不正な管理者によるゲスト情報の窃盗・改変を防止するクラウドアーキテクチャ

村上 航規 †　山田 剛史 †　山口 利恵 †　五島 正裕 †　坂井 修一 †

† 東京大学
113-8656　東京都文京区本郷 7-3-1
{murakami, yamada, rie.yamaguchi, goshima, sakai}@mtl.t.u-tokyo.ac.jp

あらまし　我々は、IaaS 型のクラウドコン ピューティング環境において、ゲストユーザが所有する情報や演算内容の、秘密性および完全性を保証するアーキテクチャを提案する。従来のクラウドアーキテクチャでは、ゲストの情報の保護はプロバイダが提示するセキュリティポリシーによってのみ保証されていた。従って、プロバイダのオペレータに悪意があれば、システム管理を行うための権限を悪用してゲスト情報の窃盗を行うことが可能であった。本手法では、既存ハイパーバイザのメモリ管理機能に改変を加えてメモリ情報を窃盗できないようにし、ゲスト情報を悪意あるオペレータや他のゲストから保護する。

A Cloud Architecture for Protectiong Guest's Information against Attacks of Malicious Operator

Koki Murakami †　Tsuyoshi Yamada †　Rie Shigetomi YAMAGUCHI †
Masahiro Goshima †　Shuichi Sakai†

†The University of Tokyo.
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, JAPAN
{murakami, yamada, rie.yamaguchi, goshima, sakai}@mtl.t.u-tokyo.ac.jp

Abstract　We introduce a novel cloud computing architecture which provides privacy and integrity for guest's information and computation. In conventional cloud architecture, a security policy proposed by a provider only ensured the protection of guest's information. This enabled malicious operators to steal or modify guest's information. Our method protect guest's information on memory with novel memory management function of hypervisor from malicious operators or guests.

1　はじめに

1.1　背景

クラウドコンピューティングは、ネットワーク、特にインターネットをベースとして、コンピュータリソースやアプリケーションをオンデマンドに利用できるようにするモデルである。このモデルにおいて利用者は、最小限の管理コストと労力で必要な処理能力・アプリケーションを利用できる。NIST は、クラウドコンピューティングにおける 3 つのサービスモデルとして、Software as a Service(SaaS)、Platform as a Service(PaaS)、Infrastructure as a Service(IaaS) を定義している [1]。

従来のITモデルに比べ、クラウドコンピューティングは多くの利点を持つ。一方で、利用者にとっては採用に際していくつかの懸念が存在する。特に顧客が企業の場合、セキュリティ懸念は最大の障壁となり、それを不安視してクラウドサービスの利用を留まる例もある。Intelが IT のプロフェッショナルに対して2012年に行った調査 [2] によれば、企業はパブリッククラウドに関連するセキュリティ違反に、既存のITインフラにおける経験と比べて28%多く直面している。健康や金融に関連する情報を取り扱う企業では、法的な要件がパブリッククラウドのプロバイダと比べより厳しいことが多く、両者の不整合を83%の IT プロフェッショナルが懸念している。

クラウドコンピューティングのプロバイダは、顧客のデータや演算情報が漏洩しないことをセキュリティポリシーによって保証している。しかし、実際には2009年にAmazonのストレージサービスが停止させられ、またGoogle Docsがセキュリティ脆弱性を利用されて顧客データの情報漏洩が発生している。

クラウドコンピューティングにおけるセキュリティ懸念は、脆弱性を利用した外部からの攻撃や、管理者の過失だけではない。クラウドサービスやデータセンターの管理者は、その権限を利用して不正に顧客のデータを窃盗・改変することができる。プロバイダが団体として掲げているセキュリティポリシーを、通常管理者は遵守するものと考えられているが、管理者に悪意のある者が就任すればその前提は崩れる。

1.2 提案手法

顧客がクラウドサービスのセキュリティを、セキュリティポリシーによってではなく機械的・電子的に保証された方法で信頼できるならば、これらのセキュリティ懸念は解決される。我々は、IaaS型のクラウドコンピューティング環境において、例えクラウドの管理者が不正を行おうとしても、顧客のメモリ上データとストレージ上データの秘密性を保証するアーキテクチャを提案する。

クラウドコンピューティング環境を実現するために、通常は仮想化の技術が用いられる。仮想マシンを管理するハイパーバイザは、固有のページテーブルを持つ。これは実際のメモリ領域を変換し、その結果をゲストに対してあたかも真のメモリ領域であるかのように通知するためのものである。

我々のアーキテクチャは、この「ハイパーバイザが持つページテーブル」に対し、固有のアクセス権限ビットを付加することで、悪意あるオペレータからゲストのメモリ上データを保護することを主眼とする。また、メモリ上データがストレージへスワップアウトする際には、それらはゲストごとに固有の鍵で暗号化される。これによって、ゲストのメモリ上データとストレージ上データを保護する。

新たなアクセス権限ビットの付加によって、不正な管理者による、顧客のメモリ上・ストレージ上データを窃盗するような命令はアクセス権限違反となり、実行できない。すなわち、カーネル設計に我々のアーキテクチャを採用したハイパーバイザが、クラウドのサービスプロバイダに導入されている限り、顧客のデータはたとえメモリ上にある時でも、不正な管理者に窃盗されない。

1.3 構成

本稿は以下の構成からなる。第2節で、我々の研究の動機を述べるとともに、クラウドコンピューティングにおけるセキュリティについての関連研究を挙げる。第3節で、提案手法における仮定を示す。第4節で、我々のアーキテクチャの設計について詳細を述べる。第5節で、我々のアーキテクチャの安全性を検証する。第6節でまとめを行う。

2 研究の動機と関連研究

2.1 研究の動機

我々の目標は、不正な管理者からゲストのメモリ上データを保護することである。従来のクラウドアーキテクチャは、管理者用OSにその

機能を果たさせるための強い権限を与えている。ストレージ上データは暗号化保存する方法が既に実用化されている [13] 一方で、メモリ上のデータは平文であり、不正な管理者は権限を用いて平文を窃盗することができる。従って、不正な管理者は特権を利用して (顧客に渡された) 仮想マシンに割り当てられた物理メモリの内容を窃盗・改変することができる。

メモリ上データの窃盗は管理者個人に起因するが、プロバイダが情報を窃盗する意図をもって管理ソフトウェアを用いたり、あるいはハイパーバイザ自体に不正な改変を加えない保証もない。

プロバイダの信頼性に疑問が持たれる一方で、企業へのクラウドサービスの導入は魅力的であると考えられている。金融や医療といった、求められるセキュリティレベルが高い業種でも導入を求める声が存在する。例えば、東日本大震災で被災地の病院のサーバに保管されていた電子カルテが津波でサーバごと消失し、その結果災害対策として電子カルテのクラウド化が提言された [19]。

単に管理者が顧客のデータに触れられないようにするには、管理インタフェースの機能を、仮想マシンの生成・削除などの最低限に制限すればよい。しかし、これは柔軟な管理を困難にする。顧客の要望に合わせてハードウェア整備などを容易に行えなくなる恐れもあり、クラウドサービスの魅力を失わせることになる。

我々のアーキテクチャは、クラウドサービスの管理者が業務を遂行するのに必要な権限の保持と、ゲスト情報を保護するための権限の制限を両立させる。これは不正な管理者だけでなく、悪意あるゲストがハイパーバイザの脆弱性を利用して管理インタフェースを乗っ取り、それを通して攻撃を試みた場合にも効果を発揮する。この手法によって、企業が業務にクラウドを導入するセキュリティ懸念のうちの一つである、不正な管理者によるデータの窃盗・改変が不可能になる。

我々の手法は、ハイパーバイザがそれを採用しており、かつ不正な改変を受けていない際に情報の保護を行うことができる。すなわち、ハイパーバイザがこれらを満たすことを顧客が確認する方法が必要である。この認証には、Intel Trusted Execution Technology(TXT)[21] などの手法を用いる。

2.2 関連研究

仮想化環境において、悪意あるゲストがハイパーバイザの脆弱性を利用して特権を取得し、ハイパーバイザや他のゲストに不正な動作をさせたり、情報を窃盗する危険が問題視されている。これを防ぐためのものが広く研究されている [3, 4, 5]。これらの研究で提案された手法は、クラウドコンピューティング環境において、不正な管理者からゲストのデータを保護することは不可能である。

不正な管理者によるゲストへの攻撃を脅威とみなしている研究も存在する。Liang らが提唱した CertiKOS[6] は、一般的な認証の手法を拡張して、クラウドコンピューティングにおける情報漏洩を防ぐためのアーキテクチャである。CertiKOS はゲストに対してハードウェアリソースを割り当てる機構をハイパーバイザ内に隠ぺいし、管理 OS には持たせていない。割り当てられたリソースにはタグ付けがなされ、所有レコードもまたハイパーバイザが保持する。CertiKOS はタグ付けによってリソース割り当てをするが、CPU による処理能力の割り当てはコアごとにタグを付けることによって行われる。すなわち各々の仮想マシンが 1 つのコアを占有し、演算の時分割を行えない。

NoHype[7] は CPU アーキテクチャを拡張してハイパーバイザ層を取り除き、ゲスト OS を保護する。しかし、メモリ空間の仮想化においてアドレス変換を行うような、通常のハイパーバイザに備えられる機能がソフトウェア層に存在しない。そのような機能は専用の CPU に実装されることを前提としており、現在の商用 CPU 上では利用できない。

本稿では顧客のメモリ上データの保護に着目するが、一方でストレージ上データを保護する手法も数多く存在する。Kamara and Lauter[22] は、顧客が完全にはクラウドサービスプロバイダを信頼していない状況で、暗号化によって顧

客のストレージ上データを保護する手法を示した。Kamara and Lauter はメモリ上データの保護については考慮していない。

3 仮定

3.1 環境

我々は、プロバイダが IaaS 型のサービスを提供しており、データセンターをストレージとして利用する形態で、顧客がそのサービスを利用していることを仮定する。管理者は職務上必要な権限を与えられているが、プロバイダが提示するセキュリティポリシーには必ずしも従わない。プロバイダはクラウド環境の基盤として、Type1 ハイパーバイザ型の仮想化を行っているものとする。

顧客はその環境で通常行える処理を行い、重要な情報を含むデータをストレージに置くことがある (クラウド環境を企業で利用する場合、機密性の高い情報を一切ストレージに置かないようにすることは困難である)。ストレージへデータが保存される際には、それをゲストごとに固有の鍵で暗号化する機能がハイパーバイザに実装されているものとする。この仮定によって、顧客情報を保護する必要があるのは、それがメモリ上にある時のみになる。

3.2 提案手法における仮想化アーキテクチャ

仮想化は、コンピュータの物理リソースを抽象化する技術である。仮想化によって、複数のコンピュータを単一の高性能なマシンとして運用したり、複数の仮想マシンを構築して異なる処理をさせることができる。

仮想化を実現するためのホストとなる制御プログラムは、ハイパーバイザや仮想マシンモニタとよばれ、ゲストである仮想マシンを制御する。ハイパーバイザは、その形式によって Type 1(ベアメタル型) と Type 2(ホスト型) に大別される (図 1)。Type 1 ハイパーバイザはハードウェア上で直接動作し、すべてのゲスト OS はハイパーバイザの上で動作する。Type 2 ハイパーバイザは、ハードウェア上で動くホスト OS の上でアプリケーションとして稼働する。

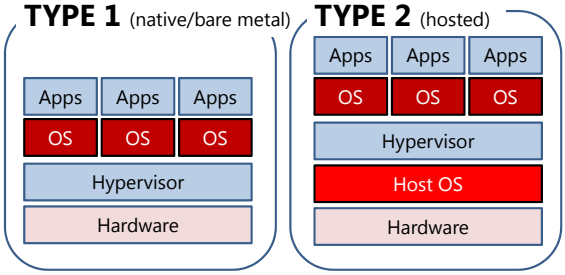

図 1: ハイパーバイザの分類

Type 1 ハイパーバイザは、ゲスト OS に対して完全にシミュレーションされた仮想環境のインタフェースを提供する完全仮想化と、専用に改変されたゲスト OS が必要な準仮想化に分けられる。Type 1 ハイパーバイザはゲストの一部のクリティカルな処理に対して割り込むが、そうでない処理は物理環境と同様に実行させる。従って、ゲスト OS の動作はより高速になる。

Type 2 ハイパーバイザは、仮想マシンがソフトウェアレベルで生成されるため、Type 1 に比べ処理速度が遅い。一方で導入が手軽なため、個人用途の仮想化で広く用いられる。

Type 1 ハイパーバイザの例としては、Xen[8]、VMware ESX Server[12]、Hyper-V[14]、Linux KVM[15] などが存在する。Type 2 ハイパーバイザの例としては、VMware Workstation[16]、VirtualBOX[17]、QEMU[18] などが存在する。

我々のアーキテクチャは、Type 1 型ハイパーバイザを採用することを前提とする。パブリッククラウドでは、プロバイダは顧客にある程度の演算能力を持つ仮想マシンを貸与するため、動作が高速な Type 1 型が通常用いられる。

4 カーネル設計

我々のアーキテクチャの課題は、クラウドの管理者がその業務に支障を来すことがないようにしながら、どのようにゲストのデータを保護するか、ということである。顧客は、従来のクラウド環境と全く同様にゲスト OS を運用でき

るべきであり、また管理者は仮想マシンの管理に必要なすべての権限を持ちながら、同時に顧客データの閲覧は制限されなければならない。

本節では、これを実現するための設計を示す。

4.1 管理インタフェース

クラウドコンピューティングにおいて、プロバイダは最低限でも、仮想マシンの追加や削除を行うためのインタフェースを備えている必要がある。Type 1 ハイパーバイザではホスト OS が存在しないため、このインタフェースは特定の仮想マシンに備えられる。同様のアーキテクチャを採用するハイパーバイザには Xen dom0[8] などが存在する。Xen dom0 では、特権を持つ管理者は、追加されたすべてのゲスト OS にログインすることができ、すべての物理ハードウェアへの直接のアクセス権を持つ。

我々のアーキテクチャでは、悪意ある管理者によるゲスト情報の窃盗を防ぐため、管理インタフェースを通した物理ハードウェアへのアクセス権限を、ハイパーバイザ層が制限する。このために、ハイパーバイザが持つ独自のページテーブルであるシャドウページテーブルのエントリ形式に改変を加え、新たな情報を追加する。

4.2 ページング

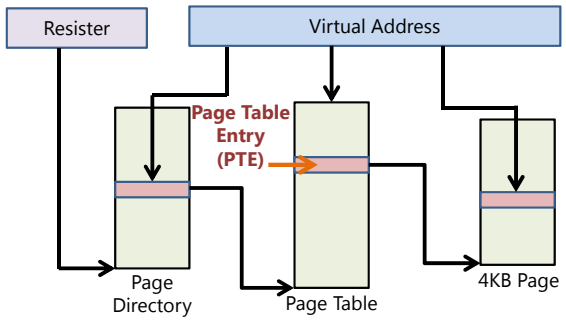

図 2: ページングにおけるアドレス変換

ページングとは、OS が記憶装置をページと呼ばれる小さな単位に分割して、各プロセスに対して割り当てを行うメカニズムである。プロセスはそのプロセスに固有の仮想アドレス空間を参照し、仮想アドレスを物理アドレス空間に変換する対応付けはページテーブルと呼ばれる構造体が行う。ページテーブルは OS によって管理される。

仮想アドレスから物理アドレスへの変換 (図 2) は、プロセス動作時にオーバヘッドを発生する原因となる。このオーバヘッドを低減するため、近年の CPU のメモリ管理ユニットには TLB(トランスレーション・ルックアサイド・バッファ) と呼ばれるキャッシュが実装されている。一度仮想アドレスから物理アドレスへの変換が行われると、その変換情報が TLB に記録され、次回以降のメモリアクセスを高速化する。CPU によっては、ページテーブルエントリの形式がメモリ管理ユニットによって定義されているものがある。そのような CPU では、プロセスがページテーブルエントリを参照するときにメモリ管理ユニットがアクセス権限をチェックする。高い権限レベルのリングに対してのみアクセスを許可するようなページテーブルエントリを、CPU の実行状態が低い権限レベルのリングにあるときに参照しようとすると、メモリ管理ユニットがアクセス保護違反を発生する。

4.3 仮想化とシャドウページテーブル

完全仮想化の環境においては、ゲスト OS が物理環境との差を考慮することなく動作する。しかし、ゲスト OS が自由に物理メモリ領域にアクセスすると、ホストや他のゲストの動作に影響を与えてしまう。これを解決するために、一般的な仮想化環境ではシャドウページテーブルと呼ばれる仕組みが利用されている。

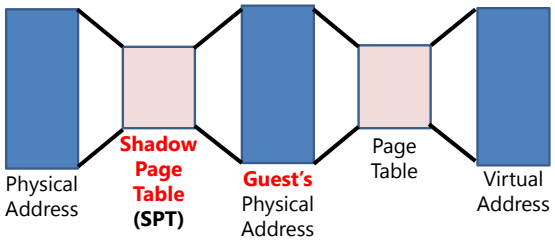

図 3: シャドウページテーブルの概念図。

アプリケーションが利用する仮想アドレス空

間を、物理アドレス空間に変換する通常のページテーブルは、OS によって管理される。一方でシャドウページテーブルは、真の物理アドレスを、ゲスト OS にとっての物理アドレスに変換するためのものであり、ハイパーバイザによって管理される。ゲスト OS 上のアプリケーションがある仮想アドレスにアクセスしようとすると、ゲスト OS はページテーブルを通して、自身が物理アドレスだと思い込んでいるアドレスへアクセスを試みる。ハイパーバイザはそれを、シャドウページテーブルのエントリを通して再度変換し、真の物理アドレスへと変換する。変換結果が TLB にキャッシュされ、次回以降のアクセスを高速化する。

4.4 シャドウページテーブルエントリへの権限ビット付加

近年の OS は、ページテーブルエントリに仮想化環境での 2 つのページテーブルを通したアドレス変換において、そのアクセス権限チェックを 2 回行うことができる。すなわち、ゲスト OS によるページテーブルエントリを通じたアクセス権限チェックと、ハイパーバイザによるシャドウページテーブルエントリを通じたアクセス権限チェックである。

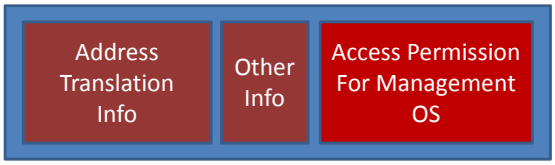

図 4: 我々のアーキテクチャにおけるシャドウページテーブルエントリの構造

我々のアーキテクチャでは、シャドウページテーブルエントリに「管理インタフェースがその物理アドレスにアクセスできるかどうか」を示すビットを付加する (図 4)。管理インタフェースでない (顧客に貸し出される) 仮想マシンを、ハイパーバイザが生成するよう指示されたとき、ハイパーバイザは物理メモリを割り当てる。割り当てられた範囲の物理アドレスは、シャドウページテーブルを通して仮想マシンに真の物理アドレスとして渡される。そのシャドウページテーブルでは、ハイパーバイザが先のアクセス権ビットを真にする。

ゲスト OS がシャドウページテーブルエントリを通して特定の物理アドレスにアクセスを試みるとき、そのゲスト OS が管理インタフェースであれば、ハイパーバイザは先のビットをチェックする。そのビットが真であった場合、当該アクセスを禁止する。

この手法によって、顧客の機密データがメモリ上にある時に、不正な管理者によってそれを窃盗されることがなくなる。管理者による命令そのものが無効になるのではなく、権限を越える命令を実行しようとすると、単にパーミッションエラーが起こる。

このようなアクセス制限を施しても、ゲスト OS は完全に物理環境と同様に運用でき、またハイパーバイザそのものの権限も何ら制限されない。従って、アクセス制限によってゲスト OS やクラウド環境の可用性が損なわれることはない。

4.5 ストレージの暗号化

我々のアーキテクチャは、ゲストのデータがメモリ上に存在する時には、管理ソフトウェアによる当該物理アドレスへのアクセスを禁止することで秘密性を保証している。しかし、実際にメモリ上に置かれているデータは平文であるので、ストレージへスワップアウトする際は暗号化によって保持しなければならない。

仮想化の段階でストレージがハイパーバイザによって抽象化されているため、ハイパーバイザがゲスト OS のストレージへのアクセスすべてに介入することは容易である。VMware[13] を用いて、ストレージを暗号化している例が既に利用されている。

4.6 カーネル認証

我々の手法は、それがハイパーバイザに採用されている限り、顧客データを保護する。ゆえに、プロバイダが我々の手法を取り入れたハイ

パーバイザを採用していることを、顧客の側から確認する手法が必要である。この手法には、Intel Trusted Execution Technology(TXT)[21]などの、ハードウェア機能を利用したハイパーバイザ認証を用いる。

5 安全性

悪意あるオペレータが顧客データの窃盗・改変を試みる場合、管理用仮想マシンで攻撃用のプロセスを生成するか、不正な OS から直接物理アドレスへアクセスを行う。ここで、仮想マシンが物理アドレスだと信じてアクセスするアドレスは、実際にはハイパーバイザが変換した仮想アドレスである。

ハイパーバイザは、仮想マシンによるすべてのメモリアクセスに割り込み、アドレス変換を行う。従って管理インタフェースが仮想マシンである限り、ハイパーバイザによるアクセス権限チェックを回避できない。

従って、顧客用の仮想マシンにメモリが割り当てられたときに、新たな権限ビットを設定する機能が不正な改変を受けない限り、我々の手法は顧客のメモリ上データを保護できる。

6 結論

本稿では、IaaS 型のクラウドサービスにおいて、不正な管理者から顧客の情報を保護するための新たなアーキテクチャを提案した。クラウドコンピューティングは、個人のみならず企業に対しても利便性をもたらすことが期待されるコンピューティングモデルである。しかし、従来のモデルに比べ未だ多いと考えられているセキュリティ懸念が、秘匿性の高い情報を取り扱う企業への導入を阻んでいた。我々のアーキテクチャは、問題視されながらも従来それを防ぐ方法があまり提案されなかった、不正な管理者からの保護を主眼に置き、この問題を解決する。本稿で提案した手法によって、クラウドコンピューティングにおける主要なセキュリティ課題の一つが解決される。

参考文献

[1] Peter M. and Timothy G. The NIST Definition of Cloud Computing. NIST Special Publication 800-145, 2011.

[2] Intel Peer Research What's Holding Back the Cloud? Intel IT Center, 2012.

[3] Sailer R., Jaeger T., Valdez E., Caceres R., Perez R., Berger S., Griffin J. L. and Doorn L. v. Building a MAC-Based Security Architecture for the Xen Open-Source Hypervisor. In ACSAC, 2005, 276–285.

[4] Wang Z. and Jiang X. HyperSafe: A Lightweight Approach to Provide Lifetime Hypervisor Control-Flow Integrity. In SP, 2010, 380–395.

[5] Azab A. M., Ning P., Wang Z., Jiang X., Zhang X. and Skalsky N. C. HyperSentry: enabling stealthy in-context measurement of hypervisor integrity. In CCS, 2010, 38–49.

[6] Liang G., Alexander V., Bryan F., Zhong S. and David C., CertiKOS: A Certified Kernel for Secure Cloud Computing. In APSys, 2011, No. 3.

[7] Keller E., Szefer J., Rexford J. and Lee R. B. Nohype: Virtualized cloud infrastructure without the virtualization. In ISCA, 2010, 350–361.

[8] P. Barham, B.Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebar, I. Pratt, and A. War

eld. Xen and the art of virtualization. In SOSP, 2003.

[9] Intel. Intel 64 and IA-32 Architectures Software Developer's Manual, Vol. 2, 2013.

[10] Intel. Intel Virtualization Technology FlexMigration: Application Note, 2012.

[11] Advanced Micro Devices. AMD64 Architecture Programmer's Manual Volume 2: System Programming, 3.22 edition, September 2012.

[12] C.A. Waldspurger. Memory resource management in vmware esx server. SIGOPS Operating Systems Review, 36(SI): 181–194, 2002.

[13] VMware Inc. Performance Evaluation of Intel EPT Hardware Assist, 2009.

[14] Microsoft. Hypervisor Functional Specification. 2008.

[15] A. Shah. Kernel-based virtualization with KVM, Linux Magazine, Issue 86, 37–39, 2008.

[16] VMware Inc. Workstation User's Manual, September 2007.

[17] Oracle Corp. Oracle VM VirtualBox User Manual, Ver. 4.2.12, 2013.

[18] F. Bellard, QEMU, a Fast and Portable Dynamic Translator, In Usenix annual technical conference, 42–46, 2005.

[19] 総務省, 平成24年度版 情報通信白書, 2012.

[20] T. C. Group. Trusted platform module (TPM) Main - Part 1 Design Principles, 2007.

[21] Intel. Intel Trusted Execution Technology: White Paper, 2012.

[22] Seny K., Kristin L. Cryptographic Cloud Storage, Financial Cryptography and Data Security, 136–149, Springer Berlin Heidelberg, 2010.