システムソフトウェアと
オペレーティング・システム 81－16
（１９９９．５．６）

細粒度保護ドメインのための多重保護ページテーブルの提案と実装

河野 健二　品川 高廣　高橋 雅彦　益田 隆司

東京大学大学院 理学系研究科 情報科学専攻
E-mail: {kono, shina, masa, masuda}@is.s.u-tokyo.ac.jp

要旨

インターネットなどの開放的な分散環境を通じてプログラムコードをダウンロードし，そのコードをローカルな計算機環境上て利用することはきわめて一般的である．本稿では，このようなプログラムコードを安全に実行するための保護モデルとして，細粒度保護ドメインによるモデルを提案し，その効率的な実装方法を示す．細粒度保護ドメインは単一のプロセス内に複数の程御メインを共存させるものであり，計算機資源を共有しながらも資源へのアクセス権のみが異なっている．保護ドメインの切替にはアクセス権限のみを切り替えればよいので，軽量な保護ドメイン間呼び出しが提供できる．本稿では，細粒度保護ドメインの実現手法として，多重保護ページテーブルと保護つきシステムコールとを提案する．この方式にもとづいた実装では，従来の保護ドメイン間呼び出しにくらべ，細粒度保護ドメイン間呼び出しのオーバヘッドは40分の1から10分の1程度まで削減できる．

Multi-Protection Page Table: A Mechanism of Fine-Grained Protection Domains

Kenji Kono　Takahiro Shinagawa　Masahiko Takahashi　Takashi Masuda

Department of Information Science,
Graduate School of Science,
University of Tokyo
E-mail: {kono, shina, masa, masuda}@is.s.u-tokyo.ac.jp

Abstract

This paper proposes the protection model for the future programming paradigm of code mobility. The novelty of the model is the support of fine-grained protection domains that enable multiple domains to co-exist within a single process. As an efficient implementation of fine-grained protection domains, this paper presents multi-protection page tables and protected system calls. A multi-protection page table is an extension of traditional page tables to provide the virtual memory mechanism for fine-grained protection domains. The protected system call endows ordinary system calls with fine-grained control of protection.

1 はじめに

インターネットなどの広域分散環境の特徴のひとつは，その匿名性の高さにある．従来の分散システムの多くが，あらかじめ登録されたユーザのみが使用する閉じた分散環境であったのに対し，広域分散環境は不特定多数のユーザが利用する開放性の高い環境であり，ユーザの匿名性が高い．こうした開放的な分散環境に接続された計算機システムでは，その匿名性を悪用した不正なアクセスから計算機資源を保護する必要がある．計算機資源の保護はオペレーティング・システム（OS）の提供するもっとも基本的なサービスであるにもかかわらず，これまでに提案されてきたOSの多くが閉じた分散環境を暗黙に仮定しており，開放的な分散環境における資源の保護機構を備えたOSは少ない．

われわれの研究グループでは，開放的な分散環境でのセキュアな実行環境を提供することを目標に，カーネルの提供すべき資源保護の機構について研究を進めている．本稿では，カーネルの提供すべき保護機構として細粒度保護ドメインの必要性を示し，その実現機構である**多重保護ページテーブル**と**保護付きシステムコール**とを提案する．細粒度保護ドメインとは，従来のOSが提供するプロセスとしての保護ドメインより粒度の小さい保護ドメインであり，ひとつのプロセス内に複数の保護ドメインが共存できるものである．従来のOSにおける保護ドメインはプロセスと一体であり，プロセスの属性（例えばUNIXにおける実効ユーザID）によって資源へのアクセス権が決定される．それに対して，細粒度保護ドメインでは単一のプロセス内であっても，プロセスごとに割り当てらた計算機資源は共有しつつ，細粒度保護ドメインごとに異なる権限を持たせることができる．プロセスと細粒度保護ドメインの関係はプロセスとスレッドの関係に類似している．

本稿で示す多重保護ページテーブルとは，従来のページテーブルを拡張したものであり，細粒度保護ドメインごとに異なったメモリアクセス権を持てるようにしたものである．また，保護付きシステムコールとは，細粒度保護ドメインごとにシステムコールの発行権限を設定できる機構である．このふたつの機構を用いることによって，ある細粒度保護ドメイン内で実行中のスレッドは，その細粒度保護ドメインの権限にしたがって，メモリアクセスやI/Oを行うよう制約される．

細粒度保護ドメインを用いると，システムの外部からダウンロードされたプログラムやコンポーネントを安全に実行することが可能になる．例えば，WWWのブラウザであるNetscape Navigatorなどでは，プラグイン(plug-in)と呼ばれるコンポーネントをダウンロードし，ブラウザのアドレス空間に組み込んで実行することができる．インターネットの匿名性ゆえに，システムの外部からダウンロードされたプログラムコードはその作成者が特定しにくく，パスワードの盗聴やローカルなファイルを破壊したりする悪意のあるコードを排除することが難しい．本稿で提案する細粒度保護ドメインを用いると，外部から入手したコンポーネントを細粒度保護ドメイン内で実行させ，悪意のあるコードによる計算機資源への不正なアクセスを防ぐことができる．

以下，2章では関連研究についてまとめる．3章では細粒度保護ドメインに基づいた保護のモデルを述べ，4章では細粒度保護ドメインの実装を示す．5章ではプロトタイプシステムによる実験結果を報告し，6章で本稿をまとめる．

2 関連研究

単一のプロセス内に複数の保護ドメインを実現する研究には，ハードウェアの機構を活用したカーネルによる方法と，ソフトウェアのみで実現する方法とに大別できる．カーネルによる保護は仮想記憶のためのハードウェア(MMU)を利用した保護であり，Multics [2] などのリングプロテクションやOpal [1] などの単一仮想記憶OSにおける保護などがある．しかしながら，リングプロテクションは数段階の階層的な保護を行う仕組みであり，階層構造を持たない任意の個数の細粒度保護ドメインを提供することはできない．また，Opalは64-bitの仮想アドレス空間内に複数のセグメントを配置し，セグメントごとにメモリアクセスの権限を変更できるようにしている．メモリ以外の資源へのアクセス権はスレッドの属性によって決定される．そのため，すべての資源を共有しつつアクセス権限のみが異なるという細粒度保護ドメインの考えとは異なっている．

ソフトウェアによる保護では，言語処理系がさまざまな手法を用いてソフトウェアのみで保護を実現している．主な方式には，(1) 仮想機械方式，(2) コード修正方式，(3) 証明添付形式，の三つがある．仮想機械方

式は，中間コードを実行する仮想機械によってチェックを行う方式であり，Java [3] などがある．コード修正方式は，実行コードの中にコードの安全性を保証するためのコードを挿入する方式である．たとえば，SFI [7] という方式では，メモリアクセス命令やジャンプ命令の直前に命令を挿入し，不正な領域へのアクセスを防止している．証明添付方式は，実行コードがあらかじめ定められた保護ポリシーに従っていることを保証する証明を添付する方式である．PCC [4] という手法が提案されている．

本稿で示す方式は，ハードウェアの機構を利用したカーネルによる方式である．カーネルによって保護を提供する利点は，ハードウェアによって不正アクセスを検出するため，実行時のオーバーヘッドが低く，どのような方法で作成された実行可能コードに対しても保護が提供できる点にある．多くの文献 [6] でハードウェアによる保護ドメインは，保護ドメイン間呼び出しのコストが大きく細粒度保護ドメインの実現には適さないと言われている．しかしながら，これは従来のOSが細粒度保護ドメインのための適切な抽象化を提供していなかったためであり，本稿では，適切な抽象化と実装を行えば，十分に軽量な細粒度保護ドメインが実現できることを示す．

3 細粒度保護ドメインのモデル

細粒度保護ドメインに基づいた保護では，図1に示したように，ひとつのプロセス内に複数の細粒度保護ドメインが共存している．同じプロセス内にある細粒度保護ドメインは，仮想アドレス空間，タイムスライス等のすべての計算機資源を共有しており，そうした計算機資源へのアクセス権限のみが異なっている．

3.1 ポリシーモジュール

それぞれのプロセスには，細粒度保護ドメインごとのアクセス権限を決めるポリシーモジュールと呼ばれるモジュールがただひとつ存在し，そのプロセスの保護ポリシーを定めている (図1を参照のこと)．われわれのモデルの特徴のひとつは，保護のポリシーと保護のメカニズムとの明確な分離にある．カーネルは保護のメカニズムのみを提供し，保護のポリシーに関しては一切関知しない．保護のポリシーはすべてポリシーモジュールが決定し，カーネルはそのポリシーにしたがって保護を行うのみである．

ポリシーモジュールは，カーネルからの保護ポリシーに関する問い合わせに返答するコールバック・ルーチンの集まりである．カーネルとポリシーモジュール間のインターフェースは厳密に定めてあり，細粒度保護ドメインごとにメモリのアクセス権，システムコールの発行権限を定められるようになっている．カーネルは必要に応じてポリシーモジュールをアップコールし，ポリシーモジュールの定める保護ポリシーにしたがって保護を行う．たとえば，ある細粒度保護ドメイン内のスレッドが，あるファイルをオープンするシステムコールを発行すると，ポリシーモジュールがアップコールされ，そのファイルに対するアクセス権限がチェックされる．その結果に応じてカーネルはファイルのオープンを許可したり拒否したりする．

3.2 多重保護ページテーブル

多重保護ページテーブルは，ポリシーモジュールによって決められたメモリ保護を実現するための機構である．ひとつのプロセス内の細粒度保護ドメインは，ひとつの仮想アドレス空間を共有しているが，メモリページに対するアクセス権は細粒度保護ドメインごとに異なっている．従来のページテーブルの各エントリは，仮想ページ番号と物理ページ番号のマッピングとそのページのアクセス権のタプルであるのに対し，多重保護ページテーブルでは，それぞれのページが複数の保護モードを持てるようになっている．図1に示されているように，保護モードの各列はひとつの細粒度保護ドメインに対応しており，対応する細粒度保護ドメインのメモリページへのアクセス権を表している．この機構によって悪意のあるコードからほかのコンポーネントを保護することができる．

なお，ポリシーモジュール自身が他のコンポーネントから不正にアクセスされることを防ぐために，ポリシーモジュール自身もひとつの細粒度保護ドメインによって保護する．通常の場合，ポリシーモジュールのメモリ領域は他の細粒度保護ドメインからは保護し，逆にポリシーモジュールからは他の細粒度保護ドメインのメモリを参照できるように設定する．これは保護ポリシーを決める際に，他の細粒度保護ドメインの持つデータに自由に参照できるようにするためである．

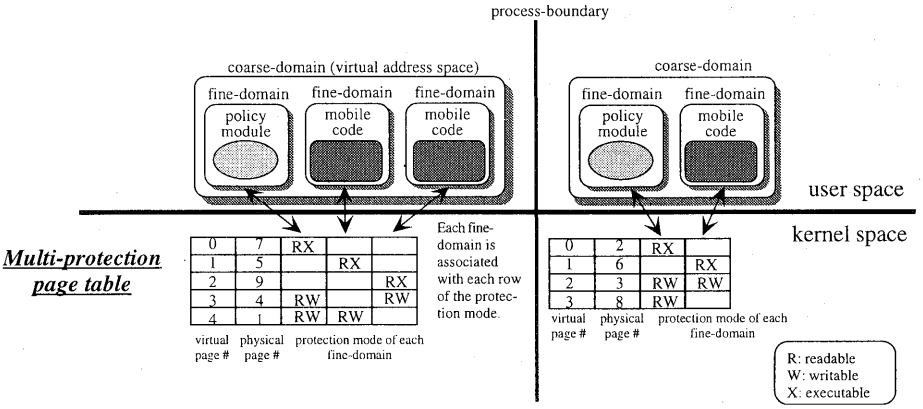

図 1: 細粒度保護ドメインのモデル

3.3 保護付きシステムコール

保護付きシステムコールは，ポリシーモジュールによって決められた発行権限にしたがって，システムコールの発行を許可したり拒否したりする機構である．ある細粒度保護ドメイン内のスレッドがシステムコールを発行すると，カーネルはポリシーモジュールをアップコールし，ポリシーモジュールからシステムコール発行の可否を知らせてもらう．発行が許可されればカーネルはシステムコールを実行する．発行が拒否された場合は，システムコールは実行せずにエラーコードを返す．

ポリシーモジュールは，細粒度保護ドメインごとにコールバックルーチンを用意する．ある細粒度保護ドメインがシステムコールを発行すると，その細粒度保護ドメインの識別子とシステムコール番号を，システムコールの引数といっしょにコールバックルーチンに引き渡す．コールバックルーチンはそれらの情報からシステムコールの発行の可否を決定する．このとき，ポリシーモジュールは他の細粒度保護ドメインのメモリ領域に自由にアクセスできるので，システムコールの引数にポインタが含まれていても，ポインタの参照先に自由にアクセスできる．

3.4 細粒度保護ドメイン間呼び出し

ひとつのプロセス内に共存する細粒度保護ドメインは，互いに密接に関連したコンポーネントであり，細粒度保護ドメイン間の呼び出しは頻繁に行われる．細粒度保護ドメイン間呼び出しでは，細粒度保護ドメインの切り替えを行ってアクセス権限の切り替えを行う必要がある．従来の OS では，アクセス権限を定める保護ドメインと，資源割り当ての単位であるプロセスとが一体であったため，保護ドメインを切り替えるには計算機資源も切り替える必要があった．われわれの提案する細粒度保護ドメインでは，計算機資源そのものは切り替える必要がなく，アクセス権限のみを切り替えればよい．

この特徴を活用して細粒度保護ドメイン切り替えを実装すると，ドメイン切り替えを軽量に実装できる．たとえば，仮想アドレス空間そのものは切り替える必要がないため，TLB のフラッシュ・再構築を行う必要はなくなる．タイムスライスも共有しているため，スケジューリングを行う必要もない．シグナルのデリバリや統計情報の収集も，資源割り当ての単位であるプロセスの切り替えまで遅延できるので，細粒度保護ドメインの切り替えの際に行う必要はない．

4 細粒度保護ドメインの実装

細粒度保護ドメインを実現する手法として，多重保護ページテーブルと保護付きシステムコールの実装方法を示す．また，細粒度保護ドメイン間呼び出しの実装方法を示す．

4.1 多重保護ページテーブルと保護ドメイン切替

多重保護ページテーブルは，従来のページテーブルを拡張して，ひとつの仮想ページに複数の保護モードを同時に設定できるようにしたものである．本節では，

多重保護ページテーブルを既存のプロセッサ上で実装する二つの方法を示す．ひとつの方式は，タグ付きTLBを備えたプロセッサ (SPARC, MIPS, Alpha 等) で利用できる方式であり，もう一方の方式はセグメント機構を持ったプロセッサ (Intel x86) で利用できる方式である．どちらかの方式を用いれば，現行の実質上すべての計算機上で多重保護ページテーブルを提供できる．

4.1.1 タグ付きTLBによる実装

タグ付きTLBとは，各TLBエントリにアドレス空間を表すタグを付けたものであり，複数のアドレス空間におけるアドレス変換 (仮想ページと物理ページのマッピング) をTLBに混在可能にする機構である．ある時点で有効なタグは，MMUの特別レジスタで指定される．この機構はコンテキストスイッチの際のTLBのフラッシュ・再構築のコストを回避するための機構であり，ページテーブルを切り替えるには有効タグを切り替えるだけでよい．本節では，この機構を用いた多重保護ページテーブルの実装を示す．

多重保護ページテーブルでは，各仮想ページに複数の保護モードを同時に設定できるため，既存のMMUが提供しているページテーブルをそのまま利用することはできない．しかし，タグ付きTLBでは，ページテーブルの切り替えは有効タグの切替えだけでよいという点に着目すると，次のように多重保護ページテーブルと細粒度保護ドメイン切替は実装できる．

- 細粒度保護ドメインごとに異なったページ保護モードを持つことができるように，各保護モードの列ごとにひとつのページテーブルを割り当てる．
- 細粒度保護ドメイン切替を行なうために，有効タグを切り替えるだけのソフトウェア割り込みを提供する．

細粒度保護ドメイン切替の際に，有効タグを切り替える操作のみを行なえばよいのは，すべての計算機資源が細粒度保護ドメイン間で共有されているからである．このように，細粒度保護ドメイン切替のコストを削減できる点が，われわれの提案する保護モデルの特徴である．

この方法を用いて，SuperSPARC(SPARCv8)上で細粒度保護ドメインの実装を行なった．この実装方法の詳細は文献[5]を参照されたい．

4.1.2 セグメントによる実装

Intelのx86アーキテクチャはタグ付きTLBを持たないため，前節で示した方法で多重保護ページテーブルを実装することはできない．しかしながら，x86アーキテクチャのプロセッサは広く利用されており，そのアーキテクチャ上での実装方法を示すことは実用上の意義が大きい．

x86アーキテクチャはページ化されたセグメントによってメモリ管理を行なう．仮想アドレスから物理アドレスへの変換は次の二段階のマッピングによって行われる．

1. **仮想アドレスからリニアアドレスへのマッピング:** 仮想アドレスは，セグメント・レジスタによって指定されたセグメントの先頭アドレスと足し合わされ，リニアアドレスに変換される．
2. **リニアアドレスから物理アドレスへのマッピング:** リニアアドレスは通常のページングと同じ手法によって物理アドレスに変換される．

このアーキテクチャの特徴は，同じ仮想アドレスであってもセグメントが異なれば異なるリニアアドレスにマッピングできる点にある．この点に着目して，細粒度保護ドメインごとにひとつのセグメントを割り当て，同じ仮想アドレスであっても細粒度保護ドメインが異なれば，異なるリニアアドレスにマップされるようにする．これによって，セグメントごとに異なるページの保護モードを持たせることが可能になる．リニアアドレスから物理アドレスへのマッピングは通常のページングと同じなので，セグメントが異なっても，同じ仮想アドレスは同じ物理アドレスにマップされるように設定する．これによって，セグメントが異なっても，同じ仮想アドレスであれば同じ物理アドレスを参照するようになる．この方式は，タグ付きTLBのタグの代替としてセグメントを用いたことに相当している．

細粒度保護ドメインの切り替えは，セグメントの切り替えによって実現できる．x86アーキテクチャでは，セグメントの切り替えはユーザモードでも実行できるので，不正なプログラムコードがセグメントを切り替えて不正にメモリ参照を行うことのないように配慮する必要がある．そのため，細粒度保護ドメインを切り替える時点で，切り替わった保護ドメインのセグメント以外のセグメントを，特権命令によって無効化して

いる．セグメントの有効・無効は特権命令でのみ切り替えられるので，ユーザプログラムが不正にセグメントにアクセスすることはない．

この機構はすでにPentiumプロセッサ上で実装済であり，その詳細については文献[8]を参照されたい．

4.2 保護付きシステムコール

保護付きのシステムコールとは，不正なシステムコールの発行を防止する機構であり，ポリシーモジュールの定める保護ポリシーに反したシステムコールの発行を禁止する．システムコールを発行すると，ポリシーモジュール内のコールバックルーチンを呼び出す．コールバックルーチンの呼び出し機構は，細粒度保護ドメイン間呼び出しの機構によって実装されており，呼び出し元の細粒度保護ドメインからポリシーモジュールの細粒度保護ドメインへと保護ドメインの切り替えを行う．ポリシーモジュールがアクセス権を検証したのち，システムコールの実行が許可されれば，通常のシステムコールと同様に実行される．

5 実験

われわれの提案する細粒度保護ドメインの機構によって，保護ドメイン間呼び出しは十分に低いコストで実現できる．このことを示すために，タグ付きTLBを用いた実装をSuperSPARC(60MHz)上に，セグメントを用いた実装をPentium(133MHz)およびPentiumII(400MHz)上にそれぞれ実装した．カーネルはLinux(version 2.0.35)を改造し，細粒度保護ドメインを組み込んだものである．

この実装を用いて細粒度保護ドメイン間呼び出しのサイクル数を計測した．この結果を表1に示す．また，比較のため，従来のプロセスを保護ドメインとした場合の保護ドメイン切り替えのオーバヘッドとして，通常のコンテキスト・スイッチのサイクル数も計測した．従来のコンテキスト・スイッチにくらべ，細粒度保護ドメインのドメイン間呼び出しのコストは10分の1から40分の1にまで削減されており，資源を共有しながら，資源へのアクセス権のみを切り替える，という細粒度保護ドメインのモデルの有効性を示唆していると言える．

方式	プロセッサ	cycles
タグ付きTLB	SuperSPARC (60MHz)	126
セグメント	Pentium (133MHz)	230
セグメント	PentiumII (400MHz)	584
context switch	Pentium (133MHz)	約5,600

表1: 保護ドメイン間呼び出しのサイクル数

6 まとめ

本稿では，開放的な広域分散環境のためのOSによる保護機構として，細粒度保護ドメインの必要性を述べ，その実装方法として多重保護ページテーブルと保護付きシステムコールの提案を行った．細粒度保護ドメインは，ひとつのプロセス内に複数の保護ドメインを共存させるもので，計算機資源を共有しながらもアクセス権限のみが異なる．この手法によって，従来のプロセスとしての保護ドメイン切り替えに比べ，細粒度保護ドメイン切り替えでは，そのコストが40分の1から10分の1まで削減されることを示した．

参考文献

[1] J. S. Chase, H. M. Levy, M. J. Feeley, and E. D. Lazowska. Sharing and protection in a single-address-space operating system. *ACM TOCS*, 12(4):271–307, November 1994.

[2] R. C. Daley and J. B. Dennis. Virtual memory, processes, and sharing in MULTICS. *Commun. of the ACM*, 11(5):306–312, May 1968.

[3] J. Gosling and H. McGilton. The Java language environments: A White Paper. Technical report, Sun Microsystems, 1995.

[4] G. C. Necula. Proof carrying code. In *ACM POPL*, Jan. 1997.

[5] M. Takahashi, K. Kono, and T. Masuda. Efficient kernel support of fine-grained protection domains for mobile code. In *Proc. of IEEE Int. Conf. on Distributed Computing Systems*, 1999. To appear.

[6] T. Thron. Programming languages for mobile code. *ACM Computing Surveys*, 29(3):213–239, September 1997.

[7] R. Wahbe, S. Lucco, E. Anderson, and S. L. Graham. Efficient software-based fault isolation. In *Proc. of ACM SOSP*, pages 203–216, 1993.

[8] 品川高廣, 河野健二, 高橋雅彦, 益田隆司. 拡張コンポーネントのためのカーネルによる細粒度軽量保護ドメインの実現. 情報処理学会論文誌, 1999. 掲載予定.