L-010

VM セキュアプロセッサの提案
A Proposal of VM Secure Processor

千田 拓矢 † 宮永 瑞紀 † 山口 利恵 † 五島 正裕 ‡ 坂井 修一 †
Takuya Chida Mizuki Miyanaga Rie Shigetomi Yamaguchi Masahiro Goshima Shuichi Sakai

1. はじめに

近年，企業や個人のコンピュータから機密情報が漏洩する事件が増加し，深刻化している．情報が漏洩する主な原因はアクセスやマルウェアであり，このような不正な手段によってコンピュータから情報を読み取る行為は**タンパ**と呼ばれている．タンパは，ハードウェアを直接ロジカルアナライザなどで解析する**ハードウェアタンパ**とソフトウェアによって暗号や鍵情報を解析する**ソフトウェアタンパ**に分類できる．

特に，AmazonEC2 をはじめとする IaaS 型クラウドサービスではユーザの仮想マシン（VM）の情報がタンパによって漏洩する可能性が指摘されている．ユーザ VM の情報は遠隔地のクラウドサービスのデータセンターで管理されるため，ハードウェアタンパの危険性がある．また，クラウドサービスでは様々なコンピュータがネットワークでつながれているため，ソフトウェアタンパの危険性もある．そのような危険性が存在する一方で，クラウドサービスは導入コストが低く管理が簡単である．このように，クラウドサービスは従来の仮想サーバによるサービスと比べて大きなメリットがあるため，クラウドサービスでの情報漏洩への対策は急務となっている．

これらの問題を解決するために，我々はタンパへの対策として研究されている**セキュアプロセッサ**に着目し，セキュアプロセッサを VM に対応させることを考えた．セキュアプロセッサは通常のプロセッサに暗号化・復号やハッシュ値計測を行う独自の命令を追加し，プロセスをタンパから保護するプロセッサである．セキュアプロセッサには以前から，OS のプロセス管理に関する根幹部分の改変が必要であり既存の OS はそのまま利用できないという技術的な問題が存在し，これが実用化への大きな障害となっていた．一方，VM は OS やプロセスを含むコンピュータシステム全体であり，ハイパーバイザと呼ばれるソフトウェアによって管理される．ハイパーバイザがハードウェアとソフトウェアの中間で制御を行うため，仮想化環境ではハードウェアに依存せずソフトウェアを動かすことが出来る．したがって，セキュアプロセッサの保護対象をプロセスではなく VM に変更すれば，ハイパーバイザの改変が必要だが OS の改変は不要である．セキュアプロセッサを VM に対応させれば，クラウドサービスでのタンパ対策だけでなく，既存の OS を利用することが可能になり実用化に向けて大きな進歩になるといえる．

本稿では VM を保護対象とした **VM セキュアプロセッサ**の提案を行う．従来のセキュアプロセッサではプロセスを保護対象としていたのに対し，VM セキュアプロセッサは VM を保護対象とする．これにより，VM 管理に関してハイパーバイザを改変する必要が生じるが，VM 管理はプロセス管理よりも構造が単純であり，改変の負担は軽減される．また，直接 OS を改変することがないので既存の OS をそのまま導入することができる．

以下，2 章でセキュアプロセッサの概要と機能を述べ，3 章で提案手法である VM セキュアプロセッサの概要を述べる．4 章では VM の情報の機密性の確保のための機能について述べ，5 章では VM の動作するプラットフォームの真正性の確保のための認証手法について述べる．最後に本研究のまとめと今後の課題を述べる．

†東京大学 The University of Tokyo
‡国立情報学研究所 National Institute of Informatics

2. セキュアプロセッサ

セキュアプロセッサはプロセスの持つ機密情報をタンパから保護するプロセッサである．これまでに，XOM [6]・L-MSP [9]・Ascend [2]・AEGIS [8] などといったセキュアプロセッサが研究されている．

セキュアプロセッサの動作環境では，プロセッサ内部のみを信頼できる領域と想定し，プロセッサ外部（主記憶や OS など）は信頼できない領域であると想定している．プロセッサ内部では演算処理を行うため，データは平文でなければならない．したがって，プロセッサ内部の情報は不正にアクセスされないようにアクセス制御によって保護する．プロセッサ外部ではデータに対して処理は行わないため平文である必要がない．したがって，プロセッサ外部の情報は暗号化によって保護する．

2.1. セキュアプロセッサで想定されるタンパ

セキュアプロセッサはプロセスの機密情報をタンパから保護する．タンパには具体的に以下のようなものがある．

ハードウェアタンパ　ハードウェアタンパはハードウェアを利用してコンピュータを解析し，機密情報を不正に取得する行為である．

例えば，プロセッサとメモリや入出力機器の通信はバスを通して行われるので，コンピュータを分解し，ロジカルアナライザによってバスに流れる信号を直接読み取るような攻撃が想定される．また，揮発性のメモリを冷却し別のコンピュータに移して読み込むコールドブートアタック [3] や消費電力からプロセッサ内部の挙動を観測するサイドチャネルアタック [1]，電子顕微鏡によって半導体デバイスを解析する攻撃も想定される．

ソフトウェアタンパ　ソフトウェアタンパはソフトウェアを利用してコンピュータを解析し，機密情報を不正に取得する行為である．

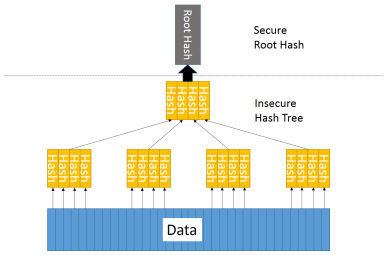

図 1: ハッシュツリーの構造

例えば，主記憶やストレージに保存された秘密鍵や平文のデータがリバースエンジニアリングによって解析されたり，不正な OS や不正なプロセスによって盗聴・改変される攻撃が想定される．

2.2. セキュアプロセッサの機能

セキュアプロセッサは制御部や演算部は通常のプロセッサと同じであるが，それに加えて暗号化処理に対応したメモリ管理ユニット (MMU) や暗号化・復号処理の演算部，ハッシュ値を計測・保存する機能を持つ．これらを利用して，アクセス制御と暗号化を行い，情報の機密性を保障する．また，完全性検証を行い，情報の完全性を保障する．

アクセス制御　セキュアプロセッサ内部で保護しなければならないのはレジスタとキャッシュ上のデータである．Lie らの提案する XOM [6] ではキャッシュラインには制御タグ，レジスタには制御ビットを用意することで読み書きを制御している．また，L-MSP [9] ではプロセスごとに識別子を用意し，読み込み時にキャッシュラインに識別子を書き込む．次にキャッシュラインにアクセスするときにはプロセスの識別子とキャッシュラインの識別子を照合する．

暗号化・復号　セキュアプロセッサは共通鍵方式によって暗号化・復号を行う．暗号化はプロセッサ外部にデータをストアする際に行われ，復号は外部からデータをロードする際に行われる．暗号化の鍵はプロセスごとに固有の鍵を生成し，プロセッサ内部で管理する．したがって，悪意のあるプロセスにメモリ上の情報を覗き見されたり，ハードウェアを直接解析されたとしても，プロセッサ以外に復号することは出来ないため情報が漏洩することはない．

完全性検証　Suh らの提案する Aegis [8] ではデータの完全性を検証するためにハッシュツリーを用いている．まず，キャッシュラインごとにハッシュ値を計測する．しかし，このままではハッシュ値の数が大きくプロセッサ内部で管理することが出来ない．そこで，計測したハッシュ値をいくつかまとめてさらにハッシュ値を計測する．これを繰り返すことで，キャッシュラインごとに計測したハッシュ値が一つのハッシュ値（Root Hash）に集約される．Aegis では Root Hash をプロセッサ内部で管理し，新たに計算したハッシュ値と照合することでデータの完全性を保障している．

3. VM セキュアプロセッサの概要

クラウドサービスでユーザ VM の情報へのタンパの対策として，セキュアプロセッサを VM に対応させた VM セキュアプロセッサを提案する．従来のセキュアプロセッサはプロセスをタンパから保護するために，OS のプロセス管理に関する根幹部分を改変する必要があり，それが実用化への大きな障害であった．一方，VM は OS やプロセスも含めたコンピュータシステム全体を仮想化したものである．したがって，VM を保護対象とすることでハイパーバイザの VM 管理の根幹部分を改変する必要が生じるが，VM 管理はプロセス管理よりも単純なため改変の負担は軽減される．さらに既存の OS を利用することが可能であり，実用化に向けて大きな進歩になるといえる．

3.1. VM セキュアプロセッサで想定されるタンパ

VM セキュアプロセッサでは従来のセキュアプロセッサと同様にハードウェアタンパとソフトウェアタンパを受けると想定する．さらに，仮想化環境特有のタンパとして，管理 VM からユーザ VM のメモリの情報を覗き見する VM Intorospection [7] や不正なハイパーバイザによるデータ改ざんも想定される．また，仮想化環境では遠隔地からプロセッサやメモリなどの物理ハードウェアに接続して VM を利用することが可能である．したがって，ユーザとプロセッサの動作するプラットフォームの間で生じる「中間者攻撃」や，不正なプロセッサによる「なりすまし」も想定される．

3.2. VM セキュアプロセッサの満たすべき条件

VM の情報をタンパから保護するとは以下の 3 つの条件を満たすことである．

- VM の情報の機密性の確保
- VM の情報の完全性の確保
- プラットフォームの真正性の確保

VM の情報の機密性の確保　VM セキュアプロセッサの動作環境では，プロセッサ内部のみを信頼できる領域と想定し，プロセッサ外部は信頼できない領域であると想定する．

まず，プロセッサ内部では演算処理を行うため，データは平文でなければならない．したがって，プロセッサ内部の情報は不正にアクセスされないようにアクセス制御によって保護する．一方，プロセッサ外部ではデータに対して処理は行わないため平文である必要がない．したがって，プロセッサ外部の情報は暗号化によって保護する．

VM の情報の完全性の確保　VM セキュアプロセッサはデータの改ざんを防ぐために，ハッシュ値によるデータの完全性検証を行う．

プラットフォームの真正性の確保　本稿では VM セキュアプロセッサが動作する動作環境をプラットフォームと定義する．遠隔から VM セキュアプロセッサを利用する場合，中間者攻撃やなりすましを防止するためにプラットフォームの真正性を検証する必要がある．そこで，公開鍵暗号とデジタル署名を用いた認証を行う．

認証には認証局としてユーザとプラットフォーム以外の第三者が必要である．そこで，VM セキュアプロセッサのメーカを認証局として信頼できるものと想定する．

4. VM セキュアプロセッサの機能

4.1. アクセス制御

VM セキュアプロセッサはプロセッサ内部のレジスタとキャッシュ上のデータの機密性を確保するためにアクセス制御を行う．仮想化環境では VM ごとにプロセッサの機能が割り当てられるため，異なる VM からユーザの VM のレジスタとキャッシュに不正にアクセスできないように制御する．VM セキュアプロセッサでは VM ごとに固有の ID を割り当てる．この ID を利用してキャッシュラインに制御タグを用意する．レジスタには制御ビットを用意してアクセス制御を行う．これにより，異なる VM が不正にレジスタとキャッシュにアクセスすることを防止することが出来る．

4.2. 暗号化・復号処理

VM セキュアプロセッサはプロセッサ外部のデータの機密性を確保するために暗号化･復号処理部を持つ．プロセッサ外部のデータは以下のように分類される．以下，それぞれのデータをどのように暗号化するか述べる．

- メインメモリ上のデータ
- ディスク上のデータ
- ネットワークカードで通信されるデータ

メインメモリ上のデータの暗号化・復号　メインメモリはプロセッサの外部に置かれるため，メモリ上の情報は暗号化によって機密性を確保する．VM セキュアプロセッサではプロセッサからメモリにデータを書き込む際に暗号化を行い，メモリからデータを読み込む際に復号を行う．暗号化・復号には共通鍵暗号方式を用いる．鍵は VM ごとに異なるものをプロセッサ内部で生成し，保管する．またデータの読み書きはキャッシュライン単位で行われるため，アクセス制御で用いた VM 固有のタグを利用してメモリに対するアクセス制御も行う．これにより，異なる VM やハイパーバイザが不正にメモリ上の情報を復号したりアクセスすることは出来ない．

ディスク上のデータの暗号化･復号　ディスクはデータを保管するための装置であるから，データをメモリからディスクに書き出す際は暗号化された状態で書き出す．ディスクからデータを読み込む際は暗号化されたデータをそのままメモリに読み込む．これにより，異なる VM やハイパーバイザ，異なるプロセッサで不正にディスク上の情報を復号することは出来ない．

ネットワークカードで通信されるデータの暗号化・復号　ネットワークでやり取りする情報は SSL/TLS によって暗号化される．このとき，暗号化は通信相手の公開鍵で行わなければならない．したがって，VM セキュアプロセッサでは通信相手の公開鍵によってプロセッサ内部で情報を暗号化し，ネットワークカードに送る．また，SSL/TLS では通信相手もこちらの公開鍵を知る必要がある．したがって，VM セキュアプロセッサは通信前にあらかじめ公開鍵を通知する．これにより，ネットワークで通信される情報の機密性は保たれる．

4.3. 鍵の生成・管理

VM セキュアプロセッサは鍵生成のための乱数生成器を持つ．また，生成した鍵管理のためのキャッシュメモリと製造時に埋め込まれた鍵を保管する ROM を持つ．これにより，プロセッサ内部で機密性を確保しながら鍵を生成・管理することが出来る．

4.4. ハッシュ値の計測・管理

VM セキュアプロセッサはデータの完全性検証のためのハッシュ値を計算･保管する部分を持つ．ハッシュ値は機密情報なので，プロセッサ内部に保存する．ディスク上のデータは改ざんの可能性があるため，ディスクに書き出す際にはハッシュ値を計測しプロセッサ内部に保管する．ディスクから読み込む際には再びハッシュ値を計測し，プロセッサ内部のものと照合する．これにより，ディスク上のデータの改ざんを検出することが出来る．

5. プラットフォームの認証

遠隔からプラットフォームの真正性を検証するために公開鍵暗号とデジタル署名を用いる．

5.1. 鍵の用意

認証に用いる公開鍵と秘密鍵のペアは以下の二つである．

- プロセッサメーカの公開鍵ペア
  $(PublicKey_m,\ PrivateKey_m)$
- プロセッサの公開鍵ペア
  $(PublicKey_p,\ PrivateKey_p)$

プロセッサメーカ用の公開鍵はメーカが管理する．また，プロセッサの公開鍵ペアはプロセッサ製造時にメーカによってプロセッサ内部の ROM に記録される．$PrivateKey_p$ はユーザにも知られないように厳重に管理する．

5.2. 証明書の用意

なりすましや中間者攻撃があった場合，プロセッサの公開鍵 $PublicKey_p$ が攻撃者によって偽装されるという危険性がある．したがって，$PublicKey_p$ の発行者がメーカであることを確認するために公開鍵証明書を用意する．公開鍵証明書は以下の手順で用意する．

1. プロセッサメーカは秘密鍵 $PrivateKey_m$ を用いて $PublicKey_p$ にデジタル署名を施す．
   これを *Certificate* とする
2. プロセッサメーカはプロセッサ製造時に公開鍵ペア（$PublicKey_p$，$PrivateKey_p$）と *Certificate* をプロセッサ内部の ROM に記録する

5.3. 認証の手順

用意した公開鍵ペアと公開鍵証明書を用いてプラットフォームの認証を行う．認証は以下の手順で行う．

1. ユーザはメーカから $PublicKey_m$ を得る

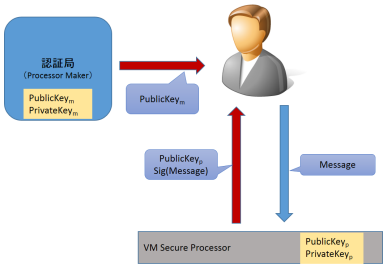

図 2: プラットフォームの認証

2. ユーザは適当な文字列 *Message* をプロセッサに送る
3. プロセッサは $PrivateKey_p$ を用いて *Message* にデジタル署名を施す (Sig(*Message*, $PrivateKey_p$) の作成)
4. プロセッサは署名の検証に用いる $PublicKey_p$ とその証明書 *Certificate*, Sig(*Message*, $PrivateKey_p$) をユーザに返す
5. ユーザは $PublicKey_m$ を用いて *Certificate* を検証する
6. ユーザは $PublicKey_p$ を用いて Sig(*Message*, $PrivateKey_p$) を検証する

以上の手順により，プロセッサの真正性を検証し，遠隔からプラットフォームの認証を行うことができる．

5.4. VM セキュアプロセッサのまとめ

VM セキュアプロセッサはプロセッサ内部の情報はアクセス制御によって保護し，プロセッサ外部の情報は暗号化によって保護する．また，ハッシュ値を計測・照合することによりディスクに保存した情報の完全性を検証する．さらに，公開鍵暗号とデジタル署名を用いて VM セキュアプロセッサの動作環境の真正性を検証する．

以上により，VM セキュアプロセッサは VM の情報の機密性・VM の情報の完全性・プラットフォームの真正性を確保することができる．したがって，VM セキュアプロセッサで想定されるタンパから VM の情報を保護することができる．

6. おわりに

本研究のまとめ　本稿では VM を保護対象とした VM セキュアプロセッサの提案を行った．まず，セキュアプロセッサの概要と既存の OS に改変が必要であることを述べ，その問題が仮想化環境の導入により解消されることを示した．次に，VM セキュアプロセッサの満たすべき条件を述べ，それを満たすための機能と手法を示した．VM セキュアプロセッサはクラウドサービスで VM の情報をタンパから保護することが出来る．

今後の課題　本研究室では FPGA 上で OpenRISC [5, 4] を用いた VM セキュアプロセッサの実装を進めている．具体的な仕様や構成に関しては宮永による「VM セキュアプロセッサの構成」に続く．

謝辞

本論文の研究の一部は，公益財団法人セコム科学技術振興財団の助成を受けたものである．

参考文献

[1] Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao, and Pankaj Rohatgi. The EM side—channel (s), pp. 29–45. Cryptographic Hardware and Embedded Systems-CHES 2002. Springer, 2003.

[2] Christopher W. Fletcher, Marten v. Dijk, and Srinivas Devadas. A secure processor architecture for encrypted computation on untrusted programs. pp. 3–8, 2012.

[3] J. A. Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten. Lest we remember: cold-boot attacks on encryption keys. *Communications of the ACM*, Vol. 52, No. 5, pp. 91–98, 2009.

[4] Damjan Lampret. Openrisc 1200 ip core specification. *September June*, 2001.

[5] Damjan Lampret, Chen-Min Chen, Marko Mlinar, Johan Rydberg, Matan Ziv-Av, Chris Ziomkowski, Greg McGary, Bob Gardner, Rohit Mathur, and Maria Bolado. Openrisc 1000 architecture manual. *Description of assembler mnemonics and other for OR1200*, 2003.

[6] David Lie, Chandramohan Thekkath, Mark Mitchell, Patrick Lincoln, Dan Boneh, John Mitchell, and Mark Horowitz. Architectural support for copy and tamper resistant software. *ACM SIGPLAN Notices*, Vol. 35, No. 11, pp. 168–177, 2000.

[7] Kara Nance, Brian Hay, and Matt Bishop. virtual machine introspection. *IEEE Computer Society*, 2008.

[8] G. E. Suh, Charles W. O'Donnell, and Srinivas Devadas. Aegis: A single-chip secure processor. *Information Security Technical Report*, Vol. 10, No. 2, pp. 63–73, 2005. ID: 5.

[9] 橋本幹生, 春木洋美, 川端健. オープンソース os と共存可能なセキュリティプロセッサ技術 (特集情報セキュリティ技術–安心・安全な社会へのソリューション). 東芝レビュー, Vol. 60, No. 6, pp. 44–47, 06/00 2005. ID: 2.