

単一アドレス空間におけるオブジェクトの きめの細かい保護機構の設計

5 F-3

繁田 聡一 谷森 徹 清水 謙多郎 芦原 評
電気通信大学 情報工学科

1 はじめに

オペレーティングシステムを構造化して設計するには、きめの細かい柔軟な保護機構が不可欠である。本稿では、受動オブジェクトをベースとするオペレーティングシステムにおいて、オブジェクトの柔軟な保護を実現する機構を提案する。提案方式は、次のような特徴を持つ。

- きめの細かい保護：スレッドごとにキーを持ち、ACL (Access Control List) には、メソッド単位の設定が可能である。
- 柔軟なアクセス権の設定：複数のキーを組み合わせ、オブジェクトへのアクセスを許可する条件を柔軟に設定することができる。
- サブジェクト側の制御：サブジェクトの側で、アクセスできるオブジェクトを限定するための SCL (Subject Control List) 機構を導入した。

柔軟性を目指した保護機構として、Conditional Capability [1]、Password Capability があるが、これらはケイパビリティにアクセス権を付加するだけのものであり、提案方式のような柔軟性はない。Mungi [5] は、Password Capability を拡張し、ケイパビリティに階層を持たせているが、アクセス条件の設定は制限される。CACL [3] は、メソッド単位での保護設定を許し、言語処理系の支援により効率化を図っているが、提案方式のように、メソッド呼び出しを許す条件を柔軟に設定することはできない。

2 オブジェクトモデル

本研究では、細粒度オブジェクトにより構成されたオブジェクト/スレッド モデルのオペレーティングシ

テム [2, 4] を想定している。オブジェクトとは、データとメソッドをカプセル化したものであり、スレッドとはプログラムコードの実行実体である。複数のオブジェクトが同一のアドレス空間上にアクティベートされ、それらのオブジェクト上を複数のスレッドが走行する。

同一のアドレス空間上に複数のオブジェクトがアクティベートされるため、アドレス空間を切替えることなく、高速なオブジェクト間通信が実現される。一方、オブジェクトは同一のアドレス空間上で、不当なアクセスから互いに保護されなければならない [6]。

3 基本設計

3.1 キー

キーは、ユーザ、スレッド、オブジェクトに対して定義される。キーには、次の5種類がある。

- ユーザキー
ユーザ単位のアクセス権を規定するために用いる。
- 所有者キー
各々のスレッド及びオブジェクトは、所有者キーを持つ。所有者キーには、ユーザキーが設定される。
- クラスキー
特定のクラスのオブジェクトにアクセス権を与えるために用いる。あるクラスから生成されたオブジェクトは、そのクラスのクラスキーを継承する。
- オブジェクトキー
オブジェクトの個々のインスタンスにアクセス権を与えるために用いる。
- ドメインキー
ユーザが任意のオブジェクトのインスタンスの集合に同じアクセス権を設定したい場合に用いる。ユーザが必要に応じて明示的に生成することができるのは、ドメインキーのみである。

3.2 オブジェクト呼び出しの機構

スレッドは、サブジェクトとしての権限を表すキーのリスト (スレッドキーリスト) を持つ。スレッドが生成

“Fine-grained Protection Mechanism for Single Address Space, Object-based Operating Systems”
Soichi Shigeta, Toru Tanimori, Kentaro Shimizu,
and Hyo Ashihara
Department of Computer Science,
The University of Electro-Communications
1-5-1, Chofugaoka, Chofu-shi, Tokyo, Japan.

された時点では、所有者キー（スレッドの所有者のユーザキー）だけが設定されている。オブジェクトは、そのオブジェクトを実行しているスレッドに与える権限を表すキーのリスト（オブジェクトキーリスト）を持つ。オブジェクトキーリストは、所有者キー、クラスキー、オブジェクトキーと0個以上のドメインキーを保持する。

各オブジェクトは、3.3節で述べる ACL (Access Control List) を持つ。スレッドがオブジェクト（のメソッド）を呼び出すとき、システムはそのオブジェクトの ACL を調べ、スレッドがキーリストに保持しているキーとマッチするエントリを探索し、そのエントリに実行しようとするメソッドが記載されていれば、メソッドの呼び出しを許す。このとき、そのオブジェクトの持つキーがスレッドキーリストに追加され、呼び出しから復帰すると、それらのキーはスレッドキーリストから削除される。

3.3 Access Control List (ACL)

ACL は、どのサブジェクトにどのメソッド呼出しを許すかを規定する。各オブジェクトは ACL を持ち、所有者だけが設定を変更できる。ACL の各エントリは、キーの集合、キーの間の論理演算の指定 (AND, OR)、メソッドの集合、メソッドの実行に関する指定 (GRANT, DENY) から構成される。

ACL に、複数のキー、および、サブジェクトが提示するキーとこれら複数のキーとのマッチング条件を柔軟に指定できる点が、提案方式の重要な特徴である。例として、ファイルオブジェクトから *FIFO* 複合オブジェクトを構成する場合を考える。ユーザ *foo* がクラスキー *K_{FIFO}* を持っている *FIFO* オブジェクトを作成する時、その複合オブジェクトであるファイルの ACL に $\{K_{foo}, K_{FIFO}\}$, AND, $\{\text{read}, \text{write}\}$, GRANT > というエントリを設定すると、*foo* のスレッドが *FIFO* オブジェクトを操作する時にだけこのファイルを read/write できることになる。

3.4 Subject Control List (SCL)

SCL は、サブジェクトの側で、呼び出すことのできるオブジェクトを限定する。各々のサブジェクトは SCL を持つことができる。各エントリは、オブジェクト名、(呼出し可能な) メソッドのリストから構成される。実行時に、SCL に記載されていないメソッドの呼び出しは却下される。挙動の不明な信頼できないコード（トロイの木馬が潜んでいる恐れのあるコード）を実行する時

にユーザが明示的に SCL を設定する。サブジェクト自身が SCL を参照、変更することはできない。

4 実装

筆者らは、現在、細粒度オブジェクトによって構成された オブジェクト/スレッド モデルに基づくオペレーティングシステムを開発しており、その保護機構として提案方式を実装している。

各スレッドは、アドレス空間とスレッドの実行状態、スレッドキーリスト、スレッドが使用しているメソッドのアドレスを保持する MAT (Method Activation Table) を持つ。最初にオブジェクトを呼び出す場合は、トラップが発生してカーネルに制御が移り、提案方式による保護のチェックが行なわれる。アクセスが許されると、MAT のエントリにメソッドのコードのアドレスがセットされ、以後は MAT を用いたユーザレベルの間接呼び出しとしてメソッドが呼び出される。アクセス権が変更された時は、そのオブジェクトの MAT のエントリが無効化される。

5 まとめ

キー/ロック 方式を独自に拡張した、きめの細かい柔軟なオブジェクトの保護機構を設計するとともに、オブジェクト/スレッド モデルに基づくオペレーティングシステムに適用し、その効率的な実現について検討した。

参考文献

- [1] K. Ekanadham and A. J. Bernstein, "Conditional Capabilities", *IEEE Transactions of Software Engineering*, SE-5, 5, pp.458-464 (1979).
- [2] P. Dasgupta, R. J. LeBlanc, Jr., M. Ahamad, and U. Ramachandran, "The Clouds distributed operating system", *IEEE Computer*, 24, 11, pp.34-44 (1991).
- [3] J. Richardson, P. Schwarz, and L. Cabrera, "CACL: Efficient Fine-Grained Protection for Objects", *Proceedings of OOPSLA '92*, pp.263-275 (1992).
- [4] G. Hamilton and P. Kougiouris, "The Spring Nucleus", *Proceedings of Summer USENIX Technical Conference*, pp.147-159 (1993).
- [5] J. Vochtelloo, S. Russel, and G. Heiser, "Capability-Based Protection in the Mungi Operating System", *Proceedings of Object Orientation in Operating Systems*, pp.108-115 (1993).
- [6] J. S. Chase, H. M. Levy, M. J. Feeley, and E. D. Lazowska, "Sharing and Protection in a Single Address Space Operating System", *ACM Transactions on Computer Systems*, 12, 4, pp.271-307 (1994).