L-040

仮想計算機 Xen における vTPM のアクセス制御の改良

森川浩明†, 榎原博之‡, 大西克実†, 中野秀男†,
Hiroaki Morikawa†, Hiroyuki Ebara‡, Katsumi Onishi†, Hideo Nakano†

1 はじめに

近年，計算機の高機能大容量化が進み，効率的に計算資源を利用する手段として，仮想計算機による計算資源の仮想化・統合がおこなわれている．しかし，仮想計算機を提供するホスト OS により，仮想計算機が攻撃にさらされるケースがある．その解決方法の 1 つとして Trusted Platform Module(TPM) の利用が提案されている．しかし，仮想計算機上から TPM にアクセスする virtual TPM(vTPM)[1] において，中間者攻撃やリプレイ攻撃の糸口となる TPM-仮想計算機間での認証パケットが増大し，仮想計算機内の個人情報などが漏洩する危険性がある．

本研究では，この問題を解決するために仮想計算ソフト Xen[2] の TPM アクセスライブラリの改良をおこなう．提案ライブラリでは，プロセス終了後もアクセス制御をおこなうアクセス制御機能と TPM への同時アクセスを実現する共有機能を実装することで，安全で利便性の高いユーザ保護を確立する．更に，これらの機能の性能を実験により評価する．

2 virtual Trusted Platform Module(vTPM)

2.1 Trusted Platform Module(TPM)

TPM は，暗号鍵の情報などを直接メモリ空間に配置しないための手段として，マザーボード上に暗号化と各種データの保存・管理をおこなう機能を実装したチップセットであり，以下の 3 つの機能を持つ．

- 耐タンパ性が保障された不揮発領域
- 公開鍵による暗号化，署名
- 乱数発生器

これらの機能により，TPM を利用した計算機ではそのメモリ上に暗号化鍵を載せる必要が無いため，安全性を保つことができる．

2.2 vTPM

vTPM は，複数の仮想計算機が TPM の領域に対してアクセスするために仮想計算ソフトから提供される仮想のデバイスドライバである．Xen では，仮想計算機に TPM を提供する手段として，vTPM のアドレスを実際の TPM のアドレスに変換しアクセス制御をおこなう TPM managerd と，vTPM の内容を他のノードへ移行するための TPM migrationd の 2 つのプログラムを実装している．

2.3 TPM の問題点

TPM の利用では，認証プロトコルで TPM プロセスを認証しなければならない．OSAP と呼ばれる認証プロトコルでは，1 つの認証情報で異なる TPM 命令 (例：鍵の読込と書込など) を利用できるため，TPM プロセス・TPM 間での認証情報を取得することで，リプレイ攻撃できるという問題 [3] がある．

また，TPM プロセスが TPM の利用領域を占有するため，TPM への同時アクセスができず，デッドロックが発生する問題がある．

特に，仮想計算機を用いた場合，TPM の利用や TPM プロセス・TPM 間での通信が多くなるため，これらの問題の影響が大きくなる．

3 提案ライブラリ

3.1 提案ライブラリの概略

本研究では，2.3 節で述べた問題に対し，Xen の TPM アクセスを制御する TPM managerd を改良することで解決を図る．第一に，他の仮想計算機に対してリプレイ攻撃をおこなわせない手段として，仮想計算機からアクセスできない識別子による認証機能を付与する．加えて，ホスト OS から TPM 領域へ代理アクセスすることで，同じ TPM 領域へ複数の TPM プロセスが同時アクセスできるようにする．

† 大阪市立大学創造都市研究科
‡ 関西大学システム理工学部

3.2 不揮発領域へのアクセス制御機能

本節では，リプレイ攻撃の問題を解決するために，TPM managerd がプロセス終了後も認証情報を必要とするアクセス制御プログラムを提供する．

Step 1 *DomainU* は利用する不揮発領域を指定し，TPM 使用許可を求める

Step 2 TPM managerd は指定した領域が利用されていなければ，*DomainU* の Domain ID と領域情報，乱数 r を暗号化し，秘密情報 $Secret$ として *DomainU* に送信する

Step 3 *DomainU* ではアクセス制御開始メッセージと $Secret$ を TPM managerd に送信する

Step 4 $Secret$ を復号し，Domain ID などを取り出すことができれば TPM の利用を開始させる

Domain ID は Domain 0 でのみ呼び出し可能な各 Domain U に与えられる ID である．このため，Domain U での $Secret$ の偽造は困難である．

3.3 TPM 領域の共有機能

現在，不揮発領域の内容は計算機内の複数の仮想計算機から同時に参照することができない．しかし，限られた計算資源への同一情報の配置は効率が悪い．

このため，本研究では Domain U からの TPM 共有メッセージを受け取ると，Domain 0 のプロセスが TPM へのアクセスを代替し，Domain U からの複数プロセスの同時アクセスを実現する．ただし，共有中の不揮発領域は書込禁止としている．以下では，$DomainU_i$ が TPM を利用する際の動作を説明する．

Step 1 $DomainU_i$ が不揮発領域にアクセスを開始するこのとき，アクセス制御用情報 $Data_i$ をブラインド署名用乱数 R_i で暗号化しておく

Step 2 TPM managerd は $Data_i$ に対して署名し，その内容を $DomainU_i$ に返却する

Step 3 $DomainU_i$ は乱数 R_i と署名データ $Sign$ から共有用データ $Sync$ を受け取り，共有成功メッセージを TPM managerd に送信する

Step 4 TPM managerd は，Domain U-TPM 間の通信の仲介となる．このとき，共有情報 Sign と $DomainU_i$ の ID を所有者として設定する

共有側 ($DomainU_j$) では登録側と同様の動作をするが，Step2，3 で $Sync$ を $DomainU_i$ に要求し，DH 法で $Data_i$ を共有する．

4 実験結果

4.1 実験環境

本章では，3 章で提案したプログラムを PC (Pentium 4 2.6GHz，256MByte RAM) に実装することにより，性能を評価する．

4.2 アクセス制御機能の計測

アクセス制御では，TPM へのアクセス開始，終了，鍵情報の更新に必要な処理 (TPM アクセス) が初回で約 630ms，2 回目以降では約 380ms かかった．これは，通常の TPM managerd の約 32ms に対して，公開鍵で暗号化をおこなう時間を加えたものとほぼ等しくなっている．しかし，TPM へのアクセスは頻度が低いため，他の処理への影響は小さいと思われる．

4.3 共有機能の計測

共有機能の計測では，TPM アクセスに登録側で約 390ms，共有側で約 440ms の速度低下が見られた．特に，共有側では DH 鍵共有法の処理が必要になるため，時間・計算資源ともに増加した．

5 おわりに

本研究では，TPM を安全に利用するためのライブラリの開発をおこなった．TPM ライブラリを改良することで，完全仮想化であれば計算内容が外部に漏洩する危険性が減ると考えられ，準仮想化であっても計算機を Secure Boot することで不正なプログラムの侵入を抑制できる．

参考文献

[1] S. Berger, R. Caceres, K. A. Goldman, R. Perez, R. Sailer and L. van Doorn: vTPM: virtualizing the trusted platform module, USENIX-SS' 06, 2006.

[2] Xen: http://www.xen.org/ .

[3] D. Bruschi, L. Cavallaro, A. Lanzi and M. Monga: Replay Attack in TCG Specification and Solution, ACSAC 2005, 2005.