社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2005－DPS－122（11）
2005－CSEC－28（11）
2005／3／22

DVCS を拡張した複数方式タイムスタンプ検証サーバの開発

谷川嘉伸[1]　本多義則[1]　小黒博昭[2]　高村昌興[2]

[1]株式会社 日立製作所 システム開発研究所
[2]株式会社 NTT データ 技術開発本部

タイムスタンプ相互運用性やユーザ利便性を高める複数方式タイムスタンプ検証サーバを提案する。複数方式タイムスタンプ検証サーバは、任意の方式で作成されたタイムスタンプを統一的に検証する。我々は、実用性と拡張性を重視し、RFC 3029 を拡張したタイムスタンプ検証プロトコルを設計・開発した。また、検証対象とするタイムスタンプ方式として独自方式を用いた商用サービスのタイムスタンプを含めることを目標とし、ソースコードが非公開の検証モジュールを複数方式タイムスタンプ検証サーバにアドオンするために使用できる共通インタフェースを設計・開発した。実装した複数方式タイムスタンプサーバを用いて、RFC 3161 準拠のタイムスタンプと ISO/IEC 18014-2 のアーカイブ準拠方式のタイムスタンプの検証を行い、動作確認を行った。

DEVELOPMENT OF UNIVERSAL TIME-STAMP VALIDATION SERVER BASED ON DVCS

Yoshinobu Tanigawa[1]　Yoshinori Honda[1]　Hiroaki Oguro[2]　Masaoki Takamura[2]
[1]Systems Development Laboratory, Hitachi, Ltd.
[2]Research and Development Headquarters, NTT DATA Corporation

In this paper, we present the universal time-stamp validation server which improves the interoperability and usability of time-stamp validation. The server universally verifies the validity of any time-stamp token produced by any time-stamping scheme. We designed and developed the time-stamp validation protocol based on RFC 3029 for its practicality and extensibility. Moreover, we designed a common interface for verification modules of original non-standard time-stamping schemes. We constructed a prototype client-server system, and confirmed that the server successfully performed time-stamp validation of two types of time-stamps which are based on RFC 3161 and the ISO/IEC 18014-2 archiving scheme.

1. はじめに

電子データの完全性を保証する技術の一つとして、タイムスタンプ技術が注目されている。タイムスタンプ技術とは、電子データと時刻情報を暗号学的に結びつける技術であり、電子データの存在証明機能と電子データの完全性証明機能を有している[1]。この機能により、タイムスタンプ技術の利用者及び検証者は、タイムスタンプが付与された電子データがいつ存在したのか、また、それ以降に改ざんされていないのかどうかを確認することができる。

タイムスタンプ技術は、数多く研究開発されている。タイムスタンプ作成メカニズムの観点からシンプルプロトコル、リンキングプロトコル、分散プロトコルと呼ばれる方式が提案されている[1]。最近、シンプルプロトコルとリンキングプロトコルのアイデアが採用された RFC 3161[2]や ISO/IEC 18014[3][4][5]などの標準規格が策定された。ISO 規格は、独立トークン方式 3 種類とリンクトークン方式 2 種類のタイムスタンプ作成メカニズムを定義している。

このように、複数の方式のタイムスタンプ技術が、標準技術として定められているが、これは一方でタイムスタンプ技術にはユーザ利便性や相互運用性の問題が発生する可能性があるとも言える。一般的に、タイムスタンプ検証者は、想定されるタイムスタンプ方式に応じた検証ソフトウェアをそれぞれ作成あるいは導入していなければならない。もし、そうしなければ、タイムスタンプによっては検証できない可能性がある。また、タイムスタンプ方式毎に検証ソフトウェアを用意しなければならないというユーザ利便性を損ねる課題もある。今後、様々なコミュニティがオープンにつながれば、標準技術に準拠したタイムスタンプや独自方式のタイムスタンプなど様々なタイムスタンプが流通することが想定される。よって、検証者の利便性を高めると同時にタイムスタンプの相互運用性を確保することが重要である。

総務省は、e-JapanⅡ戦略の一環として、2003 年度から 3 ヶ年計画で、日本標準時を利用し有効かつセキュリティの高いタイムスタンプを高速に付与することができる「タイムスタンププラットフォーム技術」を確立するための研究開発を、産学官の連携のプロジェクトにより推進している[6]。研究開発のテーマの一つに、ユーザ利便性の向上とタイムスタンプの相互運用性の確保を目的としたタイムスタンプ検証技術の確立があり、ここでは、様々な方式によって作成されたタイムスタンプを統一的に検証するための複数方式タイムスタンプ検証サーバが開発されている[7]。

本論文は、上記のプロジェクトにおける 2003 年度に開発した複数方式タイムスタンプ検証サーバの設計と実装に関して記述したものである。複数方式タイムスタンプ検証サーバは、任意の方式で作成されたタイムスタンプを統一的に検証できるように設計されている。実用性と拡張性を重視し、RFC 3029[8]を拡張したタイムスタンプ検証プロトコルを設計・開発した。また、検証対象とするタイムスタンプ方式として商用サービスの独自方式を含めることを目標とし、ソースコードが非公開の検証モジュールをタイムスタンプ検証サーバにアドオンするために使用できる共通インタフェースを設計・開発した。

本論文の構成は以下の通りである。2 章では、複数方式タイムスタンプ検証サーバの目的と要件について述べる。3 章では、タイムスタンプ検証プロトコルの設計方針を述べ、4 章では、複数方式タイムスタンプ検証サーバの設計について記述する。5 章では、複数方式タイムスタンプ検証サーバの実装について述べる。6 章では、考察を行い、7 章では、結論と今後の課題についてまとめる。

2. 目的と要件

2.1 目的

複数方式タイムスタンプ検証サーバは、ユーザ利便性の向上とタイムスタンプ相互運用性を確保することを目的としている。様々な方式で作成されたタイムスタンプが流通した環境を想定し、タイムスタンプ検証者がタイムスタンプ方式毎に検証クライアントを用意・運用する手間を削減させること、また、様々な方式のタイムスタンプを統一的に検証することを目標としている。タイムスタンプ検証サーバの想定ユーザとしては、TTP としての検証局、統一的なポリシーの下で検証を行う企業内利用、が考えられる。

2.2 要件

前セクションで述べた目的を達成するために、タイムスタンプ検証サーバの要件を以下のように定めた。

(1)統一的なタイムスタンプ検証
独立トークン方式、リンクトークン方式、商用の独自方式など様々な方式で作成されたタイムスタンプの妥当性を統一的に検証できるようにする。また、検証結果に信頼性を与えるとともに、検証者にタイムスタンプ方式を意識させないようにする。

(2)拡張性
検証機能のエンハンスに耐え得る拡張性を持つようにする。検証機能のエンハンスは、二つの観点から考えられる。一つは、検証対象としてサポートするタイムスタンプ方式を増やすこと、もう一つは、検証機能の高度化である。検証機能の高度化として想定しているのは、タイムスタンプに含まれる時刻の信頼性検証（時刻トレーサビリティ検証）、検証記録の長期保証によるタイムスタンプの長期保証、である。

2.3 用語の定義

本論文で使用する用語を以下に定義する。

タイムスタンプトークン検証：タイムスタンプトークンの完全性（非改ざん性）を検証する。

タイムスタンプ検証：タイムスタンプトークン検証だけでなく、タイムスタンプトークンが付与された電子データの存在時刻と完全性（非改ざん性）を検証することも含む。

3. タイムスタンプ検証プロトコルの設計方針

2 章の要件を備えた複数方式タイムスタンプ検証サーバを設計するにあたり、タイムスタンプ検証やデータ検証に係る標準的な仕様及び公開された仕様を調査した。タイムスタンプ検証プロトコルに関しては、ISO/IEC 18014-1 と RFC 3029(Data Validation and Certification Server Protocols: DVCS プロトコル)を調査した。

3.1 ISO/IEC 18014-1 の概要

ISO/IEC 18014-1 で規定されたタイムスタンプ検証プロトコルは、タイムスタンプトークン検証のためのプロトコルである。主に、リンクトークン方式のタイムスタンプのように、タイムスタンプを発行したタイムスタンプ局へタイムスタンプの検証を委任する時や、第三者検証サービスを提供する検証局へタイムスタンプの検証を委任する時に使用される。

3.2 RFC 3029 の概要

RFC 3029（DVCS プロトコル）は、データ認証とデータ検証に関する 4 つのサービスを定義している。cpd サービスと ccpd サービスは、任意の電子データの所有を認証するサービスである。vsd サービスは、デジタル署名文書を検証するサービスであり、vpkc サービスは、公開鍵証明書の検証サービスである。

3.3 ISO/IEC 18014-1 と RFC 3029 の比較

タイムスタンプ検証サーバが満たすべき要件の観点から ISO/IEC 18014-1 の検証プロトコルを調査した。その結果、以下に示す問題点が判明した。

(1) タイムスタンプ方式が限定される
検証対象として指定するタイムスタンプトークンには、作成メカニズムを示すオブジェクト識別子が含まれる必要がある。そのため、オブジェクト識別子を持たないタイムスタンプトークンをサポートできない。商用のタイムスタンプサービスの中にはオブジェクト識別子を持たないものもあるため、様々なタイムスタンプ方式を扱うことには問題がある。

(2) 電子データの存在時刻と完全性の検証ができない
検証対象としてタイムスタンプ付与対象の電子データを含めることができない。そのため、タイムスタンプ検証者の一番の関心事であるタイムスタンプ付与対象の電子データの存在時刻と完全性を検証するサービスを提供できない。

(3) 独自の情報を扱うことができない
拡張データ領域が用意されていないため、独自の情報を扱うことができない。そのため、検証機能の拡張に耐え得るとは言い難い。

(4)検証結果の信頼性を示せない
検証結果の信頼性を示す情報を格納する領域が確保されていないため、検証クライアントに対して検証結果の信頼性を示すことができない。

一方、DVCS プロトコルでは、データ検証とデータ認証に係る 4 つのサービスを 1 種類の検証要求メッセージと DVC（Data Validation Certificates）と呼ばれるデジタル署名が付与された検証応答メッセージで表現している。つまり、サービスの種類毎に異なる検証・認証に係るデータ項目を統一的に扱う枠組みを持っている。また、検証サービスの追加を許す構造を持っているとともに検証要求メッセージと検証応答メッセージには、ユーザ定義の拡張情報を格納する領域が確保されている。

このように、タイムスタンプ検証サーバが満たすべき要件の観点から RFC 3029 を見た場合、RFC 3029 が持つ拡張性を利用すれば、これらの要件を満たせる見通しが得られた。そこで、我々は、実用性と拡張性を備える RFC 3029 をベースにタイムスタンプ検証プロトコルを設計することにした。

4. 複数方式タイムスタンプ検証サーバの設計

4.1 タイムスタンプ検証の抽象化

複数方式タイムスタンプ検証サーバ上で、統一的なタイムスタンプ検証を実現するために、タイムスタンプ検証内容を抽象化した。RFC 3161 や ISO/IEC 18014 などの標準化技術仕様に記載されたタイムスタンプ検証内容を分析し、タイムスタンプ方式に依存しない共通的な検証項目として、次の 3 つの検証内容に整理した。我々は、これらの検証を総称してタイムスタンプの正当性検証と呼ぶ。

(1) タイムスタンプトークンの形式検証
タイムスタンプ方式が規定するフォーマットに合致しているのかどうかを検証する。また、複数方式タイムスタンプ検証サーバが扱うことのできるタイムスタンプトークンなのかどうかを検証する。RFC 3161 準拠のタイムスタンプの場合、ASN.1 バイナリ符号化の妥当性、タイムスタンプトークンに含まれるデータ間の整合性、暗号アルゴリズムのサポート有無などを確認する。

(2) タイムスタンプトークンの正当性検証
タイムスタンプトークン検証を表す。RFC 3161 準拠のタイムスタンプの場合、タイムスタンプに付与された TSA の署名や TSA の公開鍵証明書の検証を行う。また、ISO/IEC 18014-2 のアーカイブ方式のタイムスタンプの場合、そのタイムスタンプを発行したタイムスタンプ局に問い合わせることによりタイムスタンプの改ざん有無を確認する。

(3) タイムスタンプトークンと電子データの対応検証
タイムスタンプが使用するハッシュ関数を用いて電子データのハッシュ値を求め、その値が、タイムスタンプトークンに含まれる該当ハッシュ値と同一なのかどうかを確認する。

これら(1)(2)(3)の一連の検証により、タイムスタンプ付与対象の電子データが、ある時点に存在し、それ以降改ざんされていないことを確認することができる。

4.2 タイムスタンプ検証プロトコル

タイムスタンプ検証プロトコルは、タイムスタンプ検証を要求するクライアントとタイムスタンプ検証サーバ間で規定されるアプリケーションレベルの通信プロトコルである。以下に RFC 3029 を拡張したタイムスタンプ検証要求メッセージとタイムスタンプ検証応答メッセージを示す。

4.2.1 タイムスタンプ検証要求メッセージ

タイムスタンプ検証プロトコルにおけるタイムスタンプ検証要求メッセージは、RFC 2630 で定義された CMS ContentInfo である。ContentInfo 内の content は、RFC 3029 で定義された DVCSRequest を表す。

我々は、RFC 3029 の DVCSRequest で定義される version、service、data の 3 つのフィールドを拡張した。バージョンを示す version は、RFC 3029 で指定された 1 ではなく、2 とした。サービスの種類を示す service には、タイムスタンプ検証サービスを示す vtst(5)を追加した。data は、検証対象のデータを格納するフィールドである。ここでは、RFC 3029 で規定されている任意のバイナリデータを示す OCTET STRING データ型を持つ message フィールドを利用する。ただし、タイムスタンプ検証サーバは、検証要求メッセージに含まれる message フィールドのバイナリデータは、新規に設計した TimeStampInfo データ型を持つタイムスタンプ情報であると解釈する。

```
TimeStampInfo ::= SEQUENCE {
    tstType  ContentType  OPTIONAL,
    tst      OCTET STRING ,
    content  OCTET STRING OPTIONAL
}
```

TimeStampInfo は、タイムスタンプ方式を示す tstType フィールド、タイムスタンプトークンのバイナリデータ列を示す tst フィールド、タイムスタンプ対象電子データのデータ列を表す content から構成される。タイムスタンプ情報の必須フィールドは、tst フィールドだけである。そのため、検証クライアントは、検証対象データとして、タイムスタンプトークンだけを指定するだけでもよい。また、tst フィールドは、任意のバイナリデータ列を格納することができるので、任意の方式のタイムスタンプトークンを検証対象にすることができる。つまり、本検証プロトコルは、サポートするタイムスタンプ方式を容易に増やせるという拡張性を持つ。

4.2.2 タイムスタンプ検証応答メッセージ

タイムスタンプ検証プロトコルにおけるタイムスタンプ検証応答メッセージは、RFC 2630 で定義された CMS SignedData である。SignedData 内の encapContentInfo は、RFC 3029 で定義された DVCSResponse を表す。DVCSResponse は、dvCertInfo、あるいは、dvErrorNote から選択する。前者は、検証サービスの実行が成功した時の応答であり、後者は、検証サービスの実行に失敗した場合に発行される。なお、dvCertInfo が選択された SignedData は、DVC（Data Validation Certificate）と呼ばれる。

以下、dvCertInfo、及び dvErrorNote の詳細を説明する。

(1) dvCertInfo
タイムスタンプ検証サービス向けに拡張したフィールドは、version、dvStatus、certs、extensions である。version は、RFC 3029 で指定された 1 ではなく、2 とする。dvStatus と certs は、RFC 3029 の考え方と同様、検証結果を格納するフィールドとして使用する。従って、dvStatus は、グローバルな検証結果を示し、certs は、詳細な検証結果を表すのに使用する。extensions は、タイムスタンプ方式に依存せずに共通的に存在する時刻情報やドキュメントハッシュ値などのタイムスタンプトークンの内容を示す情報を格納する。

以下、拡張したフィールドを説明する。

(a) dvStatus

dvStatus は、グローバルな検証結果を示す。RFC 2510 で定義される PKIStatusInfo データ型である。グローバルな検証結果とは、タイムスタンプ検証サーバが実行する複数の検証処理結果から判断される総合的な検証結果である。

(b) certs

certs は、詳細な検証結果を格納し、複数の TargetEtcChain から構成される。一番目の TargetEtcChain は、正当性検証の個々のサブ検証結果を格納する。二番目以降は、時刻信頼性（時刻トレーサビリティ）の詳細な検証結果などを格納する予定である。

実装した正当性検証結果の詳細情報を格納する TargetEtcChain の仕様は、**表** 1の通りである。

表 1：正当性検証における TargetEtcChain の仕様

フィールド名	説明
target	総合的な正当性検証結果を示す。PKIStatusInfo で表現する。
chain	正当性検証結果の詳細。以下の順番で並んでいる (1) タイムスタンプトークン形式検証 (2) タイムスタンプトークン正当性検証 (3) タイムスタンプトークンと電子データの対応検証
pathProcInput	―

正当性検証の総合的な評価である target は、クライアントから送信される検証対象データの種類に応じて解釈される。検証対象として、タイムスタンプトークンだけが指定された場合は、タイムスタンプトークン自体の正当性検証結果を表す。一方、タイムスタンプトークンとタイムスタンプ付与対象となる電子データが指定された場合は、タイムスタンプトークン自体の正当性だけでなく、電子データの存在時刻と完全性の検証結果として解釈できる。

chain には、正当性検証の 3 つの検証内容を格納する。それぞれ、タイムスタンプトークン形式検証、タイムスタンプトークン正当性検証、タイムスタンプトークンと電子データの対応検証である。

(c) extensions

extensions は、RFC 2459 で定義された Extension のシークエンスである。複数方式タイムスタンプ検証サーバは、一つの Extension を使用する。このデータ構造へタイムスタンプ方式に依存しないタイムスタンプトークンの内容を示す情報を格納する。このタイムスタンプトークン情報として、RFC 3161 や ISO/IEC 18014 で規定される TSTInfo というデータを採用する。これにより、タイムスタンプフォーマットを解析できない検証クライアントに対して、タイムスタンプトークンが主張する時刻情報、時刻精度情報、ハッシュ情報（ハッシュ関数の識別子とハッシュ値）を提示することが可能になる。検証クライアントは、これらの情報に含まれるハッシュ情報を用いることで、タイムスタンプトークンと電子データの対応検証をローカルに実行することができる。その結果、検証クライアントは、電子データの存在時刻と完全性を知ることができる。

(2) dvErrorNote

検証サーバは、検証サービス実行に失敗した場合は、dvCertInfo ではなく、dvErrorNote を作成する。dvErrorNote は、DVCSErrorNotice というデータ構造を持つ。エラー内容は、transactionStatus に格納される。

4.3 共通インタフェース

商用のタイムスタンプサービスの中には、TSA 事業者が提供する検証ソフトウェアを使用してタイムスタンプの検証を行うものがある。TSA 事業者は、自身が提供するタイムスタンプサービスの普及促進のため、その検証サービスを、複数方式タイムスタンプ検証サーバが提供するサービスの一つに追加して欲しいという要望があると考えられる。一方、事業者はその検証ソフトウェアのソースコードおよび詳細仕様を企業秘密として検証サーバ管理者に公開したくないという要望もあると考えられる。このような商業的な事情を考慮し、図 1 のように、事業者が提供する検証ソフトウェアと複数方式タイムスタンプ検証サーバを接続するための共通インタフェースを設計した。

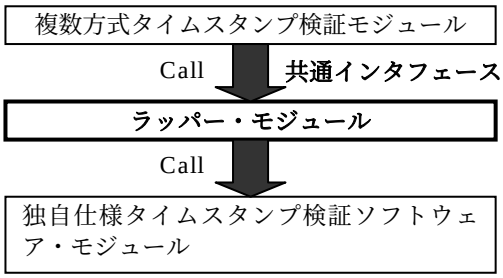

図 1：複数方式タイムスタンプ検証サーバのソフトウェアスタック

これにより、事業者は共通インタフェースに準拠したラッパー・モジュールを構築しさえすればよく、事業者の検証ソフトウェアを隠蔽することができる。その結果、事業者は、企業秘密情報を検証サーバ管理者へ公開することなく、検証サーバがサポートするタイムスタンプに自身のタイムスタンプを追加することが可能になる。共通インタフェースに準拠したラッパー・モジュールの実装例を表 2 に示す。

表 2：共通インタフェースに準拠したラッパー・モジュールの実装例

(1) タイムスタンプトークンの形式検証	isSStoken(struct SSParam param, struct Result *result, int token_length, unsigned char *token);
(2) タイムスタンプトークンの正当性検証	SSVerify(struct SSParam param, struct Result *result, int token_length, unsigned char *token);
(3) タイムスタンプトークンと電子データの対応検証	int SSVerifyWD(struct SSParam param, struct Result *result, int token_length, unsigned char *token, int doc_length, unsigned char *doc);

5. 実装

複数方式タイムスタンプ検証サーバを Sun Blade™ 150、CPU:550MHz×1、メモリ:640MB、OS：Solaris™ 8 上で C 言語を用いて実装した[1]。また、nShield™を用いて私有鍵などの暗号情報を保護すると共にデジタル署名作成などの暗号処理を実装した[2]。

タイムスタンプ検証サーバは、RFC 3161 準拠のタイムスタンプ検証ソフトウェアをベースに開発した。TSA 公開鍵証明書の検証に関しては、トラストアンカとなる CA のルート証明書を予め検証サーバに導入するモデルとした。また、証明書検証結果として、GPKI の政府認証基盤相互運用性仕様書で規定されている証明書検証結果コードを使用した[9]。共通インタフェースの動作確認のため、RFC 3161 準拠以外のタイムスタンプ検証にこの共通インタフェースを利用した。

実装したタイムスタンプ検証サーバがサポートするタイムスタンプ形式は 2 つである。一つは、デジタル署名技術を使用した RFC 3161 準拠方式のタイムスタンプである。もう一つは、アーカイブ技術を使用した ISO/IEC 18014-2 アーカイブ準拠方式のタイムスタンプである。ISO/IEC 18014-2 アーカイブ準拠方式のタイムスタンプを検証するモジュールは、共通インタフェースの規約に基づいて構築されている。

図 2のシステム構成で、タイムスタンプ検証サーバの動作確認を行った。タイムスタンプ検証サーバは、同一 LAN 内で Microsoft® Windows® XP ベースの検証クライアントとディレクトリサーバと接続している[3]。タイムスタンプ検証クライアントとタイムスタンプ検証サーバ間は、HTTP を用いた。ディレクトリサーバは、RFC 3161 準拠のタイムスタンプに関わる公開鍵証明書の失効リストを公開している。タイムスタンプ検証サーバは、このディレクトリサーバへアクセスし、公開鍵証明書の失効リストを取得する。また、タイムスタンプ検証サーバは、インターネットを介して ISO/IEC 18014-2 アーカイブ準拠方式の TSA と通信を行い、ISO/IEC 18014-2 アーカイブ準拠方式のタイムスタンプ検証の一部を委任する。

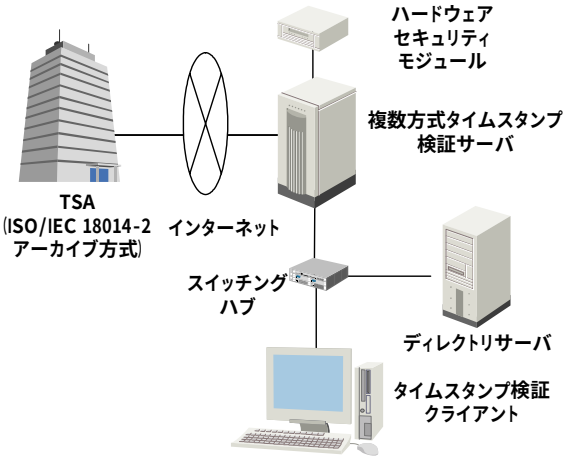

図 2：複数方式タイムスタンプ検証システム構成

タイムスタンプ検証サーバの動作確認を行ったところ、仕様通り、RFC 3161 準拠方式と ISO/IEC 18014-2 アーカイブ準拠方式の 2 方式のタイムスタンプ検証が可能であることが確認された。図 3は、検証クライアントのディスプレイ上に表示される検証結果画面の例である。

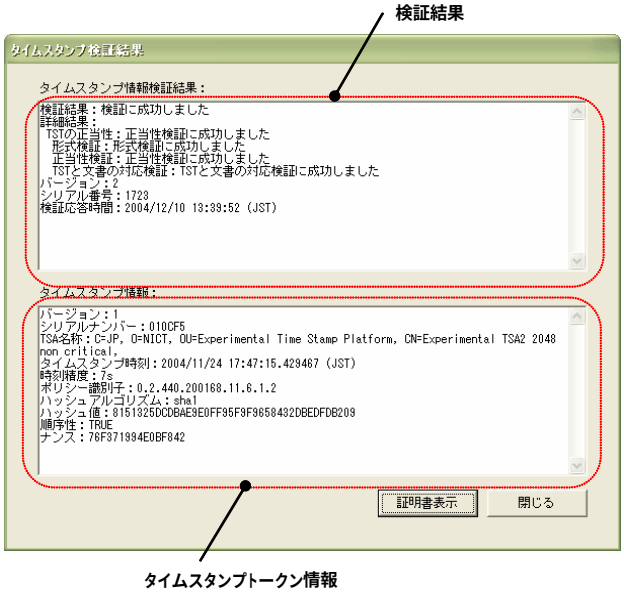

図 3：検証クライアント画面例

検証結果画面の上側のエリアは、タイムスタンプの検証結果を示す。上から順に、総合的な検証結果、形式検証、正当性検証、タイムスタンプトークンと電子データの対応検証の結果が表示される。一方、検証結果画面の下側のエ

[1] Sun、Blade、および Solaris は、米国および他の各国における Sun Microsystems, Inc. の商標または登録商標です。

[2] nShield は，英国 nCipher Corporation Ltd. の商標又は登録商標です。

[3] Microsoft 及び Windows は、米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

リアは、タイムスタンプトークンの内容を示す TSTInfo の情報である。このように、複数方式タイムスタンプ検証サーバは、複数の方式のタイムスタンプを統一的に検証した結果を発行していることが示される。

6. 考察

開発したタイムスタンプ検証サーバにおいて、実用性の観点から、任意のタイムスタンプ方式の検証可能性と性能を考察する。

6.1 任意のタイムスタンプ方式の検証可能性

開発したタイムスタンプ検証サーバを用いて、RFC 3161 準拠のタイムスタンプと ISO/IEC 18014-2 アーカイブ準拠方式のタイムスタンプを統一的に検証することができた。

タイムスタンプ検証プロトコルの検証要求では、タイムスタンプ方式を特定する情報を必須としておらず、任意の方式のタイムスタンプを検証要求として含めることができる。そのため、タイムスタンプ検証プロトコルは、今回サポートしたタイムスタンプ方式だけでなく、商用の独自方式のタイムスタンプを含めた様々な方式のタイムスタンプも対応することができると言える。

商用の独自方式のタイムスタンプ検証をサポートするために、共通インタフェースを開発した。今回は、共通インタフェースの実用性を確認するために、ISO/IEC 18014-2 アーカイブ準拠方式のタイムスタンプを検証対象とした検証モジュールを構築した。任意の方式のタイムスタンプをサポートできるのかどうかについては、タイムスタンプトークン情報のマッピングが可能かどうかに依存する。共通インタフェースは、タイムスタンプ検証モジュールに対して、TSTInfo 構造のデータ作成を求めている。TSTInfo には、バージョン、シリアル番号、ハッシュ値、時刻情報などが含まれるが、商用の独自方式のタイムスタンプの中には、これらの情報の一部を含まない、あるいは、表現が異なっている可能性がある。これらのマッピングに関しては十分に検討する必要がある。

6.2 性能

開発したタイムスタンプ検証サーバは、検証クライアントとの間の通信時間を除き、1 検証要求あたり、1〜4 秒で検証を実行した。ユーザが GUI などを介して対話的に検証する方法であれば、実用性があると思われる。なお、アプリケーションの要件によっては、より性能を求められる可能性がある。例えば、2005 年 4 月から施行される e-文書法においては、国税関係書類にタイムスタンプを付与することが求められる。さらに、後日、複数のタイムスタンプを一括して検証することが要求されるため、検証速度が重要になる。検証速度向上に関しては、今後の課題である。

7. まとめ

複数方式タイムスタンプ検証サーバの設計と実装を行った。実用性と拡張性を重視し、RFC 3029 を拡張したタイムスタンプ検証プロトコルを設計・開発し、任意の方式で作成されたタイムスタンプを統一的に検証できるようにした。また、検証対象とするタイムスタンプ方式において商用サービスの独自方式を含めることを目標とし、非公開の検証モジュールをタイムスタンプ検証サーバにアドオンするために使用できる共通インタフェースを設計・開発した。実装したタイムスタンプ検証サーバを用いて RFC 3161 準拠方式と ISO/IEC 18014-2 アーカイブ準拠方式の 2 方式のタイムスタンプの検証動作を確認した。

今後の課題は、タイムスタンプに含まれる時刻の信頼性を検証する機能（時刻トレーサビリティ検証機能）を追加することである。また、e-文書法などでは、長期にわたってタイムスタンプの有効性を保証する必要性がある。これを踏まえ、検証記録の長期保証によるタイムスタンプの長期保証機能を実現することも課題である。

謝辞

本研究は、総務省が推進しているタイムスタンププラットフォーム技術の研究開発を行っている NICT が進める実証実験プロジェクトの中で行われたものである。

参考文献

[1] Bruce Schneier : Applied Cryptography: Protocols, Algorithms, and Source Code in C, Second Edition, Wiley; 2 edition (1995)

[2] C. Adams, P. Cain, D. Pinkas, and R. Zuccherato.: Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP) - RFC 3161. IETF (2001)

[3] ISO/IEC: ISO/IEC 18014-1:2002 Information technology -- Security techniques -- Time-stamping services -- Part 1: Framework (2002)

[4] ISO/IEC: ISO/IEC 18014-2:2002 Information technology -- Security techniques -- Time-stamping services -- Part 2: Mechanisms producing independent tokens (2002)

[5] ISO/IEC: ISO/IEC 18014-3:2004 Information technology -- Security techniques -- Time-stamping services -- Part 3:Mechanisms producing linked tokens (2004)

[6] 総務省：平成 15 年度版 情報通信白書、http://www.johotsusintokei.soumu.go.jp/whitepaper/ja/h15/index.html (2003)

[7] NICT：時刻認証基盤技術実験装置（Experimental System of Time Stamping Based on the Japan Standard Time）仕様書 (2003)

[8] C. Adams, P. Sylvester, M. Zolotarev, and R. Zuccherato. : Data Validation and Certification Server Protocols – RFC 3029. IETF (2001)

[9] 総務省：政府認証基盤（GPKI）政府認証基盤相互運用性仕様書 (2003)