社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－DSM－48 (6)
2008－QAI－26 (6)
2008／3／6

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

‡豊田工業高等専門学校
〒471-8525 愛知県豊田市栄生町 2-1
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

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

†Nara Institute of Science and Technology
8916-5, Takayama, Ikoma, Nara 630-0192, Japan
{kimihiro-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を遵守する機構が必要である．セキュリティポリシは，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をセキュリティポリシ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モデル

セキュリティポリシには，形式的・数学論理学的に分析する基点となるセキュリティポリシモデルがある．セキュリティポリシモデルは，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をセキュリティポリシ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の決定をユーザにて行うことができる．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う．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強制を行う機構を提案する．このポリシ強制機構では，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

から仮想計算機に対して発行した命令を実計算機に対しての命令を仮想計算機モニタ (VMM) にて変換するものとする．このように仮想計算機環境は，実計算機，仮想計算機モニタ，仮想計算機，ゲストOSの4階層に成り立っているものとする．

3.2 ポリシ強制機構の提案

ユーザが利用するOS環境をゲストOSとし，他の仮想計算機環境の階層は管理者により，運用上の安全性が保証されているものとする．提案するポリシ強制機構は図1のようになる．ゲストOSは，OSのファイルイメージを仮想計算機に読み込むことで起動する．よって，OS環境はOSのファイルイメージと対応しているといえる．そこで，管理者が指定したOSのファイルイメージのみ起動させるようにし，OSのファイルイメージにID (OSID) を割り当てる．OSIDに対してのアクセス制御をすることで，OS環境に対してのアクセス制御を行う．また，ユーザに対してもユーザIDを割り当てる．ユーザがゲストOSを起動する際に，アクセス主体をユーザID，アクセス対象をOSIDとすることで，ゲストOSの起動制御を行う．

次に，アクセス主体をユーザとOS環境，アクセス対象をネットワーク上にあるオブジェクトとすることで，ネットワークアクセス制御を行う．実計算機環境では，ネットワークアクセス制御にIPアドレスやMACアドレスを用いることができるが，仮想計算機環境では，IPアドレスやMACアドレスが変化する可能性があり，802.1xなどのポート認証では，1つの実計算機上の複数の仮想計算機が同一に扱わされてしまう．各アクセス制御を，VMMにて行うことにより，ゲストOSの内部状態やWindowsやLinux等のOSの種別に依存せずにアクセス制御を行うことができる．また，ゲストOSから送出されるパケットとアクセス主体の関係を維持できるようにし，サーバでもアクセス制御を行う．さらに，実計算機やゲストOSが停止した場合に，サーバと仮想計算機環境では強制すべきセキュリティポリシが異なってくるため．ゲストOSが起動したままか停止したかの生存情報を，強制するセキュリティポリシに反映する．

セキュリティポリシは，アクセス主体とロール，ロールとアクセス対象を記述した抽象化ポリシとして記述する．ゲストOSの起動時と停止時に，実際に各強制を行うにあたって必要な具体化ポリシを生

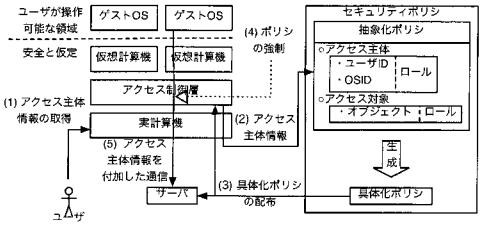

図 1: ポリシ強制機構

成し，ユーザの利用している仮想計算機環境とサーバに配布する．また，偽装した具体化ポリシを配布をされないように，具体化ポリシの配布元に偽装防止機能を備える．

4 設計

ここでは，アクセス対象としてオブジェクトをファイル，アクセス制御の単位をファイルの保存してあるサーバとしてセキュリティポリシを定める．また，後述するアクセス制御層で用いるスマートカードは，ユーザごとに1枚ずつ配布されており，スマートカードが抜かれるとゲストOSが停止すると仮定する．各アクセス制御層に一意に定める固有のID (ACLID) が割り当てられているとし，ゲストOSの起動時に割り当てられた仮想計算機の仮想計算機環境内でのIDをVMIDとする．提案内容をもとに，(1) 抽象化ポリシ，(2) ポリシマネージャ，(3) アクセス制御層の3つの機能にわけて設計する．抽象化ポリシはセキュリティポリシの記述，ポリシマネージャはポリシの生成と配布を行う．また，アクセス制御層は仮想計算機モニタにてゲストOSの起動制御やネットワークアクセス制御，パケットとアクセス主体の関係の維持を行う．ポリシの強制機構の全体の構造は図2のようになる．

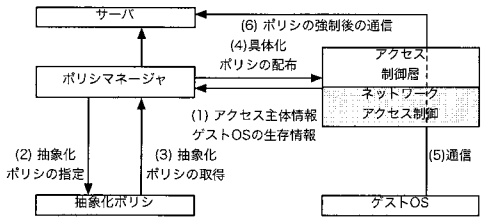

図 2: ポリシ強制機構の構造

4.1 抽象化ポリシ

抽象化ポリシにはゲストOSの起動制御ポリシとネットワークアクセス制御ポリシがある．ゲストOSの起動制御ポリシは，アクセス主体であるユーザIDからアクセス対象であるOSIDを起動できるかどうかの直行関係に基づいて記述する．

ネットワークアクセス制御ポリシはRBACに基づいて記述する．ネットワークアクセス制御ポリシは，アクセス主体であるユーザIDとOSIDにロールを割り当てる．また，アクセス対象であるファイルに対してもロールを割り当てる．アクセス主体からアクセス対象へのアクセスの可否は，ロールに基づいて決定する．また，アクセス対象であるファイルとアクセス制御単位であるファイルを保存しているサーバの関係を記述する．

4.2 ポリシマネージャ

ポリシマネージャはアクセス制御層からユーザIDとOSID，また起動しようとしているVMID，ACLIDを受け取り，保持しておく．受け取ったユーザIDが起動可能なOSIDを抽象化ポリシから検索し，受け取ったOSIDと比較することで，起動可能であるかどうかの起動制御用の具体化ポリシを生成する．

起動する権限があった場合，ネットワークアクセス制御用の具体化ポリシを生成する．受け取ったユーザIDとOSIDに割り当てられたロールを検索し，同一のロールが割り当てられたファイルを検索する．さらに，検索されたファイルを保存してあるサーバを抽出する．ユーザID，OSID，サーバの情報を基に，ネットワークアクセス制御用の具体化ポリシをサーバ用とアクセス制御層用に生成し，配布する．

また，アクセス制御層からゲストOSの停止が通知された場合には，ACLIDとVMIDに対応するユーザIDとOSIDを検索し，停止したゲストOSに関するポリシを削除する具体化ポリシを生成し，サーバアクセス制御に配布する．生存情報の通知が一定時間無かった場合には，通知がこないACLIDに対応するユーザIDとOSIDを保持していた情報から検索し，ポリシを削除する具体化ポリシを生成してサーバに対して配布する．これらの通信は，偽装したポリシの配布を防ぐために，正規のポリシマネージャであることを証明し，暗号化通信を行う．

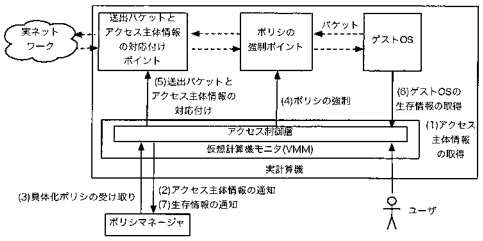

図 3: アクセス制御層の構造

4.3 アクセス制御層

アクセス制御層は，ID管理機能 [5] を介して，スマートカードよりユーザIDを取得し，ユーザの入力によりOSIDを取得し，VMIDを与えた仮想計算機を割り当てる．ユーザID，OSID，ACLID，VMIDをポリシマネージャに送り，具体化ポリシを受け取り，ポリシの強制をゲストOSのファイルイメージが仮想計算機に読み込まれる前に行うことで，確実にポリシの強制を行う．まず，ゲストOSの起動制御を行い，ゲストOSの起動が許可された場合，ユーザIDとOSIDごとにネットワークアクセス制御を行う．また，送出パケットに対してユーザIDとOSIDを関連づけ，暗号化通信をすることでサーバでのアクセス制御を可能にし，アクセス制御層の偽装を防ぐ．さらに，ゲストOSの生存情報を具体化ポリシに反映するため，ACLIDとVMIDごとのゲストOSの生存情報をポリシマネージャに対して一定時間ごとに通知する．偽装したアクセス制御用のポリシの配布を防ぐために，正規のアクセス制御層であることを証明し，ポリシマネージャと暗号化通信を行う．これらをまとめると図3のようになる．

5 実装

設計に基づき実装を行う．ここでは，実装を簡略化するためVMMを搭載している仮想計算機環境を導入した実計算機のIPアドレスは固定とし，ACLIDとして扱う．後述するポリシマネージャやアクセス制御層の偽装防止に用いるIPSecを実装したracoon2やStrong SwanではDHCPに対応しているため，実計算機にDHCPでIPアドレスが割り当てられていても構築でき，IPSecのAHヘッダをACLIDとして用いることもできる．さらに，実装を簡略化するためにゲストOSからの送出パケットに対してアクセス主体の情報を関連づけた後の暗号化は省略

する．

5.1 抽象化ポリシ

ネットワークアクセス制御のポリシを (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るユーザ 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から，XSLT を用いて具体化ポリシを生成し，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，専用の証明書を用いた IPSec の通信でサーバとアクセス制御層へのポリシ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と，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，具体化ポリシを受け取り，強制する．

ポリシの強制を行う際に，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の強制ポイントとし，ゲスト OS からの送出パケットへのアクセス主体の関連付けも行う．今回は，送出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を強制できるようにしたこ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, Enriquillo Valdez, Ramon Cacers, Ronald Prez, Stefan Berger, John Linwood Griffin, and Leendert van Doorn. Building a MAC-Based Security Architecture for Xen Open-Source Hypervisor. In *21 st Annual Computer Security Applications Conference (ACSAC 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.