイルのセキュリティを細かく設定することが分かっている場合は方式 A3 が良い．可能性は低いと考えるが，50%以上のファイルのセキュリティを細かく設定することが分かっている場合は，条件によっては方式 U3 が良いことがある．

格納領域が 4 bit から 6 bit である場合は基本的には方式 SU4，SU5，SU6 が良い．

格納領域が 7 bit 以上である場合は，全体的に低く安定する方式 U7，U8 が良い．

### 4.6　CIFS によるアクセスの評価

SPECsfs 2008 では CIFS の所有者特権は特に影響しないため，所有者，非所有者とも同様の結果となる．

SPECsfs 2008 の発行するオペレーションでは 6 bit あれば完全にすべてのオペレーションのアクセスマスクがカバーできる．つまり格納方式 U6 ならばどんな状況でも基本属性の情報のみでアクセス可否の判定が可能となる．

**図 8** に，格納方式 U3，U4，U5，S3，SU4，SU5，A3 を比較した結果を示す．

格納領域が 3 bit である場合は，方式 S3 が良い．ただし，12%以上のファイルのセキュリティを細かく設定することが分かっている場合は方式 A3 が良い．

格納領域が 4 bit である場合は，基本的に方式 SU4 が良い．

格納領域が 5 bit 以上である場合は，全体的に低く安定する方式 U5，U6 が良い．

### 4.7　所有者の NFS および CIFS によるアクセスの評価

所有者が NFS と CIFS の両方を用いる場合，8 bit あれば完全にすべてのオペレーションのアクセスマスクがカバーできる．つまり格納方式 U8 ならばどんな状況でも基本属性の情報のみでアクセス可否の判定が可能となる．

**図 9** に，格納方式 U3，U4，U5，U6，U7，S3，SU4，SU5，SU6，SU7，A3 を比較した結果を示す．

格納領域が 3 bit である場合は，方式 S3 が良い．ただし，20%以上のファイルのセキュリティを細かく設定することが分かっている場合は方式 A3 が良い．可能性は低いと考えるが，50%以上のファイルのセキュリティを細かく設定することが分かっている場合は，条件によっては方式 U3 が良いことがある．

格納領域が 4 bit である場合は，基本的に方式 SU4 が良い．

格納領域が 5 bit 以上である場合は，全体的に低く安定する方式 U5–8 が良い．

### 4.8　非所有者の NFS および CIFS によるアクセスの評価

非所有者が NFS と CIFS の両方を用いる場合，10 bit あれば完全にすべてのオペレーショ


© 2010 Information Processing Society of Japan

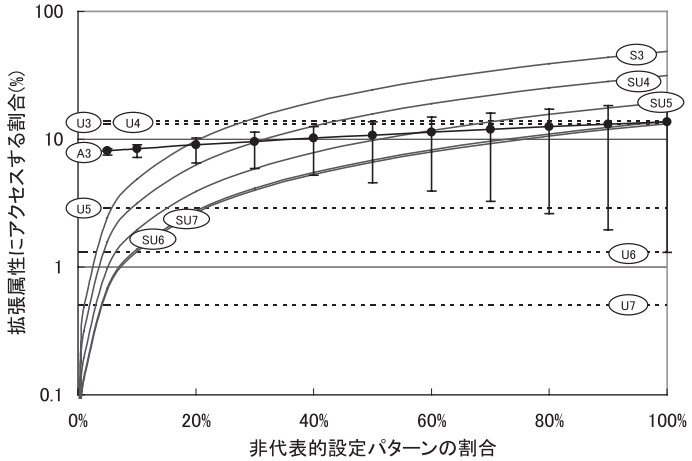


**図 9**　所有者の NFS および CIFS アクセスの評価結果
Fig. 9　Evaluation result of both NFS and CIFS access by owner.

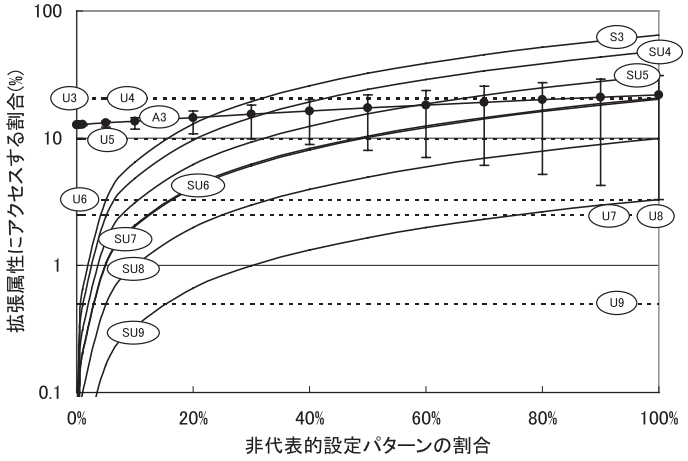


**図 10**　非所有者の NFS および CIFS アクセスの評価結果
Fig. 10　Evaluation result of both NFS and CIFS access by nonowner.

ンのアクセスマスクがカバーできる．つまり格納方式 U10 ならばどんな状況でも基本属性の情報のみでアクセス可否の判定が可能となる．

**図 10** に，格納方式 U3，U4，U5，U6，U7，U8，U9，S3，SU4，SU5，SU6，SU7，SU8，SU9，A3 を比較した結果を示す．

格納領域が 3 bit である場合は，方式 S3 が良い．ただし，22%以上のファイルのセキュリティを細かく設定することが分かっている場合は方式 A3 が良い．可能性は低いと考えるが，45%以上のファイルのセキュリティを細かく設定することが分かっている場合は，条件によっては方式 U3 が良いことがある．

格納領域が 4 bit から 5 bit である場合は，基本的に方式 SU4，SU5 が良い．

格納領域が 6 bit 以上である場合は，全体的に低く安定する方式 U6–10 が良い．

### 4.9　格納方式がパーミッション参照更新オペレーションに与える影響

3 章で述べたようにアクセス制御統合にともない，パーミッション参照更新オペレーションは，拡張属性へのアクセスが必要になり性能が低下する．パーミッション参照更新オペレーションとはたとえば “`ls -l`” や “`chmod`” である．

この性能低下を改善する方針は大きく 2 つある．

第 1 の方針は，パーミッション参照更新オペレーションで必要となる情報を，部分アクセス制御情報に格納することである．たとえば，格納方式 U であれば，パーミッションに対応するアクセスマスクビット（`FILE_READ_DATA`，`FILE_WRITE_DATA`，`FILE_EXECUTE`）を部分アクセス制御情報として格納すればよい．格納方式 A であれば，対応するアクセスマスクビットもしくはその論理積を部分アクセス制御情報領域に格納すればよい．格納方式 S であれば，パーミッションビットが “`- - -`” の場合を非代表的設定パターンとして割り当て，それ以外を従来と同じ NFS の設定パターンに割り当てておけばよい．

格納方式 U と格納方式 A については，アクセス制御処理でよく使用されるビットとパーミッション参照更新オペレーションで使用されるビットが必ずしも一致するわけではないので，この観点ではあまり良い格納方式とはいえない．

格納方式 S については，NFS の設定パターンと CIFS の簡易設定パターンをマップすることにより，アクセス制御処理の性能低下改善とパーミッション参照更新オペレーションによる性能低下改善を両立させることができる．

第 2 の方針は，ユーザビリティを犠牲として性能を改善することである．たとえば “`ls -l`” 時には部分アクセス制御情報のみの情報で応答する．つまり不完全なパーミッション情報の応答を許容する．この方針での性能改善の効果は，採用する格納方式に依存しないというメリットがある．運用への影響を考えるとあまり望ましい方針ではないが，Microsoft Windows


© 2010 Information Processing Society of Japan

**表 3**　評価結果まとめ
Table 3　Summary of evaluation results.

| | 3bit | | 4bit | 5bit | 6bit | 7bit以上 | 完全カバー可能ビット数 |
|---|---|---|---|---|---|---|---|
| | S3-A3境界 | A3-U3境界 | 最適な方式 | | | | |
| NFS所有者 | 12% | なし | U4 | U5 | | | 5 |
| NFS非所有者 | 20% | 50% | SU4 | SU5 | SU6 | U7, U8 | 8 |
| CIFS | 12% | なし | SU4 | U5 | U6 | | 6 |
| NFS,CIFS所有者 | 20% | 50% | SU4 | U5 | U6 | U7, U8 | 8 |
| NFS.CIFS非所有者 | 22% | 45% | SU4 | SU5 | U6 | U7 - U10 | 10 |

Storage Server のように一部のユーザビリティを犠牲にしている実例はある．Microsoft Windows Storage Server のユーザビリティ仕様の詳細については 6 章の関連研究で述べる．

本節で述べた性能低下の影響については 5 章であわせて議論する．

### 4.10　アクセス制御処理性能改善に適した格納方式まとめ

本節では 4.4 節から 4.9 節までの検討結果をまとめる．

**表 3** に示すように 1 ユーザに割り当てる格納領域をパーミッションの 1 ユーザに割り当てるサイズの 3 bit とする場合には格納方式 S3 が最も優れていると考える．非代表的設定パターンの割合が 12〜22%以上の場合には格納方式 A3 が，45〜50%以上の場合には格納方式 U3 が優れていることもある．ただしこの場合は，パーミッション参照更新オペレーションのユーザビリティが犠牲となる．また，非代表的設定パターンの割合が所有者と非所有者で大きく異なり，最適な格納方式がそれぞれ異なる場合には，4.1 節で述べたとおり，所有者と非所有者（所有グループ，その他ユーザ）で格納方式を変えてもよい．

また，1 ユーザに割り当てる格納領域をパーミッション領域のすべてである 9 bit とする場合には格納方式 U が最も優れている．NFS，CIFS 非所有者のパターンを除いて，適切なアクセスマスクを格納すれば，完全に基本属性の情報のみでアクセス可否の判定が可能である．この場合，格納対象者以外のパーミッション参照更新オペレーションのユーザビリティが低下する．

## 5. 提案方式の実装と性能評価

本章では，まず提案方式の実装について述べる．次に，部分アクセス制御情報のみでアクセス判定が可能となった場合，どの程度性能が改善するかを事前見積りし，その見積りを実測により検証する．そして，ある事例における提案方式の改善効果を計算する．最後に，格納方式の選択に対する指針をまとめる．

### 5.1　提案方式の実装

提案方式の実装は Linux 2.6.12.5 + XFS に対して行った．なおアクセス制御機能そのものについては，NEC SC-LX の GPL コードを一部参考にした．

提案方式を組み込んだアクセス制御機能はカーネルモジュールとして実装し，Linux Security Module（LSM）を利用して，カーネルの VFS レイヤが LSM 経由でアクセス制御処理を呼ぶ仕様とした．ただし VFS レイヤだけで完全に Windows ACL 互換のアクセス制御機能を実現するのは困難だったため，一部 Samba サーバ，NFS サーバ，XFS も改造して，アクセス制御カーネルモジュールを呼び出す処理を追加している．また標準の LSM フック関数だけでは一部不足があったので，追加でフック関数を独自定義している．

以上の工夫により，提案方式を他のローカルファイルシステムや他のネットワークファイルシステムプロトコルへ容易に適用できる構造とした．

### 5.2　提案方式により期待される性能改善効果の事前見積り

性能改善効果の見積りを行う場合にも，ファイルアクセスパターンに何らかの仮定をおく必要がある．本節でも 4 章と同様に，仮定するファイルアクセスパターンを SPECsfs2008 とする．

SPECsfs2008 ベンチマークではキャッシュの効果を考えなければ，ランダムディスクアクセス性能と CPU 性能が性能支配要因となる．

ランダムディスクアクセス性能の観点では，ユーザデータアクセス量を $D_U$，統合により追加で発生するアクセス制御情報アクセス量を $D_A$ とすると，アクセス制御統合により，ベンチマーク性能は $D_U/(D_U + D_A)$ となることが予測される．$D_A$ は各ファイルのアクセス制御のリスト長に依存し，アクセス制御が実行される総オペレーション回数を $O_A$，各ファイルの平均アクセス制御情報サイズを $L$ とすると，$D_A = O_A \times L$ となる．したがってベンチマーク性能の予測式は $D_U/(D_U + O_A \times L)$ となる．

たとえば，要求負荷 16 KOPS の場合の $D_U$ は 575 GB，$O_A$ は 2.1 M 回である．$L$ は最小で 4 KB，最大で 64 KB となるため，予測式の値は 0.986（4 KB 時）〜0.815（64 KB 時）となる．つまり，アクセス制御に関する性能低下率は 1.4%（4 KB 時）〜18.5%（64 KB 時）程度になると考えられる．要求負荷にかかわらず $D_U$ と $O_A$ の比率はほぼ一定であるため，上記の性能低下率は要求負荷に依存せず一定である．この要因による性能低下は，提案方式により改善される．

また，3 章，4.9 節で述べたパーミッション参照更新オペレーションの性能低下についても同


© 2010 Information Processing Society of Japan

様に計算すると，性能低下率は 1.3%（4 KB 時）〜17%（64 KB 時）と予測される．パーミッション参照オペレーションのユーザビリティを犠牲にすれば，性能低下率は 0.13%（4 KB 時）〜2%（64 KB 時）となり大幅に軽減される．パーミッション更新オペレーションのユーザビリティも犠牲にすれば，この要因による性能低下は改善される．またすでに議論したように格納方式 S を採用すれば，ユーザビリティを犠牲にせずこの要因による性能低下は改善できる．

ほかにも上記以外のランダムディスクアクセス性能に起因する性能低下要因として，ファイルの作成削除オペレーションの性能低下がある．アクセス制御統合により，これらのオペレーションには，アクセス制御情報自体を作成削除する処理が追加されるためである．同様に計算すれば本要因による性能低下は 0.097%（4 KB 時）〜1.5%（64 KB 時）程度になると考えられる．この要因による性能低下の改善は，提案方式の適用対象範囲外である．

次に，CPU 性能の観点では，処理プログラムの増加による性能低下がある．実装したプログラムのコード量を調査したところ，従来処理に対して 1 割程度増加していたため，CPI が同程度とすれば，性能も 1 割程度低下することが予想される．この要因による性能低下の改善も，提案方式の適用対象範囲外である．

以上により，アクセス制御統合によりベンチマーク性能は最悪で約 40%以上低下するが，本提案の適用により，最良で約 10%の性能低下まで改善することが期待できる．より一般的な条件での性能改善効果は 5.4 節で検討する．

### 5.3　実機測定による性能評価

提案方式の適用により，部分アクセス制御情報でアクセス判定が可能になった場合に，どの程度性能が改善されるかを実機測定により評価する．測定環境は**表 4** のとおりである．

本評価システムで実装した格納方式は S3 である．他の格納方式での実装を行わなかった最大の理由は，他の格納方式ではパーミッション参照更新オペレーションの性能低下を改善しにくいためである．ただし，前述したように他の格納方式でもユーザビリティを犠牲にすれば，この性能低下の影響を排除可能であり，その場合の性能は基本的に本節に示す格納方式 S3 の性能と同様の傾向を示す．

また，通常条件と最悪条件での性能改善効果を見積もるためアクセス制御情報サイズ L は 4 KB，64 KB とした．

測定方法は次のとおりである．NFS 版の SPECsfs2008 ベンチマークを用い，12 KOPS から 18 KOPS までの要求負荷を 1 KOPS 刻みに与え，その最大値を集計する．測定はそれぞれ 3 回実施し，各測定で得られた達成負荷の最大値の平均値を算出する．アクセス制御統合を行った試作システムにおいて，全オペレーションのうち拡張属性にアクセスするオペレーションの割合を 0%から 44%まで 4.4%刻みで変化させて測定を行う．なお本測定条件では，全オペレーションの 56%はアクセス制御処理自体が実行されない．これはアクセス者にかかわらず 28%のオペレーションはアクセス制御処理自体が不要なのに加えて，SPECsfs2008 は所有者権限によりファイルアクセスが行われるため所有者特権によりさらに 28%のオペレーションのアクセス制御処理が省略されるためである．また，比較のため，統合前のシステムについても測定を行う．

**表 4**　測定環境
Table 4　Measuring environment.

| 項目 | 台数 | 仕様 |
|---|---|---|
| サーバ | 1 | CPU: Intel Xeon CPU 5140 2.33GHz (Dual Core) x 2<br>Memory: 10,960,444KB<br>Kernel: Linux 2.6.12.5 + XFS + 提案方式アクセス制御機能<br>OS distribution: SUSE Linux Professional 9.3<br>NFS utils package version: 1.0.7 |
| ディスク | 38 | 108GB/LU (RAID1+0, 2D+2D) x 38 |
| クライアント | 4 | CPU: Intel Xeon CPU 2.80GHz (Dual Core)<br>Memory: 3,632,416KB<br>Kernel: Linux 2.6.26-2-686<br>OS distribution: Debian 5.0.2 |

結果を**図 11** に示す．横軸は図 6 から図 10 の縦軸の表記と合わせるため，拡張属性にアクセスするオペレーションの割合とした．縦軸の値は，3 回測定で得られた達成負荷の最大値の平均値である．3 回測定の最大値，最小値も誤差範囲として示した．拡張属性にアクセスするオペレーションの割合 44%〜0%の範囲の値を表現できる代表的な関数を調べた結果，双曲線関数が一番よく一致した．そこで，本節以降での議論では，100%〜44%の範囲については双曲線近似による推定値を用いる．なお本測定値は，SPEC が指定する測定ルールに厳密に従っているわけではないため，SPEC のサイトに登録，公開されている値とは単純比較はできない．ただし，本測定のデータは同一条件にて測定しているため，相対的な違いは比較可能である．

単にアクセス制御を統合しただけのシステムに，所有者がアクセスする場合，全オペレーションの 70%で拡張属性アクセスが発生する．非所有者の場合は 98%となる．これがアクセス制御統合したシステムでの最悪性能となる．横軸値 98%での達成負荷は，統合前の達成負荷に対し，$L = 4$ KB で 23%，$L = 64$ KB で 44%の性能低下が見込まれる．


© 2010 Information Processing Society of Japan

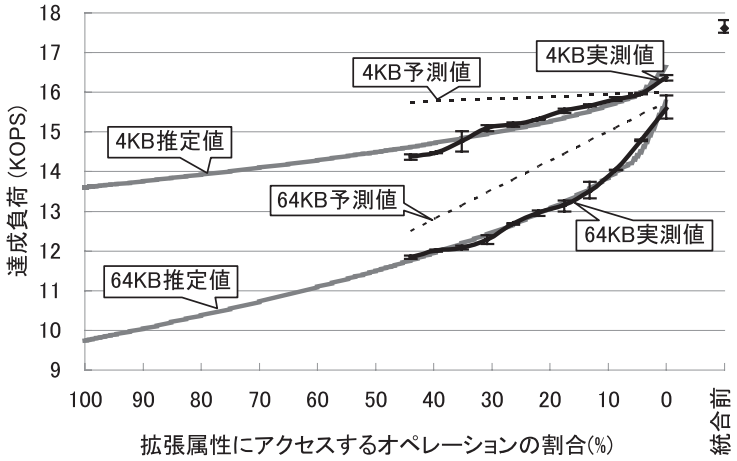


**図 11**　提案方式適用システムによる性能実測結果
Fig. 11　Performance measurement result of proposed method.

これに対し，本提案方式の格納方式 S を適用すれば，拡張属性にアクセスするオペレーションの割合を所有者の場合 44%以下に，非所有者の場合 72%以下に抑えることができる．

さらに，格納方式 S の効果が最大，つまりすべてのアクセスユーザが部分アクセス制御情報の対象ユーザで，すべてのファイルの部分アクセス制御情報の対象アクセス制御情報の設定パターンが代表的設定パターンである場合（横軸値 0%），統合前の最大達成負荷に対する性能低下率は $L = 4\,\mathrm{KB}$ で 7%，$L = 64\,\mathrm{KB}$ で 11%に改善された．

5.1 節の予測値に対する実測値の傾向を $L = 4\,\mathrm{KB}$，$L = 64\,\mathrm{KB}$ のそれぞれで考察する．

$L = 4\,\mathrm{KB}$ の場合，拡張属性にアクセスする割合が小さい領域（< 10%）では予測値と実測値はよく一致しているが，拡張属性にアクセスする割合が大きくなればなるほど，乖離が大きくなる．この理由としては，小さいサイズの拡張属性アクセスにより，単位 I/O あたりの平均データサイズが小さくなることから，ランダムアクセス傾向がより強くなることによる性能低下の拡大が考えられる．

$L = 64\,\mathrm{KB}$ の場合，全体の傾向はある程度一致しているが拡張属性にアクセスする割合が 10%〜30%に乖離が見られる．これも同様にランダムアクセス傾向が強くなることと，キャッシュの影響が考えられる．これは，拡張属性にアクセスする割合が小さい領域では性能の立ち上がりが大きくなる傾向を示していることが根拠となる．予測値と実測値で乖離の見られる部分に関しては，実入出力回数の計測や実データアクセス量などの統計情報を取得分析することにより，今後検討を行う必要がある．

### 5.4　提案方式の具体的事例での性能改善効果

提案方式の性能改善効果を具体的事例で見積もるためには，以下の 4 つの情報が必要となる．

(1)　各ファイルに付与されるアクセス制御情報の平均サイズ
(2)　各ファイルの部分アクセス制御情報対象ユーザ（所有ユーザ，所有グループ，その他ユーザ）のアクセス制御情報が代表的設定パターンと一致する確率
(3)　各ファイルに対する全アクセスリクエストのうち，部分アクセス制御情報非対象ユーザからのアクセスリクエストの割合
(4)　各ファイルに対する全アクセスリクエストのうち，所有者からのアクセスリクエストの割合

このうち (3)，(4) については，実際のアクセスログを解析するのが望ましいが，そのようなデータは一般に取得困難であるため，本論文では次の仮定を行った．(3) については，あるファイルの部分アクセス制御情報対象ユーザの数（そのエントリがあるかどうかに依存するが，最も一般的には所有ユーザと所有グループのメンバ数の合計値）を $\alpha$ とし，部分アクセス制御情報非対象だがそのファイルに対するアクセス権を持つユーザの数を $\beta$ とする．仮に $\alpha$ のユーザと $\beta$ のユーザのファイルに対するアクセス頻度を同一とした場合，(3) の割合は $\beta/(\alpha+\beta)$ として計算される．(4) についても同様の仮定をおいた．(3)，(4) の情報と 5.3 節に記載した各条件での拡張属性アクセス割合から，提案方式適用前後の拡張属性にアクセスするオペレーションの割合が計算可能となる．

これらの情報について，ある組織で運用している約百個のファイル共有を調査したところ，(1)，(2) はどれも同じ傾向だったが，(3)，(4) は大きく 4 つの種類の事例に分類できることが分かった．具体的には，(a) 個人共有フォルダ，(b) 公開共有フォルダ，(c) 組織共有フォルダ，(d) 非組織共有フォルダの 4 種類である．この情報を**表 5** にまとめる．

(1) のアクセス制御情報の平均サイズは，調査した事例の範囲においてはいずれも 4 KB 未満だった．

(2) の代表的設定パターンの一致確率は，調査した事例の範囲においては 100%だった．したがって最適な格納方式は，いずれの事例においても格納方式 S3 となる．ただし (a)，(b) の事例においてはアクセス対象者が限られるため，格納方式 U9 であってもよい．

(3) の非対象ユーザからのアクセス割合は，事例 (a)，(b) では所有者もしくはその他ユー


© 2010 Information Processing Society of Japan

**表 5**　実事例のアクセス制御に関する情報
Table 5　Information about access control on actual cases.

| # | 事例 | (1) アクセス制御情報平均サイズ | (2) 代表的設定パターン確率 | 最適な格納形式 | (3) 非対象ユーザからのアクセス割合 | (4) 所有者からのアクセス割合 | 拡張属性にアクセスするオペレーションの割合 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | 提案方式適用前 | 提案方式適用後 |
| (a) | 個人共有フォルダ | 4KB未満 | 100% | S3, U9 | 0% | 100% | 70% | 0% |
| (b) | 公開共有フォルダ | 4KB未満 | 100% | S3, U9 | 0% | 0.2% | 98% | 3% |
| (c) | 組織共有フォルダ | 4KB未満 | 100% | S3 | 18% | 4% | 97% | 13% |
| (d) | 非組織共有フォルダ | 4KB未満 | 100% | S3 | 76% | 8% | 96% | 55% |

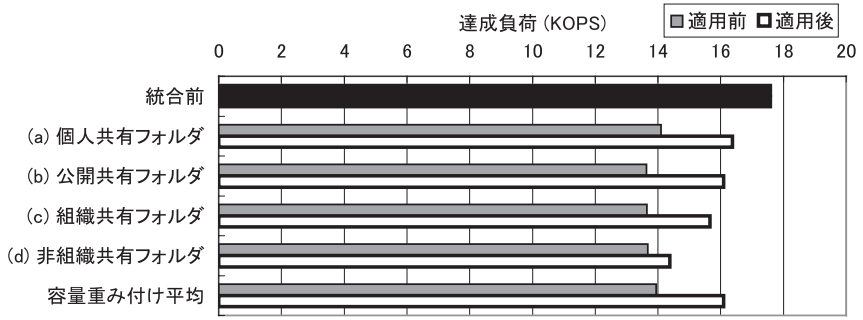


**図 12**　提案方式適用による性能の改善効果
Fig. 12　Performance improvement by proposed method.

ザのエントリが存在するため 0%となる．事例 (c) では適切なグループが設定されているため，18%と比較的低い確率となっている．事例 (d) では組織の構造と連携しない単位の共有のため，適切なグループが設定されていないことにより，76%と高い確率となっている．

(4) の所有者からのアクセス割合は，事例 (a) は 100%となるが，それ以外の事例では 10%未満となる．それぞれの事例における拡張属性にアクセスするオペレーションの割合の変化は表に示すとおりとなる．

図 11 と表 5 のデータより推定した，各事例における統合後の提案方式適用前の性能，提案方式適用後の性能を**図 12** に示す．さらに各事例の容量での重み付けを行った平均の性能もあわせて示す．平均では，提案方式適用前では統合前性能に対して 21%性能が低下するが，提案方式を適用すれば性能低下率を 9%に抑えることができる．

### 5.5　格納方式の選択に対する指針

これまでの議論をふまえ，システム設計者が格納方式をどのように選べばよいかを本節にまとめる．まず，パーミッション参照更新オペレーションのユーザビリティの低下が許容できるかどうかを選択する．

ユーザビリティの低下が許容できない場合は，所有者，所有グループ，その他ユーザ（以降まとめて，パーミッション対象ユーザ）に対してそれぞれ格納方式 S3 を適用するのが望ましい．

ユーザビリティの低下が許容できる場合において，アクセスが想定されるユーザ（もしくはグループ）が限定的である場合，想定アクセスユーザ（もしくはグループ）のアクセス制御情報に対して格納方式 U9 を適用するのが望ましい．アクセスが想定されるユーザが限定的でない，もしくは事前にそのような情報を得ることが難しい場合は，パーミッション対象ユーザで 3 bit ずつ使用するのが望ましい．この場合パーミッション対象ユーザに設定されるアクセスマスクパターンの割合が得られる場合は，その割合に従って S3，A3，U3 の中から選択する．割合を得るのが難しい場合は，S3 を選択するのが望ましい．

## 6. 関 連 研 究

NFS と CIFS の相互接続性に関する研究論文は少ないが，パーミッションと Windows ACL の統合に関する NetApp の実装を示した事例として文献 5)，6) がある．

ファイルシステムにおけるアクセス制御の仕様に関する最近の標準化動向として，NFSv4 ACL がある[9),12)]．NFSv4 ACL は Windows ACL との親和性を強く意識して策定されている．NFSv4 ACL と POSIX ACL とのマッピングに関する検討として，Michigan 大学 CITI による発表[13)] や，標準化を目指した RFC ドラフト[14)] がある．ただし同ドラフトはすでに失効している．

オープンソースコミュニティにおける動向としては，Samba[1)]，NTFS-3G[15)] などがある．これらは基本的に POSIX ACL をベースとして実装されているため，Windows ACL への親和性が低い．

アクセス制御の統合を実現化した製品の実装として，NetApp FAS シリーズ security style[2)]，Microsoft Windows Storage Server[3)]，Sun Solaris 10[4),16),17)]，NEC iStorage NV シリーズ SC-LX[18)]，日立 Essential NAS Platform[19)] などがある．

NetApp の security style は，Windows との親和性重視の ntfs mode，UNIX との親和性重視の unix mode，両方のプロトコルの使用を考慮した mixed mode からなる．アクセ


© 2010 Information Processing Society of Japan

ス制御統合の観点では mixed mode が最も適している．しかし mixed mode では，NFS からパーミッション変更した場合は，すでに設定されている Windows ACL を破棄してしまう．逆に，CIFS から ACL 変更した場合は，すでに設定されているパーミッションを破棄してしまう．つまり NetApp の方式ではセキュリティを犠牲にして，アクセス制御統合時の性能低下を改善していると考えられる．

Microsoft Windows Storage Server は，Windows ACL をネイティブ ACL とする Windows に，UNIX のストレージサービスを統合した製品である．NFS サービスやそのアクセス制御仕様であるパーミッションなどをサポートしている．Windows のネイティブファイルシステムである NTFS のファイル属性情報は，基本的に Master File Table（MFT）と呼ばれる領域のレコードに格納される[21]．Windows ACL は Security Descriptor というファイル属性情報の一部であるため，基本的には MFT レコードに格納される．ただし，全属性情報のサイズの合計値が MFT レコードサイズを超えた場合，一部の属性情報は MFT とは別の領域に格納される．つまり，ACL サイズが大きくなったり，他の属性情報のサイズが大きくなったりすると，本論文で述べた課題と同様の性能低下が発生する．これに対し提案方式は ACL サイズが大きくなった場合にも，性能低下を抑えることができる．

また，Windows Storage Server ではユーザビリティが一部犠牲になっている．たとえば，Windows Storage Server において NFS からファイル作成や “`chmod`” を行うと，その後は CIFS から Windows ACL が更新できない仕様となっている．また，Windows ACL の Everyone エントリの情報が，所有グループとその他ユーザのパーミッション表示に反映される仕様となっているため，所有グループのパーミッション表示が正確ではない．これに対し，提案方式の格納方式 S はユーザビリティを低下させることなく性能改善が可能である．

Sun Solaris 10 は NFSv4 ACL をネイティブ ACL とする ZFS を採用しており，パーミッションや Windows ACL も利用可能である．ZFS では基本属性領域である znode に最大で 6 エントリの ACL が格納可能であるが，7 エントリ以上になると ACL のすべてが基本属性領域以外の領域に格納される[22]．したがって提案方式の ZFS に対する優位性は，Windows Storage Server に対するものと同様である．

NEC SC-LX は Windows ACL の Linux における優れた実装がなされている．しかし，アクセス制御はそれぞれのプロトコルごとで実行する方式となっているため，アクセス制御の一貫性を保ちにくい方式となっている．また ACL キャッシュ機能を実装することにより，本論文で述べた課題にアプローチしているが，ACL キャッシュのためのメモリ領域が，他のファイルメタデータ，ファイルデータを圧迫するため，性能改善効果は限定的と考えられる．

## 7. おわりに

本論文では，CIFS のアクセス制御仕様である Windows ACL を Linux で実装し，アクセス制御を統合する際の課題であるアクセス制御処理の性能低下を改善する，階層型アクセス制御方式を提案した．そして，提案方式で用いる基本属性の部分アクセス制御情報領域に格納する情報について，基本属性の情報のみでアクセス判定可能な確率を評価指標として，高頻度使用アクセスマスクビット，高頻度設定アクセスマスクパターン，アクセスマスクビット論理積の 3 格納方式を比較した．その結果，3 bit の格納領域の場合には高頻度設定アクセスマスクパターン，9 bit の格納領域の場合には高頻度使用アクセスマスクビットの格納方式が最も適していることを明らかにした．提案方式を試作したシステムを用い性能評価を実施したところ，4 事例の容量重み付け平均で，アクセス制御統合前の性能に対し，提案方式適用前は約 21%性能低下していたが，提案方式適用により性能低下率を約 9%に改善できることを明らかにした．

### 商標について

Microsoft，Windows は，米国およびその他の国における米国 Microsoft Corp. の登録商標です．Linux は，Linus Torvalds 氏の日本およびその他の国における登録商標または商標です．UNIX は，X/Open Company Limited が独占的にライセンスしている米国ならびに他の国における登録商標です．Sun，Sun Microsystems，Solaris は，米国 Sun Microsystems, Inc. の米国およびその他の国における商標または登録商標です．NetApp は，米国およびその他の国における米国 NetApp, Inc. の登録商標です．Intel，Xeon は，米国およびその他の国における米国 Intel Corp. の商標です．その他記載されている会社名および商品名は各社の登録商標または商標です．

## 参考文献

1) 高橋基信：Samba のすべて，翔泳社 (2005).
2) Berriman, E.: *NetApp Storage System Multiprotocol Use Guide*, NetApp Technical Report TR-3490 (2006).
3) Windows Services for UNIX (onile). http://technet.microsoft.com/ja-jp/interopmigration/bb380242.aspx (accessed 2009-08-06)
4) Week, L. and Falkner, S.: Solaris NFSv4 ACL Implementation Experience, *Con-*



© 2010 Information Processing Society of Japan

*nectathon Talks 2005*. http://www.connectathon.org/talks05/falkner.pdf (accessed 2009-08-06)
5) Allison, B., Hawley, R., Borr, A., Muhlestein, M. and Hitz, D.: File System Security: Secure Network Data Sharing for NT and UNIX, *Proc. Large Installation System Administration of Windows NT Conference*, p.3 (1998).
6) Hitz, D., Allison, B., Borr, A., Hawley, R. and Muhlestein, M.: Merging NT and UNIX Filesystem Permissions, *Proc. 2nd USENIX Windows NT Symposium*, p.10 (1998).
7) Leung, A., Pasupathy, S., Goodson, G. and Miller, E.: Measurement and Analysis of Large-Scale Network File System Workloads, *Proc. 2008 USENIX Annual Technical Conference*, pp.213–226 (2008).
8) The Official Samba 3.2.x HOWTO and Reference Guide, Chapter 24. Winbind: Use of Domain Accounts (online). http://www.samba.org/samba/docs/man/Samba-HOWTO-Collection/winbind.html (accessed 2009-08-06)
9) RFC3530 Network File System (NFS) version 4 Protocol (online). available from http://tools.ietf.org/html/rfc3530 (accessed 2009-08-06)
10) 長原宏治，佐藤通敏，今井悟志，加藤久慶：ZFS 仮想化されたファイルシステムの徹底活用，アスキー・メディアワークス (2009).
11) SPECsfs2008 User's Guide (online). http://www.spec.org/sfs2008/docs/usersguide.html (accessed 2009-08-06)
12) Network File System (NFS) Version 4 Minor Version 1 Protocol (online). http://tools.ietf.org/html/rfc5661 (accessed 2010-03-12)
13) Fields, J.B.: NFSv4 ACLs: Interoperability and Future Directions, *Connectathon Talks 2007*. http://www.connectathon.org/talks07/nfsv4_acl/ (accessed 2009-08-06)
14) Internet Draft Mapping Between NFSv4 and Posix Draft ACLs (online). http://tools.ietf.org/html/draft-ietf-nfsv4-acl-mapping-05 (accessed 2009-08-06)
15) NTFS-3G community site (online). http://www.ntfs-3g.org/ (accessed 2009-08-06)
16) Week, L. and Falkner, S.: NFSv4 ACLs: Present and Future, *NAS Industry Conference 2005*. http://nasconf.com/pres05/falkner.pdf (accessed 2009-08-06)
17) Falkner, S. and Week, L: NFSv4 ACLs: Where are we going?, *Connectathon Talks 2006*. http://www.connectathon.org/talks06/falkner-week.pdf (accessed 2009-08-06)
18) NEC iStorage NV シリーズ（オンライン）．http://www.nec.co.jp/products/istorage/product/nas/index.shtml（参照 2009-08-06）
19) 鰭崎克巳，檜垣誠一，金井宏樹，川崎　徹：コストパフォーマンスに優れた NAS ゲートウェイ製品「Hitachi Essential NAS Platform」，日立評論，Vol.90, No.3, pp.242–245 (2008).
20) Gruenbacher, A.: POSIX Access Control Lists on Linux, *Proc. USENIX Annual Technical Conference* (*FREENIX Track*), pp.259–272 (2003).
21) NTFS.com (online). http://www.ntfs.com/ (accessed 2010-06-14)
22) ZFS On-Disk Specification draft (online). http://hub.opensolaris.org/bin/view/Community+Group+zfs/docs (accessed 2010-06-14)

(平成 22 年 3 月 16 日受付)
(平成 22 年 9 月 17 日採録)

**中村　隆喜**（正会員）

1973 年生．1996 年 3 月大阪大学工学部精密工学科卒業．1998 年 3 月同大学大学院工学研究科精密科学専攻博士前期課程修了．同年 4 月（株）日立製作所入社．中央研究所勤務．2005 年 10 月より，同社システム開発研究所勤務．2010 年 4 月より，大阪大学大学院情報科学研究科マルチメディア工学専攻博士後期課程在籍．ファイルストレージ，オンラインストレージ，オペレーティングシステムの研究開発に従事．

**亀井　仁志**（正会員）

1980 年生．2002 年 3 月香川大学工学部信頼性情報システム工学科卒業．2004 年 3 月奈良先端科学技術大学院大学情報科学研究科情報システム学専攻博士前期課程修了．同年 4 月（株）日立製作所入社．システム開発研究所勤務．ネットワークストレージ，ファイルシステムの研究開発に従事．

**山本　　彰**（正会員）

1977 年京都大学工学部情報工学科卒業．1979 年同大学大学院修士課程修了．同年日立製作所入社．同年システム開発研究所に入所．性能評価手法，ストレージシステムの研究に従事．2007 年より同社研究開発本部所属．工学博士．



© 2010 Information Processing Society of Japan

**薦田　憲久**

1950 年生．1974 年 3 月大阪大学大学院工学研究科電気工学専攻修士課程修了．同年（株）日立製作所入社．システム開発研究所勤務．1981〜1982 年 UCLA 留学．1991 年 4 月大阪大学工学部情報システム工学科助教授，1992 年 8 月同大学教授．2002 年 4 月より，同大学大学院情報科学研究科マルチメディア工学専攻教授．工学博士．情報システムの計画，評価，電子商取引システム等の研究に従事．電気学会 2000 年度進歩賞等を受賞．IEEE，電気学会の各会員．



© 2010 Information Processing Society of Japan