M-053

分散環境におけるセキュアな資源管理方式に関する研究

橋本 正樹†, 金 美羅†, 辻 秀典†‡, 田中 英彦†
{mgs053504, mira, tsuji, tanaka}@iisec.ac.jp

1 はじめに

分散システムは、単一システムと比較して経済性・冗長性・拡張性の面で優れているため、その実現に向けて従来より多くの研究が行われてきた。特に分散環境における資源管理の研究は、元来限りある計算資源を効率よく利用することを命題として行われてきており、そのために多くの機構が考案されてきた。

しかしながら、従来の分散環境における資源管理機構はミドルウェアやアプリケーションのような上位層での実装方式が主流であるために、実装者依存によるシステム全体の信頼性低下や資源管理粒度の粗さ、柔軟性の欠如といった問題がある。また、上位層での実装はシステムソフトウェア層が信頼できることを前提としているが、実際にそれを保証する仕組みはまだ一般的ではない。

一方、システムソフトウェア層において細粒度の強制的な資源管理を実施することにより信頼性を高める研究も行われているが、これらの研究は分散環境ではなく単一システムを想定している。

本研究は、分散環境の信頼性を高めるためにシステム基盤として提供される資源管理機構の提案を目的とする。そこで、本稿では資源管理においてアクセス可否決定の基準になるケイパビリティリストに関する考察を行う。

2 関連研究

Security-Enhanced Linux[2]は、今日の代表的なセキュアOSである。これはFlaskセキュリティアーキテクチャ[3]のLinuxカーネルに対する実装であり、単一計算機における資源管理機構として以下の特徴がある。第一は機構と方策の分離を実現していること、第二は細粒度の強制アクセス制御を実現していること、第三は管理対象資源のグループ化を柔軟に行うことができる仕組みを提供していることである。Security-Enhanced Linuxはこれらの特徴により、従来のOSより信頼性の高い資源管理を実現しているが分散システムとしての利用は想定されていない。

分散環境でのOS層における資源管理に関する研究としてはAmoeba[4]があるが、これは分散配置された単一ホストの集合を一つの統合されたシステムとして扱うことができる汎用分散オペレーティングシステムである。しかし、Amoebaは単一システムにおける資源管理機構と比較すると機構と方策の分離・権限管理の粒度・拡張性といった点において問題がある。

分散環境でのミドルウェア・アプリケーション層における資源管理に関する研究としては、The STRONGMAN Architecture[5][6]があるが、これはグローバルポリシーとローカルポリシーの整合性を図ることにより、インターネットの拡張性に対して新しいアプローチを行っているものである。 The STRONGMAN Architectureは、ネットワーク上のドメイン間で矛盾のない強制アクセス制御が可能であるが、計算機毎のアクセス制御が基本となるためAmoeba同様、資源管理の粒度が粗い。

3 想定アーキテクチャ

本稿で想定する分散環境のアーキテクチャを図１に示す。計算機Aと計算機Bは、Negotiator（N_B）経由で安全な通信が可能なものとし、計算機Aのサブジェクト（S_A）が計算機Bのオブジェクト（O_B）にこの経路を通じてアクセスするケースを想定する。想定アーキテクチャにおいて、Negotiatorはリファレンスモニタ[1]としての役割を担うものとし、他計算機のサブジェクトの認証・アクセス制御の強制・確実なデータ転送・機密性保持を保証するものとする。このような特性上、Negotiatorは各計算機の資源管理に関わる外部とのあらゆる通信を制御するものとして想定する。本稿ではこの想定アーキテクチャを前提として、Negotiatorがアクセス可否の決定を下す拠り所となる効率的な分散環境向けケイパビリティリストを提案する。

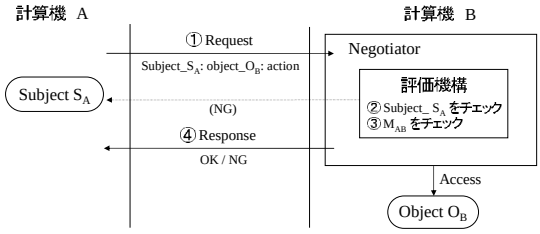

図１: 想定アーキテクチャ

ある計算機A内のサブジェクトが別の計算機B内のオブジェクトを操作する際は以下の手順を経るものとする。

① S_Aは計算機B内のNegotiatorにO_Bへのアクセスを要求する。
② N_BはS_Aを認証し、S_Aとの間に安全な通信路を確保する。以後、S_A・N_B間通信は全てこの通信路経由で行われるものとする。
③ N_Bはサブジェクトマッピングリスト（M_{AB}）を参照し、アクセス可否決定を行う。具体的には、S_Aの要求するアクセスが可能で、且つM_{AB}において、S_Aの代理アクセスを担当するサブジェクト（S_B）を計算機B内で探す。該当サブジェクトがなければアクセス不可となる。

†情報セキュリティ大学院大学情報セキュリティ研究科 ‡株式会社情報技研

④ S_BがO_Bに対する代理アクセスを行い、結果をN_B経由でS_Aへ返す。

4 分散ケイパビリティリストの提案

複数計算機のアクセス制御情報を一つの表に記述するやり方は、ケイパビリティリストが大きく複雑になり非効率的である。ケイパビリティリストはある計算機内におけるサブジェクトのオブジェクトに対する許可された操作を記述したものであり、この方法では以下の図のように表に記述する内容が総計算機数の２乗で増加する。

	O_{A1} O_{A2} ……	O_{B1} O_{B2} ……
S_{A1} S_{A2} :	計算機 A のケイパビリティリスト	追加分ケイパビリティリスト
S_{B1} S_{B2} :	追加分ケイパビリティリスト	計算機 B のケイパビリティリスト

図２： 計算機Aと計算機Bの全アクセス制御情報

そこで本提案においては、計算機Aと計算機Bのアクセス制御情報をひとつの表でまとめて管理するのではなくサブジェクト間の対応付けを記述したM_{AB}を導入する。この表には、計算機Aのあるサブジェクトが計算機Bのあるオブジェクトについて、計算機Bのあるサブジェクトと同等の操作が可能か否かを記載する。

本提案におけるM_{AB}は、以下の手順であらかじめ生成され以後は静的な表として安全に保持されるものとする。

① 計算機Aのサブジェクトで計算機Bのオブジェクトに対するアクセスを必要とするものを抽出し、アクセス対象のオブジェクトを含んだ表を生成する。

	O_{B3}	O_{B4}	O_{B13}	……
S_{A1}	r--	r-x	rwx	……
S_{A3}	rw	-wx	--x	……
:	……	……	……	……

② 計算機Bで計算機Aからのアクセスを許可するオブジェクトとそれに対応したサブジェクトを記述した表を生成する。

	S_{B2}	S_{B5}	S_{B11}	……
O_{B4}	--x	rwx	r--	……
O_{B13}	rw-	rw-	r--	……
:	……	……	……	……

③ ①および②のサブジェクト群についてM_{AB}を作成する。

	S_{B2}	S_{B5}	S_{B11}	……
S_{A1}	NG	OK	NG	……
S_{A3}	OK	NG	NG	……
:	……	……	……	……

本提案において、M_{AB}に記述されるサブジェクトは計算機Aと計算機Bのケイパビリティリストにおいてそれぞれに定義された全サブジェクトとはせず外部操作が必要なサブジェクトのみとする。計算機Aと計算機Bそれぞれのローカルシステムでのみ必要なサブジェクトをM_{AB}には記述しないことにより、対象となる計算機の数が増えても上記①・②のプロセスは変わらずM_{AB}のみが増える。結果として、アクセス制御情報の記述量を軽減することができる。

5 まとめと今後の課題

本稿ではシンプルな分散環境とリファレンスモニタの想定アーキテクチャを前提とし、効率的なケイパビリティリストの記述方法を提案した。これにより分散環境上の計算機数が増加した際のアクセス制御情報の記述量を軽減することが可能となった。ただし、本提案は最も基本的なものであって以下の項目は今後の課題である。

- 各手順に関するプロトコルの詳細化
- Negotiatorの動作メカニズムに関する検討
- 未知のサブジェクトやオブジェクトへの対応
- 上位概念としてのポリシーから動的にケイパビリティを決定するメカニズムの検討
- サブジェクトの状態に応じて動的にケイパビリティを変更するメカニズムの検討
- 分散ケイパビリティリストの保護に関して、アクセス可能な権限の明瞭化

参考文献

[1] Department of Defense, Trusted Computer System Evaluation Criteria, DOD 5200.28-STD, December 1985.

[2] “Security-Enhanced Linux,” National Security Agency, http://www.nsa.gov/selinux/.

[3] Ray Spencer, Stephen Smalley, Peter Loscocco, Mike Hibler, David Andersen, Jay Lepreau, “The Flask Security Architecture: System Support for Diverse Security Policies,” USENIX, August 1999.

[4] Andrew S. Tanenbaum, “The Amoeba distributed operating system,” Technical report, Vrije Universiteit, 1990.

[5] A Keromytis, S Ioannidis, M Greenwald, and J Smith, “The strongman architecture,” In Third DARPA Information Survivability Conference and Exposition (DISCEX III), April 2003.

[6] Matt Blaze, Joan Feigenbaum, Angelos D. Keromytis, “KeyNote: Trust Management for Public-Key Infrastructures,” In Proceedings of the 1998 Security Protocols International Workshop, April 1998.