2N-3

協調関係に基づいたアクセス制御を提供する拡張キー/ロック方式に関する一考察

繁田 聡一 † 清水 謙多郎 ‡ 曽和 将容 †

† 電気通信大学 情報システム学研究科, ‡ 東京大学 農学生命科学研究科

1 はじめに

互いに協調する細粒度なモジュールの集合として構成すること [1, 4] が, 拡張可能なオペレーティングシステムを実現する一般的な手法である. 筆者らは, 小さなナノカーネルと協調する細粒度モジュールで構成される, 単一アドレス空間方式の動的に拡張可能なオペレーティングシステムを開発している. ナノカーネルは, スレッド管理やアクセス制御, モジュールの動的バインディングなどの基本機能のみを提供する. CPU スケジューリングや仮想ページ管理といったサービスは, ナノカーネル外のモジュールとして実装される. 1 つのアドレス空間に動的にリンク, ローディングされたモジュールを, 不正なアクセスから保護する機構が盛んに研究されている [2, 3]. 筆者らが開発した拡張キー/ロック方式 [5] は, モジュール間の多様な協調関係に対応した柔軟なアクセス制御を提供できる. 本稿では, 開発中のオペレーティングシステムのアクセス制御機構に拡張キー/ロック方式を用いることについて考察する.

2 拡張キー/ロック方式

アクセスを行なう主体をサブジェクト, アクセスされる対象をオブジェクトと呼ぶ. 筆者らは, 従来のキー/ロック方式を拡張し, オブジェクト間の協調関係を記述する機能とサブジェクトのアクセス履歴を保持する機能を追加した. 拡張キー/ロック方式は, サブジェクトが特定の「アクセスルート」(協調関係にあるオブジェクト同士の一連の呼出し系列) を経ているか否かでオブジェクトへのアクセスを制御する「アクセスルートコントロール」を提供する.

3 拡張キー/ロック方式の適用

開発中のオペレーティングシステムのアクセス制御機構に拡張キー/ロック方式を適用する. 唯一の実行主体であるスレッドがサブジェクトに, 単一アドレス空間上のセグメント (連続ページ領域) がオブジェクトに対応する. ユーザやスレッド, セグメントなどの全ての識別子はキーとして表現される. また, ソフトウェア制御のタグ付き TLB (translation look-aside buffer) を備えたプロセッサ上での実装を前提としている. 拡張キー/ロック方式によるアクセス権チェックは, TLB ミス時にのみ行われる. TLB の各エントリはスレッドの識別子でタグ付けされるため, スレッド切替え時に TLB 全体をフラッシュする必要はない.

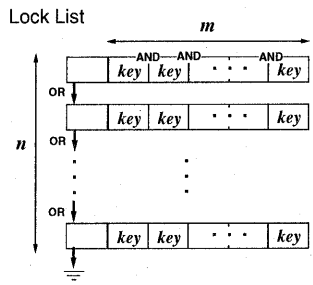

図 1: ロックリスト

4 性能評価

拡張キー/ロック方式は, アクセスルートコントロールを提供するが, アクセス権チェックの際にキーとロックのマッチングを調べるコストが大きいという欠点がある. そこで, そのコストを調べた. 開発中のオペレーティングシステムは, 現在のところ実装が完了していないため, シミュレーションによる見積もりを行った.

各スレッド *thr* にはキーリスト KL(*thr*) が, 各セグメント *seg* にはロックリスト LL(*seg*) が結びつけられている. KL(*thr*) は配列をキースタックとして使用するように実現した. 一方, LL(*seg*) は**図 1** のように, 配列で実現されたロックリストエントリの線形リストとして実装した. 1 つのロックリストエントリ内に指定されているキー間に対しては論理演算 AND が適用され, ロックリストエントリ間には論理演算 OR が適用される. すなわち, LL(*seg*) はキーの論理式の積和標準形でアクセス許可条件を保持する.

KL(*thr*) 中のキーの数をキーリストの長さ l, 各ロックリストエントリに指定されたキーの論理式のリテラル数をエントリの長さ m, ロックリストに登録したロックリストエントリ数をロックリストの長さ n とする (図 1 参照). さらに, $m \times n$ をロックリストのサイズとする. サイズ $m \times n$ のロックリストのマッチングに要するキーとロックのマッチング回数は, 最悪 $\{m(m+1)/2\}n$

"A Study of Access Control for Cooperative Objects Provided by the Extended Key/Lock Scheme"

Soichi Shigeta†, Kentaro Shimizu‡, and Masahiro Sowa†

†The Graduate School of Information Systems, University of Electro-Communications
1-5-1, Chofugaoka, Chofu-shi, Tokyo 182–8585, Japan.

‡The Graduate School of Agricultural and Life Sciences, University of Tokyo
1-1-1, Yayoi, Bunkyo-ku, Tokyo 113–8657, Japan.

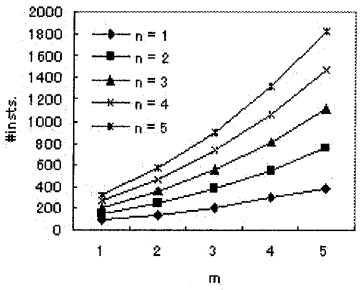

図 2: n を固定した場合の命令数の増加傾向

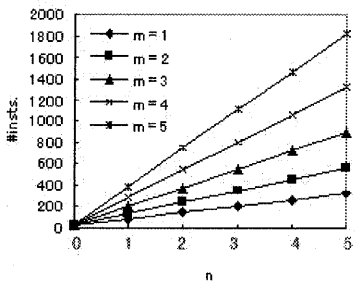

図 3: m を固定した場合の命令数の増加傾向

回である (ただし, $l = m$). **図 2** は n を固定して m を増やした場合の, **図 3** は m を固定して n を増やした場合の命令数の増加の様子である. 図 2 で, m が 1 増えるごとに増加する命令数は $(17m + 16)n$ 命令で, キーとロックのマッチング 1 回当たり 17 命令である. また, 図 3 の直線の傾きで表される, n が 1 増えるごとに増加する命令数は $\{17m(m+1)/2 + 16(m-1) + 41\}$ 命令, 切片で表される, アクセス権チェックを起動する際の固定のオーバーヘッドは 29 命令である. したがって, サイズ $m \times n$ のロックリストが設定されたオブジェクトを呼出す際のアクセス権チェックに要する命令数は, 最悪で $\{17m^2 + 49m + 25\}n/2 + 29$ 命令となる.

次に, 拡張キー/ロック方式によるアクセス権チェックのコストについて, 従来の ACL (access control list) 方式との比較を試みた. 3 エントリの ACL を用いて同様の実験を行った. その結果, アクセス権チェックに要する命令数は, 最初のエントリでアクセスが許可される場合の 39 命令～最後のエントリでアクセスが許可される場合の 81 命令であった. 従来の ACL ではアクセスルートコントロールを提供できないため単純な比較はできないが, ロックリストエントリ数 n と ACL のエントリ数を対応づけて考える. ACL 方式では, エントリ数を k とすると, アクセス権チェックに要する最悪の場合のコストは k に比例する. したがって, m が大きくなるにつれて拡張キー/ロック方式のコストが ACL 方式のコストに比べて増大するが, 実際に使用されるロックの殆んどは $m \leq 3$ であると考えられる. **表 1** は, m を大きくした時に増加する最悪の場合の命令数を示している. 例えば, ロックリストのサイズを 2×2 から

表 1: m の増加に伴う命令数の増加

	$m = 1 \rightarrow 2$	$m = 2 \rightarrow 3$
n = 1	50	67
n = 2	100	134
n = 3	150	201

3×2 にした場合は 134 命令の増加であった.

ロックリストのサイズが大きくなるとアクセス権チェックのコストも大きくなるが, 拡張キー/ロック方式によるアクセス権チェックが行なわれるのは, アプリケーション実行中に発生する TLB ミス時のみである. 通常, TLB ミス率は非常に小さいので, 拡張キー/ロック方式によるアクセス権チェックのコストの増分がアプリケーションの実行時間に与える影響は僅かであり, 実用的なオーバーヘッドでアクセスルートコントロールを実現できると考えられる.

5 おわりに

互いに協調する複数のモジュールによって構成される, 動的に拡張可能なオペレーティングシステムでは, モジュールの多様な協調関係に対応できる柔軟なアクセス制御機構が要求される. そこで, モジュール間の協調関係とスレッドのアクセス履歴に基づいたアクセス制御を実現する拡張キー/ロック方式を適用した. 拡張キー/ロック方式は, 柔軟なアクセスルートコントロールが可能である反面, 従来のアクセス制御方式よりもアクセス権チェックに要するコストが大きい. しかしながら, ソフトウェア制御のタグ付き TLB を備えたプロセッサ上で実装を行なうことでアクセス権チェックが必要となる機会を削減し, 効率化を図ることができる. シミュレーションの結果から, アクセスルートコントロールが実用的なコストで実現できると考えられる.

参考文献

[1] Bershad, B., Savage, S., Pardyak, P., Sirer, E. G., Becker, D., Fiuczynski, M., Chambers, C. and Eggers, S.: Extensibility, Safety and Performance in the SPIN Operating System, *Proc. of the 15th ACM Symposium on Operating System Principles*, pp. 267-284 (1995).

[2] Chase, J. S., Levy, H. M., Feeley, M. J. and Lazowska, E. D.: Sharing and Protection in a Single Address Space Operating System, *ACM Transactions on Computer Systems*, Vol. 12, No. 4, pp. 271-307 (1994).

[3] Grimm, R. and Bershad, B. N.: Access Control in Extensible Systems, *techinical report, Dept. of Computer Science and Engineering, University of Washington*, UW-CSE-97-11-01 (1997).

[4] Hamilton, G. and Kougiouris, P.: The Spring Nucleus, *Proc. of Summer USENIX Technical Conference*, pp. 147-159 (1993).

[5] Shigeta, S., Okamoto, S., Shimizu, K. and Sowa, M.: A Flexible Access Control Mechanism Based on the Key/Lock Scheme, *Proc. of Int. Technical Conf. in Circuits/Systems, Computers and Communications*, pp. 1385-1388 (1999).