

ユーザと OS 環境をアクセス主体とする 仮想計算機環境でのポリシー強制機構

諏訪 公洋† 平野 学‡ 奥田 剛† 河合 栄治† 山口 英†

† 奈良先端科学技術大学院大学
〒 630-0192 奈良県生駒市高山町 8916-5

‡ 豊田工業高等専門学校
〒 471-8525 愛知県豊田市栄生町 2-1

{kimihhiro-s, okuda, eiji-ka, suguru}@is.naist.jp

hirano@toyota-ct.ac.jp

あらまし 情報漏洩を防ぐためには、セキュリティポリシーを策定し、アクセス制御により実効性を担保する必要がある。アクセス主体やアーキテクチャにより、遵守できるセキュリティポリシーが異なってくるが、既存のアクセス制御では、アクセス主体としてユーザまたは OS 環境のどちらか一方しか扱えず、遵守できるセキュリティポリシーの粒度が粗くなっている。また、アクセス制御が有効に機能していることを保証しなければ、情報漏洩を防いでいることも保証することができない。本論文では、ユーザと OS 環境の両方をアクセス主体とし、仮想計算機環境を用いることでユーザの OS 環境に依存しないポリシー強制機構を提案し、設計・実装、評価した。

Policy Enforcement Mechanism based on both User Identity and User's Computer Environment in a Virtual Machine

Kimihhiro Suwa† Manabu Hirano‡ Takeshi Okuda† Eiji Kawai†
Suguru Yamaguchi†

†Nara Institute of Science and Technology
8916-5, Takayama, Ikoma, Nara 630-0192, Japan
{kimihhiro-s, okuda, eiji-ka, suguru}@is.naist.jp

‡Toyota National College of Technology
2-1, Eisei, Toyota, Aichi 471-8525, Japan
hirano@toyota-ct.ac.jp

Abstract In order to prevent information leakage, the security administrators have to make security policy and enforce the policy by using access control. Moreover, the security administrators have to make sure that a policy enforcement mechanism is not compromised and working properly. The problem here is that existing policy enforcement mechanism does not treat both user's identity and operating environment as subjects in their policies. In this paper, we propose a policy enforcement mechanism using virtual machine technology that can treat user's identity and operating environment as subjects in the policies. In this proposal, virtual machine monitor (VMM) is a policy enforcing point, and can be certified that VMM is not compromised. We explain the overview of the proposed policy enforcement mechanism.

1 はじめに

情報漏洩の増加が社会問題化してきている。情報漏洩を防ぐために、セキュリティポリシーを遵守する機構が必要である。セキュリティポリシーは、アクセ

ス主体とアクセス対象のアクセスの可否を決定するものである [1]。セキュリティポリシーをセキュリティポリシーモデルに基づいて記述し、実効性をアクセス制御によって担保する。

アクセス主体やアーキテクチャにより、遵守でき

るポリシーが異なってくるが、既存のアクセス制御機構は、ユーザか OS 環境のどちらか一方しか用いることができない。このため、アクセス制御の粒度が粗くなり、アクセス対象に対してアクセス可能なアクセス主体が不用意に増加してしまう。例えば、アクセス主体としてユーザを扱っていた時に、正規のユーザがウィルスに侵されている OS 環境を用いた場合、OS 環境に対してのアクセス制御を行えないため、情報漏洩につながる。また、アクセス制御が有効に機能していることを保証しなければ、情報漏洩を防いでいることも保証することができない。

2 情報漏洩と対策技術

2.1 セキュリティポリシーモデル

セキュリティポリシーには、形式的・数学論理的に分析する基点となるセキュリティポリシーモデルがある。セキュリティポリシーモデルは、計算機システムが備えるべきセキュリティに関する要件を明解かつ簡潔に記述したものである。セキュリティポリシーをセキュリティポリシーモデルに基づいて記述することにより、セキュリティポリシーモデルごとの目的に応じた安全性が保証される。セキュリティポリシーモデルには ChineseWall (CW) や Type Enforcement (TE), Role-based Access Control (RBAC) 等がある。

2.2 アクセス制御機構

セキュリティポリシーの実効性を担保するためにはアクセス制御を行う必要がある。アクセス制御手法には、任意アクセス制御 (DAC) と強制アクセス制御がある (MAC)。DAC はユーザに所有権のあるオブジェクトのセキュリティポリシーの決定をユーザに行うことができる。これに対して MAC は、システムによりセキュリティポリシーが決定され、ユーザは所有権のあるオブジェクトに対してのセキュリティポリシーを変更することができない。これにより、MAC はユーザ権限を奪取された際にアクセスされるオブジェクトを限定することができ、DAC に比べて情報漏洩を最小化することができる。

また、ネットワークに対するアクセス制御を実現するものには、IEEE802.1x や認証 VLAN がある。これらのアクセス制御は、アクセス主体としてユーザを用いている。仮想計算機環境では、仮想計算機モニタ (VMM) でゲスト OS からの命令をモニタリングできることを利用し、オブジェクトへのアクセ

スを制限した [2] や仮想計算機間の隔離性を活かしてネットワークアクセス制御を行う NetTop[3] がある。これらのアクセス制御は、アクセス主体として OS 環境を用いている。また、仮想計算機を用いることにより、ゲスト OS の内部状態に依存せずにポリシーの強制を行える。OS の内部状態に依存しないようにするために、ネットワークへに接続する機器の真正性を検証する Trusted Network Connect (TNC) [4] もある。ただし、OS 環境のカーネル以上の真正性の検証は、OS 環境専用のアプリケーションが必要となる。

2.3 既存のアクセス制御と情報漏洩

アクセス主体をユーザとするアクセス制御を用いることで、不正規ユーザのアクセスを防ぐことができる。また、アクセス主体を OS 環境とすることで、ユーザの誤操作を防ぐことができる。また、アクセス制御アーキテクチャとして仮想計算機環境を用いることで、OS 環境がウィルスにおかされていても、OS 環境の内部状態に依存せずにポリシーの強制を行える。TNC では OS の真正性を検証するために OS 環境専用のアプリケーションが必要なため、OS 環境の汎用性においては仮想計算機環境が優位である。

ここで挙げたアクセス制御機構は、アクセス主体としてユーザか OS 環境のどちらか一方のみを用いていた。しかし、これではアクセス制御の粒度が粗くなってしまふ。このため、正規のユーザがウィルスに侵されている計算機を用いた時に、接続を拒否することができず、ウィルスを介して情報漏洩が生じてしまう。また、アクセス制御が有効に機能していることを保証することもできない。

3 仮想計算機環境を用いたポリシー強制機構の提案

本論文では、ユーザと OS 環境の両方をアクセス主体として、仮想計算機環境でのポリシー強制を行う機構を提案する。このポリシー強制機構では、RBAC を用いることで、ロールに基づいた強制アクセス制御を行う。

3.1 仮想計算機環境

仮想計算機環境とは、ソフトウェアで実装した計算機 (仮想計算機) 上で、OS (ゲスト OS) を動作させている環境のことである。本論文では、ゲスト OS

する。

5.1 抽象化ポリシー

ネットワークアクセス制御のポリシーを (1) ユーザ ID と OSID の組に対して割り当てられているロール、(2) ファイルに割り当てられているロール、(3) ファイルと保存されているサーバの関係、の3つを記述する。また、ゲスト OS の起動許可は (1) ユーザ ID と OSID の組に対して割り当てられているロール、のユーザ ID と OSID の組を記述することで表現する。この抽象化ポリシーは、出現順序によらず互いの関係性が独自の意味を持っているため、このような記述がしやすい XML を用いて記述した。

5.2 ポリシマネージャ

アクセス制御層から送られいくユーザ ID、OSID、IP アドレスと後述するタップデバイス名を保持する。XML で記述された抽象化ポリシーから、XSLT を用いて具体化ポリシーを生成し、アクセス制御層やサーバに配布する。

ポリシマネージャの偽装防止は、専用の証明書をを用いた IPSec の通信でサーバとアクセス制御層へのポリシーの配布を行うことで実現した。

5.3 アクセス制御層

仮想計算機環境として広く使われている Xen や KVM にも用いられている QEMU[6] にて実装した。QEMU にはタップデバイスをネットワークインターフェースとブリッジ接続することにより、実ネットワークと通信するタップモード、模擬的な DNS サーバ等を VMM 内部に保持し、実ネットワークとの通信をホスト OS の IP アドレスや MAC アドレスに変換して行うユーザネットワークモードがある。今回は、タップモードを用いて実装する。

ゲスト OS を起動させようとする、仮想計算機環境内で一意に識別できる名前の付けられたタップデバイスが生成され、仮想計算機のネットワークインターフェースとなる。また、タップデバイスはゲスト OS の停止とともに消失する。そこで今回、タップデバイス名を VMID として用いることとした。タップデバイスの生存に基づいてゲスト OS の生存情報をポリシマネージャに一定時間ごとに通知する。アクセス制御層は、スマートカードの PIN の入力によりユーザがスマートカードの所有者であることを検証し、ユーザ ID をスマートカードから取得した

後、OSID をユーザの入力により取得する。ユーザ ID、OSID、IP アドレス、タップデバイス名をポリシマネージャに送り、具体化ポリシーを受け取り、強制する。

ポリシーの強制を行う際に、TAP デバイスに割り当てた MAC アドレスを用いてゲスト OS が MAC アドレスを偽装した通信を行えないようにする。アクセス主体情報とゲスト OS を関連づけることのできる。タップデバイスをネットワークアクセス制御のポリシーの強制ポイントとし、ゲスト OS からの送付パケットへのアクセス主体の関連付けも行う。今回は、送付パケットとアクセス主体の関連付けを簡略し、IP ヘッダの TOS フィールドに固有の値を入れることで行った。また、アクセス制御層の偽装防止は、ポリシマネージャと同様に IPSec による通信で担保している。

6 評価

偽装防止機能とゲスト OS の生存情報による動的ポリシーの生成と配布機能を備えることで、偽装したゲスト OS によるオブジェクトへのアクセスや偽装したサーバを介しての情報漏洩を防止している。しかし、ゲスト OS の生存情報が通知されないように、ネットワークケーブルを抜くと、生存情報が通知されないことにより、ポリシマネージャが実計算機の停止と判断するまでにタイムラグがある。このタイムラグが発生する一定時間はオブジェクトに不正アクセスされる危険がある。タイムラグを短くすると、ゲスト OS を起動させている実計算機の増加に比例して、ローカルエリア内のネットワーク帯域の消費が増加してしまうデメリットも生じる。

また、仮想計算機環境では、VMM で取得できるゲスト OS の内部状態の情報とゲスト OS の内部状態を示す情報には Semantic Gap[7] と呼ばれる意味情報の差異がある。Semantic Gap があるため、提案したポリシーの強制機構の適用範囲が限定される。これは、仮想計算機環境を用いることで、OS 環境の種別や内部状態に関わらずポリシーを強制できるようにしたこととのトレードオフである。そのため、OS の内部状態を示す情報を取得できる OS 内のアクセス制御機構と連携してポリシーを強制することで情報漏洩を防止することができる相互補完するポリシーの強制機構である。

7 考察

本論文では仮想計算機環境において、ゲスト OS 以外の階層は安全であるという前提をもとに行った。しかし、仮想計算機環境の多くは脆弱性が指摘されている [8][9]。さらに仮想計算機環境における CPU の脆弱性も指摘されており [10][11]、前段階では仮想計算機環境を安全な環境とはいいたがたい。しかしながら、仮想計算機環境を用いたアクセス制御や隔離空間の構築の需要があるため、脆弱性に関する研究や対策が行われており、仮想計算機環境の安全性の向上が図られていくと考えている。また、今回はタップモードを用いて実装を行った。しかし、ユーザネットワークモードでも iptables と同様の機能をアクセス制御層に実装することで実現可能である。このように、特定の実装にしばられずに汎用性の高い設計であると考えられる。

8 今後の課題

今回はアクセス対象のオブジェクトをファイル、アクセス制御の単位をサーバとして設計・実装したが、ゲスト OS の内部状態を示す情報をアクセス対象やアクセス制御の単位とすることはできない。この問題に対して、Semantic Gap を克服しようとする研究が行われ [12][13]、解決には至っていないが、軽減されてきている。適用範囲の拡大に Semantic Gap の軽減を活かしていくことが、今後の課題である。

9 おわりに

我々は、アクセス主体としてユーザと OS 環境を用いたアクセス制御を仮想計算機環境下で行うことを提案し、設計・実装及び評価した。提案したポリシー強制機構を用いることで、従来よりも粒度の詳細化したアクセス制御をユーザの利用する OS 環境の種別や内部状態にかかわらず強制することができるようになった。

参考文献

[1] 独立行政法人情報処理推進機構セキュリティセンター。アクセス制御に関するセキュリティポリシーモデルの調査報告書。http://www.ipa.go.jp/security/fy16/reports/access_control/documents/PolicyModelSurvey.pdf, March 2005.

[2] Reiner Sailer, Trent Jaeger, Enrique Valdez, Ramon Caceres, Ronald Prez, Stefan Berger, John Linwood Griffin, and Leendert van Doorn. Building a MAC-Based Security Architecture for Xen Open-Source Hypervisor. In *21st Annual Computer Security Applications Conference (AC-SAC 2005)*, December 2005.

[3] Robert Meushaw and Donald Simard. NetTop. <http://www.vmware.com/pdf/TechTrendNotes.pdf>, 2000.

[4] Trusted Computing Group and Microsoft Corporation. Standardizing Network Access Control: TNC and Microsoft NAP to Interoperate. https://www.trustedcomputinggroup.org/news/Industry_Data/TNC_NAP_white_paper_final_may_18_07.pdf, May 2007.

[5] Manabu Hirano, Takeshi Okuda, Eiji Kawai, and Suguru Yamacuchi. Design and Implementation of a Portable ID Management Framework for a Secure Virtual Machine Monitor. In *2nd International Workshop on Secure Information Systems (SIS'07)*, October 2007.

[6] Fabrice Bellard. QEMU, a Fast and Portable Dynamic Translator. In *USENIX 2005 Annual Technical Conference (USENIX '05)*, April 2005.

[7] Peter M. Chen and Brian D. Noble. When virtual is better than real. In *Hot Topics in Operating Systems*, pp. 133 – 138, Los Alamitos, CA, USA, 2001. IEEE Computer Society.

[8] Peter Ferrie. Attacks on virtual machine emulators. <http://www.symantec.com/avcenter/reference/Virtual.Machine.Threats.pdf>, January 2007.

[9] Tavis Ormandy. An empirical study into the security exposure to hosts of hostile virtualized environments. <http://taviso.decsystem.org/virtsec.pdf>.

[10] Joanna Rukowska. Red Pill... or how to detect vmm using (almost) one cpu instruction. <http://www.invisiblethings.org/papers/redpill.html>, November 2004.

[11] John Scott Robin and Cynthia E. Irvine. Analysis of the Intel Pentium's ability to support a secure virtual machine monitor. In *SSYM'00: Proceedings of the 9th conference on USENIX Security Symposium*, Berkeley, CA, USA, 2000. USENIX Association.

[12] Stephen T. Jones, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau. Antfarm: Tracking Processes in a Virtual Machine Environment. In *USENIX 2006 Annual Technical Conference (USENIX '06)*, Boston, MA, June 2006.

[13] Stephen T. Jones, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau. Geiger: Monitoring the buffer cache in a virtual machine environment. In *Architectural Support for Programming Languages and Operating Systems (ASPLOS XII)*, San Jose, CA, October 2006.