マルチメディア通信と分散処理 53－12
（１９９２．１．２３）

利用者毎の名前空間を提供する分散ファイルシステム
－NEBULA/DFSファイルシステム－

高野 陽介

日本電気(株) C＆Cシステム研究所

分散ファイルシステムがワークステーションやパソコンなどの広い範囲で利用されるようになり、欠く事のできない計算機環境の１つになっている。このような状況下オフィス分散環境などで問題となるアクセス権制御の弱さを改善するため、現在普及しているファイルのセマンティックスを大きく崩さずに遠隔ファイルアクセスのアクセス権制御を強化・高機能化するファイルサーバを実現した。遠隔ファイルの名前空間をプロトコル上で操作しユーザ毎の名前空間を作りだす方法を用いて実現しており、ユーザの様々な用途に適した名前空間を構成できるという利点も得られる。

User-customizable File Namespaces in a Distributed File System
-NEBULA/DFS File System-

Yousuke TAKANO

C&C Systems Research Laboratories,
NEC Corporation

Resently distributed file systems are used by many people on all classes of computers like workstations or personal computers. We develop a add-on system which enhances the access control facility of conventional distributed file systems. To make access control more flexible, this system has an user-customizable namespace approach. This paper describes the name management model of this system and the implementation of a prototype system.

1 はじめに

分散ファイルシステムがワークステーションやパソコンなどの広い範囲で利用されるようになり、欠く事のできない計算機環境の1つになっている。さらに、ファイルサーバなどに見られるように、遠隔ファイルはもはやローカルファイルのオプションでなく、ユーザの主たるリソースに位置付けられつつある。

このような状況下において、オフィス分散環境のような多様なユーザと多様な権限が存在する分散環境ではより木目細かで柔軟なアクセス権制御が要求されている。これに対し、遠隔ファイルをアクセスする際に検査されるアクセス権はローカルファイルの延長あるいはローカルファイルよりも単に権限を狭めただけのものになっているため、このような目的を満足する柔軟なアクセス権制御を行なうことが難しい。

このような点を解決するために、遠隔ファイルの名前空間を操作する方法を用いてユーザ毎の名前空間を作りだすことにより、現在普及しているファイルのセマンティックスを大きく崩さずに遠隔ファイルのアクセス権制御を強化・高機能化したファイルサーバ**NEBULA/DFS**の実現を進めている。

NEBULA/DFSは**NEBULAオペレーティングシステム**[5]の実現の1コンポーネントとして構築したもので、ユーザ毎の名前空間のデータベースを管理するdishネームサーバとファイルアクセスのプロトコルを直接操作するプロトコルフィルタによって構成される。本稿では、NEBULA／DFSが実現するユーザごとの名前管理のモデルとその実装について述べる。

2 システムの狙い

本節では、NEBULA/DFSが解決しようとする3つの課題について述べる。

2.1 利用者毎のファイル空間の提供

ファイル名前空間の中には、ストレージ管理の都合、アプリケーションシステムの都合、システム管理者の都合、多数の一般ユーザの都合などが混在するため、その各々にとって必ずしも良い使い勝手が得られていない傾向にある。ディレクトリの階層化やシンボリックリンク等を用いることで問題のいくつかは解決するが、多用すれば名前空間の構造は逆に複雑化するだけで管理しにくいものになってしまう。そこで、NEBULA/DFSではユーザ毎に自由に構成することができる名前空間を供給し、ユーザはその名前空間を資源を利用するための窓として利用することにより、様々な利用の都合をネットワークシステム上に混在させても、互いの使い勝手を損なわないようにする。

一方、システム上のすべてのファイル環境にユーザ毎の名前空間を導入することはOSカーネル層あるいは実行時ライブラリ層の改造が必要になりアプリケーションの互換性等の問題を生ずる。そこで、NEBULA/DFSではサーバクライアント型のファイル利用を主な対象に考え、遠隔サイトからネットワーク経由で供給されるファイル空間に対してのみ各ユーザの名前空間への分解を行うことにした。この場合、操作すべきなのはファイルアクセスを伝達する通信メッセージであるので、互換性に問題がないソフトウェア階層を通った所で必要な処理を実施することができる。

名前空間のカスタマイズの研究はこれまで、ファイル空間における複数コンテキストの実現、コンテキストのユーザ・タスクなどへの結合という方向で進められている。従来の研究として、Prospero[3], Plan9[4], Tilda[2], QuickSilver[1]がある。本稿で述べる研究では、このような技術をさらに遠隔ファイルのアクセス権制御に応用している。

2.2 アクセス権のより詳細な制御

ファイル供給の単位であるサブツリーは、NFS[1][6]の場合、それ全体を単位としてアクセス権の調整

[1]NFSはSUNマイクロシステムズの登録商標である。

が可能で、例えば書き込み禁止などに設定することが可能である。サブツリーの中の1つのファイルのアクセス権を変更する場合には、サーバサイト中のファイルのアクセス権そのものを修正する必要がある。より多様なアクセス権操作が可能な実装例も存在するがサブツリー全体に対する操作に限られている。

これに対し、オフィス等の分散環境では多様なユーザと多様な権限が存在するため、より詳細なファイル単位・ユーザ単位のアクセス権設定・制御が要求される。本システムでは、ユーザ毎のファイル空間を応用してこのようなアクセス権制御を実現する。すなわち、名前を仮想化するのに加え、名前に付随するアクセス権などの属性情報も仮想化できるようにする。同一のファイルを指す名前であっても、別々の名前空間上にある場合そのアクセス権を変えておけば、同じファイルを指す名前であっても異なるユーザには異なる権限でそれをアクセスさせるようにすることができる。

その一方でユーザ毎の名前空間の内部に注目すると、そこに新たなアクセス権の概念が追加されたわけではないので、ユーザから見たときのファイルのセマンティックスは変えずに済むことになる。

2.3 遠隔ファイル管理の集中化

複数のサイト間でやりとりされるリモートファイルの情報を論理的に1つのデータベースで管理し、その扱いを統合的に行えるようにする。NFSの場合、リモートサイトにエクスポートする、およびリモートサイトからインポートするなど、制御や管理情報は各サイト上でそれぞれ実施するようになっている。この体系を整理してサイト間の、また特に本システムでは対ユーザの関係構造を中央で保存し操作できるようにする。

この機能のもう1つの意図は、各ホスト内に他のホストに深く依存した情報を置かないようにすることである。NFSの場合、サーバサイトとクライアントサイトの間に様々な組み合わせの関係が存在し、それを各ホストの管理データベース中に格納し管理する必要がある。NEBULA/DFSはこのようなサーバとクライアントの間を仲介するエージェントとして動作することにより管理すべき関係の数を削減する。

3 名前管理のモデル

NEBULA/DFSでは、サーバサイトからエクスポートされたサブツリー、すなわちファイルの名前空間がユーザ毎の名前空間に分解・合成されクライアントサイトにインポートされるまでの加工の過程を表現するための名前管理のモデルを定義する。このモデルでは、加工の過程にあるサブツリーを4つの種類の名前空間に分類して表現する。そのうち1つはクライアントには直接見えずNEBULA/DFSの管理者が操作する制御上の名前空間で、のこりの3つはクライアントが意識して操作する必要のある名前空間である。本節では、この4つの名前空間について述べる。NEBULA/DFSにおける課題のうち、遠隔ファイル管理の集中化とアクセス権のより詳細な制御は以下で述べる管理名前空間と公開名前空間により、また利用者毎のファイル空間の提供は同じく私有名前空間と共有名前空間により実現する。

3.1 制御上の名前空間

3.1.1 管理名前空間

サーバサイトがエクスポートしたサブツリーをそのままNEBULA/DFSの中で保持する名前空間を管理名前空間と定義する。エクスポートされてきた各サブツリーがスーパールート形式で連結された構成で表現してある。遠隔ファイルアクセスの管理を行うためにNEBULA/DFSの管理者のみが管理名前空間を操作することができる。管理者はこの管理名前空間から次に述べる公開名前空間へ加工する過程でアクセス権の調整を行う。

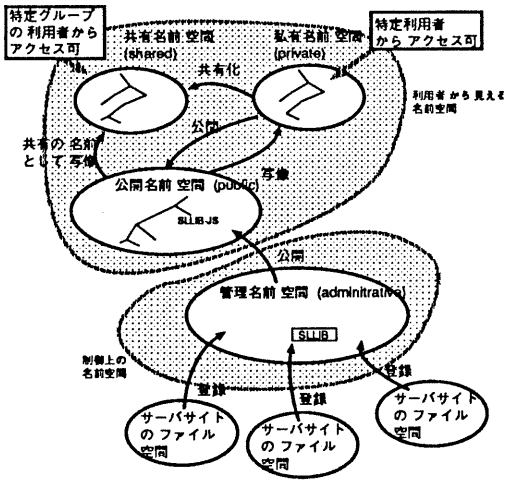

図 1: 仮想的な名前空間の実現

3.2 一般ユーザから見える名前空間

3.2.1 公開名前空間

管理名前空間を加工しアクセス権などの修正を加えた結果として生成される空間を公開名前空間と定義する。この名前空間は、ファイル名前空間が一般ユーザにわたる前にシステム管理者の手で名前空間とその上のアクセス権の状態を個々のユーザに適した形式に修正するために存在する。したがって、公開名前空間の修正はシステム管理者のみが行い、一般ユーザは名前空間を一部の操作を除き変更することはできない。一般ユーザによる名前空間の操作は、後述する私有名前空間の上に場所を移して行う。

管理名前空間からの修正では、以下の操作を行うことができる。これらは、通常の UNIX システムコールを通してでなく、dish ネームサーバが提供する名前管理用のインターフェイスを通して行う。

(1) 新規名前空間の生成・ファイル名の写像

管理名前空間上の名前を写像する空の名前空間を生成する。管理名前空間からファイルの名前をファイル単位で、あるいはサブツリー単位で写像することにより、公開名前空間の基本的な構成を作成する。写像後別の名前に付け変えることもできる。この意味では、公開名前空間は巨大なシンボリックリンクの集合体と見なすこともできる。但し、公開名前空間の場合はシンボリックリンク特有の制限は無くユーザには通常のファイルのように認識される点や、後述するアクセス権リストなど各種属性情報の変更が行なわれる点などが異なる。

公開名前空間にはそれを参照可能なユーザ (あるいはユーザグループ) のリストを定義することができる。当然、全てのユーザから参照可能な公開名前空間を生成することもできる。本システムの管理者以外にはそこに定義されたユーザのみが公開名前空間を経由してファイルアクセス、あるいは後述する私有名前空間への再写像を行うことができる。

(2) アクセス権リストの変更

サーバサイトが指定しているアクセス権の範囲で修正を行うことができる。すなわち、アクセス権を減らす方向のみで変更できる。

(3) ファイル名の所有者の変更

公開名前空間上でのファイル名の所有者を変更する。この変更に伴ってアクセス権リストの所有者、所有グループ、その他ユーザの並びも元のアクセス権と矛盾のない位置にシフトする。例えば、所有グループの一員のユーザを新しい所有者にした場合、所有グループに提供されていたアクセス権リストで所有者のアクセス権リストを書き換える。

(4) ファイル名の抹消

公開名前空間中の名前を抹消する。これによるサーバサイト上のファイル実体の消去は行わない。サーバサイトからエクスポートされたサブツリーの中に有用なファイルに紛れてユーザにとって不要なファイルが含まれる場合がある。その反面、サーバサイトでは同じサブツリーから何からの事情でこのようなファイルを移動することができない場合、

この機能を利用して、公開名前空間上から不要な名前を見えなくすることができる。これにより、ユーザが認識すべき名前空間の大きさを最適化してユーザにとって理解しやすい構造を供給することができる。

あるいは、サブツリーを複数の公開名前空間を通して公開する場合、その一方でファイル名の抹消を行うことにより、あるユーザには一部だけ公開しないという操作を行うことができる。

(5) ファイル名の複製

ファイル名の写像のバリエーションとして、同じサーバ上のファイルを指す名前を公開名前空間の内部の階層上の異なる場所や複数の公開名前空間上に生成することができる。これとアクセス権リストの変更、および、公開名前空間を利用できるユーザリストの操作を組み合わせることにより、同一ファイルを異なるユーザに対して異なるアクセス権で提示することができる。

3.2.2 私有名前空間

複数のユーザが共通に利用する公開名前空間に対して、特定の1ユーザが独占的に利用できる名前空間を私有名前空間と定義する。この名前空間では、システム管理者が管理者の立場に立って権限をしぼり込み、またユーザの立場に立ってある程度使い勝手を向上させた公開名前空間を元にして、個々のユーザが自分の手でさらに自分専用に使いやすく名前空間を改造できるようにする機能を提供する。

利用者は公開名前空間上の名前を指定し、それを私有名前空間上の別の名前に写像することにより、以後自分だけの名前空間から名前の指す資源にアクセスできるようになる。すなわち、特定のユーザが使いたいファイルだけをそのユーザにとって使いやすい構造で配置することができる。但し、写像元の公開名前空間はシステム管理者によりファイルのバリエーション、アクセス権とも制限されているので、私有名前空間にもその制限は受け継がれる。公開名前空間から私有名前空間への名前の写像では、先に述べた管理空間から公開空間への写像で行える各種属性操作とほぼ同じ操作を行うことができる。

私有名前空間は1人のユーザが複数個持つことができる。プログラミングのための空間、情報蓄積のための空間など仕事の種類に応じて使い分けることができる。

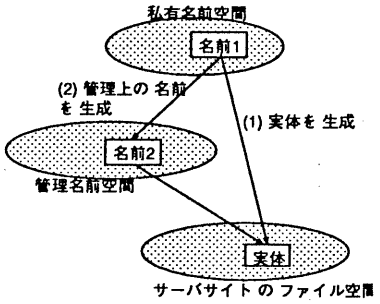

図 2: 私有名前空間上のファイル生成

私有名前空間上に利用者が新たな資源とその名前を生成した場合、図2のように管理名前空間上に同じ資源をポイントする別の名前が自動的に生成される。一方、この変更は公開名前空間には反映されない。このように、利用者が私有名前空間上に個人的に生成した資源は、その利用者がその資源を明示的に公開名前空間に写像(公開)しない限り他の一般利用者が参照することはできない。

私有空間を特徴づけている2つの属性について説明する。

(1) 利用者の属性

本名前管理モデルでは、すべての私有名前空間に空間自体の所有者を定義している。すなわち、私有空間はその所有者からの要求に対してのみ可視になる。利用者は自分の利用環境を自由にレイアウトできる権利を与えられたことになる。

(2) レポジトリ・サイトの属性

ユーザが私有名前空間上にファイルを新規作成した場合にファイルの実体を格納するためのディスク

領域が割り当てられるサーバサイトをレポジトリサイトを呼ぶことにする。私有名前空間上には異なるサーバサイトから写像されたファイル名が存在することができるため、新規生成されたファイルが置かれるサイトが一意に決まらない。そこで、本名前管理モデルでは、特定の私有名前空間でファイルを生成した時にはファイルはあらかじめ定義されたサーバサイト上に割り当てられるという概念を導入する。これはユーザが自分のデータを置くためにサーバサイトとの間で契約を行うという考えを本システムでは導入していることを意味している。

3.2.3 共有名前空間

特定の利用者グループに属する複数の利用者により共有される名前空間である。共有の範囲が特定の利用者グループに限定される点、および、私有名前空間が持っている2つの属性を備える点が公開名前空間と異なる。

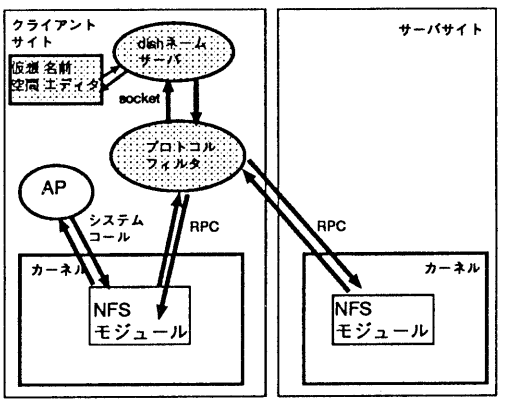

図 3: システムの実装

4 試作システム

NEBULA/DFS は NFS Ver.2 のプロトコルに対応し、EWS4800 および SparcStation[2] 上で試作している。本システムは図3に示すように2つのデーモンプロセス、dish ネームサーバとプロトコルフィルタによって構成される。二つのサーバに分離しているのは、dish ネームサーバがユーザ毎の名前空間を実現する一方で、ファイルの名前に依存しない汎用の名前管理を目的に設計したサーバであるからである。この2つのサーバ以外に NFS のセマンティックスに含まれない名前空間操作を行うためのインターフェイスとして仮想名前空間エディタというツールを作成した。

4.1 dish ネームサーバ

dish ネームサーバは、各種名前空間の構造、名前と名前に付属する属性情報を保持するデータベースを管理し、各名前空間の特徴を実現するための手続き群を実行する。対外的には名前空間操作のための機能を仮想名前空間エディタなどのツールに、名前解釈や属性取得・変更のための機能をプロトコルフィルタにそれぞれ提供する。

具体的な処理例として、名前解釈の場合の処理の手順を説明する。まず、名前空間の識別子と要求したユーザのクリデンシャル (credential: ユーザ ID やユーザグループ ID の一覧を含むデータ構造) を受け取り、それを元に名前空間がユーザに対して可視であるかどうかを決定する。私有名前空間ならば、クリデンシャルに含まれる要求者のユーザ ID と私有名前空間の利用者の属性として登録されているユーザ ID が一致する場合に可視となる。可視であるならば、指定された名前空間内で名前解釈を実施し、検索された名前のファイルハンドル[3]を求める。可視でないならばエラーを発生する。本システムではクリデンシャルを使うために NEBULA/DFS を利用するサイト間ではユーザ／グループの識別子の体系が何らかの方法で統一されていることを仮定している。

4.2 プロトコルフィルタ

プロトコルフィルタはいわゆる NFS サーバとし

[2] SparcStation は SUN マイクロシステムズの登録商標である。

[3] ファイルハンドルは NFS プロトコルにおいて遠隔ファイル識別子として用いられる情報である。

て実現している。NFSファイルへのシステムコールを受け付けたUNIXカーネルから直接RPCによるNFSプロトコルのメッセージを受け付ける。そこで、名前や属性に対する操作が必要な際にはdishネームサーバと通信を行う。その結果に基づいて、拒否される要求であるならば、その旨をRPCに対して返却し、そうでないならばNFSプロトコルのメッセージを修正し、ファイルの実体をサービスしているサーバサイトにRPCを再転送する。

ファイルの実体を参照しなくても完了できる操作の場合には、このサーバ内でRPCに対する返答を作成し返却する。このような場合にはネットワーク上の遅延が省略されるので、本システムによるオーバヘッドとある程度相殺されることが期待できる。

例として、名前検索の要求が到着した場合の処理を説明する。この要求に対して、dishネームサーバに名前空間の識別子、ファイル名、およびRPCのヘッダに含まれるクリデンシャルを転送し名前検索を要求する。アクセス権検査を含む検索処理が無事完了した場合、dishネームサーバは名前空間中の検索されたファイル名に対するファイルハンドルと名前空間中でファイル名に付随していた属性情報(アクセス権、ファイルタイプ、所有者などの情報)をプロトコルフィルタに返却する。名前検索の場合、これで処理は完了するのでこの結果をRPCの返却値としてカーネルに返却する。

この後、このファイルハンドルを使用してファイルへの読み書きが要求された時には、ファイル実体のサーバサイト上での名前に対するファイルハンドルが計算される。プロトコルフィルタはこのファイルハンドルで要求メッセージ中のファイルハンドルを書き換え、要求メッセージをサーバサイトに再転送する。サーバサイトからの返答は再びNFSコンパチブルサーバで受け取られ、カーネルに再送される。

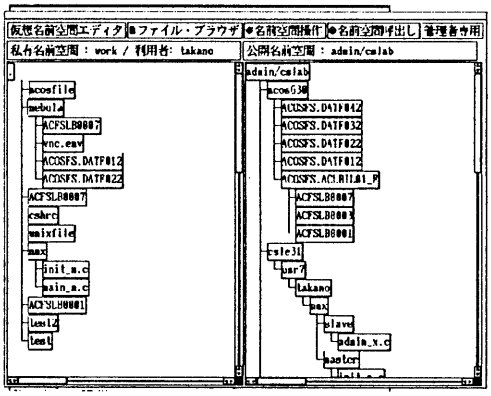

図4: 仮想名前空間エディタの表示例

4.3 仮想名前空間エディタ

各種名前空間に対する操作は、dishネームサーバの外側にXウインドウと鼎[4]を利用した仮想名前空間エディタを開発して行なえるようにした。このツールはdishネームサーバのサービスポートに通信を行い、ユーザが要求した名前空間への操作を名前データベース上に反映させる。図4に仮想名前空間エディタの表示例を示す。図の右側のウィンドウがadmin/cslabという名前の公開名前空間を、左側のウィンドウがtakanoというユーザの所有するworkという名前の私有名前空間を、それぞれ木構造の形式でその一覧を表示したものである。

このツールでは、3.2.1節で述べた名前空間の諸操作を行うことができる。例えば、公開名前空間上に表示された名前をマウスでクリックし、その名前を写像したい私有名前空間の上で再びマウスをクリックすることで、公開名前空間上の名前が指すファイルを参照する名前が私有名前空間上に生成できる。このほか、管理者用の名前空間操作ツールも現在このツール中にパスワードで保護された形で組み込まれている。

このようにして作成された公開・私有・共有名前空間は、NFS Ver2インターフェイスを実装するOS上でリモートマウントを行うことにより利用

[4]鼎はXウインドウ上のツールキットである。

を開始する。リモートマウントされるリモートノード、すなわち各名前空間内のルートディレクトリを表すファイルハンドルはdishネームサーバから取り寄せる必要があるので、マウントのためのコマンドは既存のものでなく、新たに作成したnsmountコマンドを使用する。

一方、サーバサイト側では、NEBULA/DFSにファイルをエクスポートするために新たに作成したexportnsコマンドを使用する。このコマンドはエクスポートするサブツリーの最上位ディレクトリのファイルハンドルをdishネームサーバに登録する。通常のNFSの操作と基本的に異なる点はこのマウントとエクスポートの2点だけである。

4.4 課題

今回の実装で解決できなかった問題点として3つのことが挙げられる。第一点は、名前のガベージコレクションの問題である。1つのファイル実体を引用する名前が複数の名前空間の箇所に存在するのでファイルの生滅に同期した名前の一貫性の維持が必要になる。しかしながら、この一貫性の維持は本システムにとって大きなオーバヘッドとなることが予想できる。現時点では、シンボリックリンクのように指すべき実体が存在しない名前を許すようにして処理してしまっている。

第二点は、性能の問題である。複数の名前空間を導入することにより、目的のファイル実体を見つけるために本来一回で済むはずの名前解釈が数回必要になる。適切な名前解釈のキャッシュや先行解釈などを導入する方法を検討する必要がある。

第三点は名前空間の特定のユーザへの可視化がクリデンシャルを用いて実装されているため、セキュリティレベルが低いことである。この問題を解決するにはより厳密な認証システムとの連係を考慮する必要がある。

5 おわりに

分散ファイルシステムのプロトコルを操作することにより、ユーザ毎の名前空間を生成し、それによって遠隔ファイルアクセスに適用されるアクセス権制御を強化・高機能化するシステムの設計モデルとその実現について述べた。今後の課題は前節で述べた問題点を解決し、本システムを実用化可能なレベルに引きあげることである。

謝辞

本研究の機会を与えていただくと共に有益なご指導・ご助言をいただいたC&Cシステム研究所の山本所長、久保主管研究員、コンピュータシステム研究部の小池部長、川越課長、新主任、岩崎氏をはじめとする方々にこの場をかりて深く感謝致します。

参考文献

[1] L.F.Cabrera and J.Wyllie: QuickSilver Distributed File Services: An Architecture for Horizontal Growth, Proceedings of 2nd IEEE Conference on Computer Workstations, pp.23–37, 1988.

[2] D.Comer and T.P.Murtagh: The Tilde File Naming Scheme, Proceedings of the International Conference on Distributed Computing Systems, pp.509–514, 1986.

[3] B.C.Neuman: The Need of Closure in Large Distributed Systems, Operating Systems Review, pp.28–30, Oct. 1989.

[4] R.Pike, D.Presotto, K.Thompson, et al.: Plan 9 from Bell Labs, Technical Report, 1990.

[5] 高野, 久保: 異種OS間連携を支援する拡張名前管理, 情処学会OS研究会研究報告,48-10,1990.

[6] D.Walsh, B.Lyon, et al.: Overview of the SUN Network Filesystem. Proceedings of USENIX Conference, Winter, 1985.