6Y-5

端末内のイベント発生時刻保証のための TPM とタイムスタンプサービスの連携

掛井将平† 脇田知彦‡ 毛利公美† 福田洋治†† 白石善明‡ 野口亮司‡‡
岐阜大学† 名古屋工業大学‡ 愛知教育大学†† (株)豊通シスコム‡‡

1. はじめに

日本版 SOX 法の導入以降，IT 統制の重要性が認識され，多くの企業では，その周知・徹底に向けた施策が進められている．これに伴い，サーバ群だけでなく，ユーザの端末で生じたイベントの信頼性も求められるようになってきた．例えば，端末内でのファイル操作（生成／変更／削除）やデータの送受信，アプリケーションの起動／停止／追加／削除等である．以降では，これらを端末内のイベントと称する．

データの信頼性を保証するための技術として，デジタル署名とタイムスタンプ[1] (RFC3161)がある．これらの技術により，データの作成主体と作成日時を特定することができる．しかしながら，一般にタイムスタンプは信頼できる外部サーバ TSA (Time Stamp Authority) によって発行されるものであり，その取得には毎回 TSA へのアクセスが必要になる．このことが理由で，オフラインのときに使用できないのはもちろんのこと，ユーザが指定する任意のイベントに対して，あるいは端末内のイベントが発生する度に TSA にタイムスタンプを要求するような使い方はできず，時刻保証が必要なごく限られたサービスに用途が限られているのが現状である．

このような TSA への負荷に起因するスケーラビリティの低下を改善し，オフライン状態の場合も含めて端末内の任意のイベントに対して時刻保証を実現するためには，TSA へのアクセスを最小限に留めた端末内での時刻保証の仕組みの実現が課題となる．

本稿では，その実現のために TPM とタイムスタンプサービスを連携した時刻保証方式を提案する．これにより，企業内の端末で作成されるデータの信頼性を保証することが可能になる．

2. 既存方式によるデータの時刻保証

2.1 デジタル署名

デジタル署名とはデータの正当性を確保するための技術である．電子的な記録であるデータは，簡単にその内容を複製・改ざん・偽造することができる．そこで，データとデジタル署名を関連付けることにより，以下の 2 つの性質を保証することができる．

(1) データの作成主体者証明
データの作成主体が第三者によってなりすまされていないことを証明できる．

(2) データの完全性証明
データが通信路などにおいて第三者によって改ざんされていないことを証明できる．

2.2 タイムスタンプ

タイムスタンプ[1]とは，時刻を保証するためのものである．この時刻とは，自己証明ではなく，信頼できる第三者機関(TTP)により証明されたものでないと証拠としての意味をなさない．タイムスタンプとデータを関連付けることにより，以下の 2 つの性質を保証することができる．

(1) データの存在証明
タイムスタンプを付けたデータがある時刻以前に存在していた，あるいは他のデータとの順序関係を証明することができる．

(2) データの完全性証明
2.1-(2)と同様．

2.3 デジタル署名とタイムスタンプの連携

デジタル署名がデータの作成者を証明し，タイムスタンプがデジタル署名の作成日時を証明することにより，連鎖的にデータの作成者と作成日時を証明することができる．[1]

2.4 既存方式で端末のイベント発生時刻保証を行う際の課題

既存方式は，外部サーバ TSA のみに依存した時刻保証を行っているため，以下の観点から頻繁に要求が発生する端末のイベント発生時刻保証の手段として現実的ではない．

(1) オフライン時の端末内のイベント時刻保証ができない．

(2) 組織内の多くの端末から端末内のイベントが発生する度に（あるいは任意のタイミングで）TSA にタイムスタンプを要求することになるため，TSA の負荷が大きくなる，すなわち，タイムスタンプサービスのスケーラビリティが低い．

3. 提案方式：TPM とタイムスタンプサービスを連携させたイベント発生時刻保証方式

3.1 TPM(Trusted Platform Module)

TPM[2]は TCG(Trusted Computing Group)が策定した仕様に基づき，PC 内部のマザーボード上に実装されているセキュリティチップである．TPM には TickCounter 機能と呼ばれる時間の「間隔」を保証する機能が備わっており，これを使うと PC の電源が投入されてからの“相対的な時間”を保証することができる．また，TPM には機器認証機能があり，AIK と呼ばれる鍵ペアを用いることによって端末を一意に特定できる．

3.2 提案方式のコンセプト

提案方式では，タイムスタンプサービスと TPM を連携させることにより，端末内で発生したイベントの時刻保証を行う．まず，ある時点で基準となる時刻（絶対時刻）を TSA により発行されたタイムスタンプで保証し，それ以降は TPM によって端末内で時間の間隔（相対時刻）を保証する．これにより，最初の絶対時刻取得以降は TPM のみで時刻保証を行うことができる．これは，2.4 で挙げたオフライン時の時刻保証および，TSA の負荷軽減と時刻保証対象のスケーラビリティ向上を両立できることを意味する．

3.3 提案方式のモデル

図 1 に提案方式のモデルを示す．このモデルは RFC3161 に準拠したタイムスタンプ・プロトコルに基づいている．

提案モデルでは，TickCounter 値を含んだ相対時刻付き署名(TickStampToken)(図 1 - ①)に対してタイムスタンプを要求し，TSA から受け取ったタイムスタンプトークン(TST)を絶対時刻付き署名（図 1 - ⑦）とする．相対時刻付き署名には，TPM が生成した相対的な時刻が記されており，絶対時刻付き署名には，相対時刻付き署名が作成された時点での，信頼できる時刻が記されている．この 2 つを関連付けることによって，イベント発生ごとに TSA に要求することなく端末内での時刻保証（図 1 - ⑧）が可能になる．

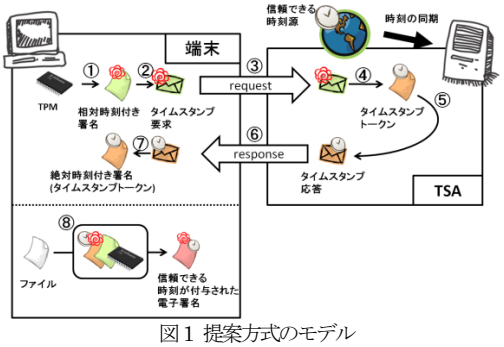

図 1 提案方式のモデル

3.4 システム構成と各処理の流れ

提案システムは，図 2 に示すような「時刻保証のための事前準備」，「端末内で発生したイベントの時刻保証」，「相対／絶対時刻付き署名の検証」の三つのフェーズから成る．各フェーズのシステム構成を図 3, 4 および 5 に示す．

Cooperation between TPM and Time Stamp Service For Guarantee of The Event Generation Time in Terminal

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

3.4.1. 時刻保証のための事前準備（図3）

[端末側]

1. 相対時刻付き署名作成部
   (1) TPMから現在のTickCounter値（TCV），Tickセッション ID（TSN），TickRate値(TRATE)，乱数（n）を取得する. 以降では，これらの情報をTickStampInfoと呼ぶ.
   (2) 1−(1)で取得した値から，SigningKey（K_{SIG}）により署名値（TickStamp）を作成する.
   $TickStamp = TPM\_TickStampBlob_{K_{SIG}}(TickStampInfo)$
   (3) TickStampとTickStampInfoを「相対時刻付き署名」としてCMS署名形式で保管する（図6）．これらのデータは検証の際に利用される.
2. 信頼時刻要求部
   (1) 相対時刻付き署名に対して，タイムスタンプ要求を作成し，送信する（図3−②）.
   (2) TSAからタイムスタンプ応答を受信する．（図3−⑥）.
3. 絶対時刻付き署名保管部
   (1) タイムスタンプ応答からタイムスタンプトークン（TST）を取り出す.
   (2) TSTの検証を行う.
   (3) 検証が正しく終了したら，TST を絶対時刻付き署名として保管する.

[TSA側]

1. 信頼時刻発行部
   (1) 端末からタイムスタンプ要求を受ける（図3−②）.
2. 要求検証部
   (1) タイムスタンプ応答に設定されているパラメータを検証する.
3. タイムスタンプトークン作成部
   (1) 検証が成功した場合のみ，TSTを作成する．（図3−⑤）
4. 信頼時刻発行部
   (1) TST をタイムスタンプ応答の中に含めて送信する．検証が失敗した場合は，TSTを含めずに送信する.（図3−⑥）

3.4.2. 端末内で発生したイベントの時刻保証（図4）

[端末側]

1. 初期設定部
   (1) 相対／絶対時刻付き署名を取得する.
   (2) TST 作成時の時刻，相対時刻付き署名作成時のTickCounter値，現在のTickCounter値を取得する.
2. 信頼時刻生成部
   (1) 1−(2)で取得した2つのTickCounter値の差を求める.
   (2) 2−(1)で計算した値が単位をミリ秒とした時刻保証を受けた時点からの経過時間[ms]（TickRate=1000 の場合）となるため，この経過時間をTSTが保持している時刻情報に足すことによって現在の時刻を計算する.

3.4.3. 相対／絶対時刻付き署名の検証（図5）

インシデント発生後に行われる検証は次のようになる.

[端末側]

1. 検証データ提示部
   (1) 検証に必要なデータ（相対時刻付き署名，絶対時刻付き署名）を検証者（第三者）に提示する.

[検証者]

1. 相対／絶対時刻付き署名比較部
   (1) 相対時刻付き署名のハッシュ値を計算する.
   (2) 1−(1)の結果と絶対時刻付き署名内のMessageImprintを比較し，相対／絶対時刻付き署名が紐付いていることを確認する.
   $Hash(TickStampToken) == MessageImprint$
2. 絶対時刻付き署名検証部
   (1) 絶対時刻付き署名の署名値を検証することにより，TSAと絶対時刻付き署名が紐付いていることを確認する.
   $Verify_{PK_{TSA}}(TSTInfo, Sig_{SK_{TSA}}(TSTInfo))$
3. 相対時刻付き署名検証部
   (1) 相対時刻付き署名の署名値を検証することにより，TPMと相対時刻付き署名が紐付いていることを確認する.
   $Verify_{K_{SIG_{pub}}}(TickStampInfo, Sig_{K_{SIG}}(TickStampInfo))$

3.5 開発環境

OSは端末側にWindows7，TSA側にCentOS5.4を用いた．言語はJavaを使用し，通信部部をTCPで実装した．タイムスタンプ関連の処理は，IAIK_TSP2.1[3]，IAIK_JCE4.0[3]を使用した．TPM関連の処理は，jTSS0.6[3]を使用した.

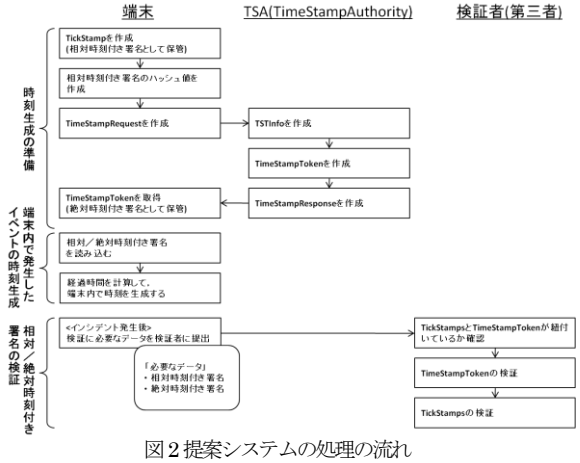

図2 提案システムの処理の流れ

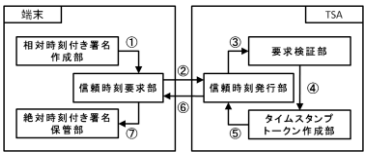

図3 信頼時刻保証のための事前準備

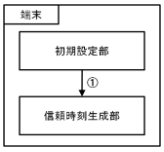

図4 端末内で発生したイベントの信頼時刻保証

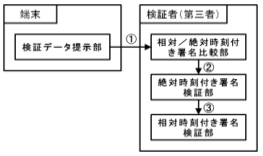

図5 相対／絶対時刻付き署名の検証

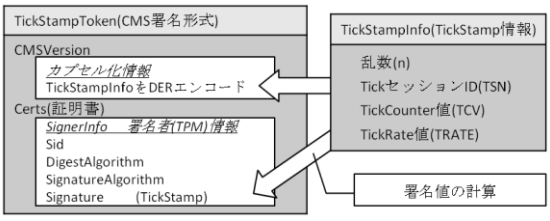

図6 相対時刻付き署名のフォーマット

4. むすび

本稿では，TPMとタイムスタンプサービスを連携させた，端末内のイベント発生時刻保証の方式を提案し，実装を行った．TSAへの集中アクセスに起因するタイムスタンプサービスのスケーラビリティの低下に対して，提案方式では，TPMとTSAを連携させることにより，TSAのみに依存することなく（サーバの負荷軽減），端末内で信頼性の高い時刻保証を実現できる．これにより，オフラインも含めて端末内で発生したあらゆるイベントに対する柔軟かつ高信頼な時刻保証が可能となった.

参考文献

[1] 情報処理推進機構, “タイムスタンプ・プロトコルに関する技術調査,” 2004年2月.
[2] 中村智久, 東川淳紀, “PC搭載セキュリティチップ(TPM)の概要と最新動向,” IPSJ Magazine Vol.47 No.5, 2006年5月.
[3] http://javadoc.iaik.tugraz.at/tsp/2.01/index.html (2011/1/8参照)