1W-2

インベントリ証明書によるアクセス制御システムの設計と実装

脇田　知彦[†]　福田　洋治[‡]　毛利　公美[††]　白石　善明[†]　野口　亮司[†††]
名古屋工業大学[†]　愛知教育大学[‡]　岐阜大学[††]　(株) 豊通シスコム[†††]

1. はじめに

クラウドの導入により，企業情報システムが端末や場所を問わずインターネット経由で利用できるようになっている[1]．この流れを受けて私用端末の業務利用を認める BYOC という考え方が現れているが[2]，私用端末での利用はキュリティリスクを高めるため，企業は利用する端末に関するポリシーを定めてアクセス制御を行うべきである．

我々はインベントリの証明書を用いたアクセス制御モデルを論文[3]で提案している．ここでのインベントリは端末のハードウェア・ソフトウェアとその設定情報を意味する．このモデルでは端末で認証と署名を行う必要があり，我々は私用端末でも秘密鍵を安全に管理して認証と署名を行える TPM の公開鍵を管理するシステム TKMS を開発した[4]．本稿では TKMS を利用したインベントリ証明書によるアクセス制御システム IB-ACS の設計と実装を行う．

2. インベントリ証明書によるアクセス制御

論文[3]で提案しているアクセス制御モデルを図に示す．このモデルは 3 つのエンティティで構成される．

企業情報システム利用端末（CSR，Corporate System Requester）
企業情報システムを利用する端末．なお，CSR の収集するインベントリは正しいものとする．

端末構成保証局（ICA，Inventory Credential Authority）
CSR のインベントリに対してインベントリ証明書（以降，証明書という）を発行する信頼できる第三者機関．

企業情報システム提供者（CSP，Corporate System Provider）
CSR に企業情報システムを提供する企業など．CSR の提示した証明書に従い，システム利用の可否を決定する．

本モデルの利点はインベントリが証明書という日時，内容，端末が ICA に保証された形で記録されるため，現在の端末の状態だけでなくこれまでどのように運用管理されてきたかという過程を元にアクセスできることと，アクセス制御時には確認しなかったインベントリも後から追加で確認できることが挙げられる．

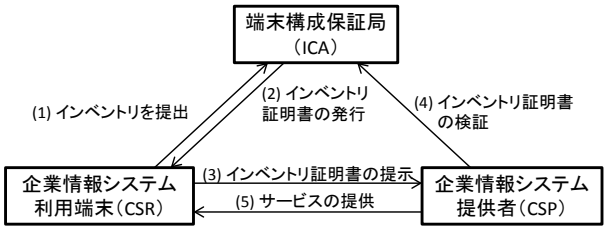

図 1　インベントリ証明書によるアクセス制御モデル

Design and Implementation of Inventory Credential-Based Access Control System

† Tomohiko Wakita and Yoshiaki Shiraishi・Nagoya Institute of Technology
‡ Youji Fukuta・Aichi University of Education
†† Masami Mohri・Gifu University
††† Ryoji Noguchi・Toyotsu Syscom Corp.

3. TPM 公開鍵管理システム

CSR が私用端末である場合，認証と署名を行うときに必要な秘密鍵をどのように保管するかが問題となる．そこで我々は TPM（Trusted Platform Module）[5]というハードウェアに鍵を保管して認証や署名を行うことを検討した．TPM は TCG（Trusted Computing Group）[6]が仕様を策定しているセキュリティチップであり，端末のセキュリティを高める機能を持っている．その機能の 1 つに鍵を保管する機能があり，鍵を TPM 外部に取り出すことなく暗号化や復号が可能である．

TCG はプライバシーの保護を重要視しており，TCG の定めた仕様通りに実装するだけでは，公開鍵に対応する秘密鍵をどの TPM が保管しているか知ることができない．しかし，企業情報システムでは端末や所有者が特定できないと困る場合がある．

そこで企業の情報管理担当者が端末に ID を割り振り，TPM の公開鍵をその ID と関連づけて管理する TPM 公開鍵管理システム TKMS（TPM Key Management System）を開発した[4]．TKMS の概要は図 2 のようになっており，公開鍵管理サーバが公開鍵を管理する．TPM の秘密鍵で生成された署名や認証情報を検証する場合は公開鍵管理サーバに端末 ID と署名，および署名対象のダイジェストを送り，公開鍵管理サーバが検証を行う．

4. TKMS を利用した IB-ACS の設計

TKMS を利用して端末認証やインベントリの署名検証を行う IB-ACS の構成を図 3 に示す．本システムのアクセス制御の流れは以下の順に行う．なお，ACL とは Access Control List のことであり，アクセス制御の具体的なルールが記されたリストである．

証明書の発行

1. インベントリ収集部はインベントリを収集し，TPM の秘密鍵で署名する
2. 証明書管理部はインベントリと署名，端末 ID，端末認証

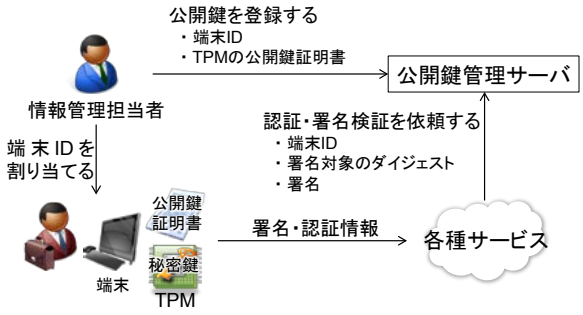

図 2　TPM 公開鍵管理システム TKMS の概要

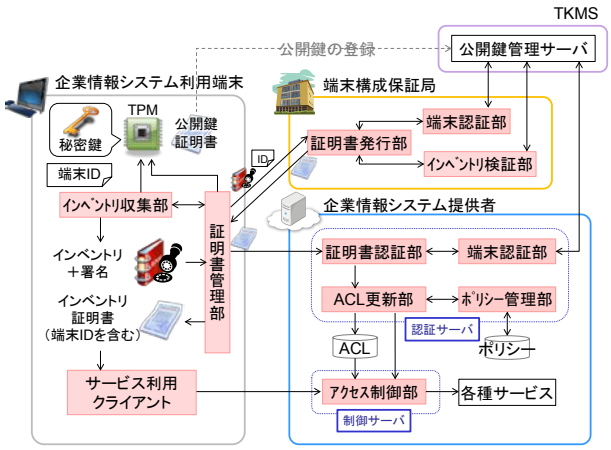

図 3　TKMS を利用したインベントリ証明書による
アクセス制御システム

に必要な情報を証明書発行部に送る

3. 証明書発行部は端末認証部，インベントリ検証部と連携して端末認証とインベントリの署名検証を行う．なお，署名検証自体は TKMS に依頼する
4. 証明書発行部はインベントリ証明書を発行する．証明書には端末 ID を含める

証明書の提出

1. 証明書管理部はインベントリ証明書と端末認証に必要な情報を証明書検証部に送る
2. 証明書認証部はインベントリ証明書が正当な端末構成保証局から発行されたものか検証する
3. 証明書認証部は端末 ID を証明書から取り出し，端末認証部と連携して端末認証する
4. 証明書認証部は ACL 更新部に更新を依頼する
5. ACL 更新部はポリシー管理部と連携して，端末のインベントリとポリシーからその端末のアクセス制御のルールを生成する
6. ACL 更新部は生成されたルールを ACL に反映させ，アクセス制御部に ACL 更新を通知する
7. アクセス制御部は ACL を再読み込みする

サービスの利用

1. サービス利用クライアントはアクセス制御部経由でサービスにアクセスを試みる
2. アクセス制御部は ACL にもとづいてアクセスを制御する．アクセスを許可するのであれば，各種サービスにデータを転送する

このようにインベントリに署名をして端末認証をすることでインベントリの改ざんやなりすましを防ぐことができる．また，インベントリ証明書に端末 ID を含めて，その端末 ID で端末認証することで，不正なインベントリ証明書の提出を防ぐことができる．

5. 実装と評価

TKMS を利用した IB-ACS を Java で実装した．使用したライブラリなどを表 1 に示す．実装により TPM で署名と認証を行って不正なインベントリやインベントリ証明書の提出を防ぐことができ，端末のインベントリによりアクセス制御を行えることを確認した．

設計した IB-ACS ではすべてのサービスへのリクエストはアクセス制御部を経由するので，アクセス制御部の処理に時間がかかるとサービス全体が遅延することになる．そこでクライアントからサービスにリクエストを送り，レスポンスが返送されるまでの応答時間を測定する実験を行った．

表 1　利用したソフトウェア，機器とバージョン

JDK/JRE	JDK 1.7.0[7]
Web コンテナ	GlassFish Server 3.1[8]
HTTP ライブラリ	HttpClient 4.1.2[9]
データベース	MySQL Ver. 14.14[10]
JDBC	MySQL Connector/J 5.1.28[11]
TPM	IFX 製 Ver. 1.2[12]
TCG Software Stack	jTSS 0.7[13]

アクセス制御部では ACL に基づいてアクセス制御するため，ACL のサイズが大きくなるとルールを検索するコストが増えて処理時間が増加すると思われる．実験では ACL に記載されたルール数を 10000 として，アクセス制御部の処理に要する時間をクライアントが Core i5 M560 @2.67GHz，4GB RAM の計算機，サーバが Core2 Duo P8700 @2.53GHz，3GB RAM の計算機で測定した．100 回の試行を行い平均を算出したところ 5600 マイクロ秒で処理が行えるという結果が得られた．

6. おわりに

本稿ではインベントリの証明書を発行し，証明書によるアクセス制御を行うシステム IB-ACS を TPM の公開鍵管理システム TKMS を利用して設計と実装を行った．IB-ACS では，インベントリに対して証明書が発行されることで最新のインベントリだけでなく，過去のインベントリを検証することができ，その端末がどのように運用・管理されてきたかという過程をもとにアクセス制御できる．端末で署名や端末認証を行う必要があり，私用の端末では秘密鍵が複製される恐れがあるが，TPM で鍵を管理することでこれを防ぐことができる．

実装によりアクセス制御の処理に要する時間を測定したところ，ルール数が 10000 の場合に 5600 マイクロ秒で行えることを確認した．これは十分実用可能な値であると考える．

IB-ACS を既存のユーザ認証などと組み合わせることによりきめ細かいアクセス制御が可能となる．

参考文献

[1] 稲月修：クラウドコンピューティングと企業情報システムの構造変革，知的資産創造，2011 年，1 月号，pp.44-55（2011）
[2] Wiggy, Z.: BYOC: Bring Your Own Computer , http://www.windowsitpro.com/content1/tabid/57/catpath/deployment/topic/byoc-bring-your-own-computer （参照 2011-10-07）
[3] 脇田知彦，福田洋治，白石善明，毛利公美，野口亮司：クラウド環境におけるインベントリ証明書を用いた端末制御，マルチメディア，分散，協調とモバイル（DICOMO2010）シンポジウム 論文集，pp.1448-1452（2010）
[4] 脇田知彦，毛利公美，白石善明，野口亮司：TPM を用いたインベントリ証明書による端末認証のための証明書発行・検証システムの設計と実装，情報学ワークショップ 2011（WiNF2011）論文集，pp.155-160（2011）
[5] Trusted Computing Group - Developers - Trusted Platform Module, http://www.trustedcomputinggroup.org/developers/trusted_platform_module （参照 2012-01-10）
[6] Trusted Computing Group - Home, http://www.trustedcomputinggroup.org/ （参照 2012-01-10）
[7] Java.com: あなたと Java，http://java.com/ （参照 2012-01-10）
[8] GlassFish - Open Source Application Server — Java.net, http://glassfish.java.net/ （参照 2012-01-10）
[9] Apache Software Foundation, Apache HttpComponents, http://hc.apache.org/ （参照 2012-01-10）
[10] MySQL :: The world's most popular open source database, http://www.mysql.com/ （参照 2012-01-10）
[11] MySQL :: MySQL Connectors, http://www.mysql.com/downloads/connector/（参照 2012-01-10）
[12] Infineon Technologies, http://www.infineon.com/（参照 2012-01-10）
[13] Trusted Computing for the Java(tm) Platform, http://trustedjava.sourceforge.net/ （参照 2012-01-10）