



INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN

2003 情経第 0351 号

情報セキュリティ対策研究開発評価等事業
PKI 相互運用テストスイートへの機能追加開発および関連調査

テストスイートのタイムスタンプ・プロトコル検証機能概要

2004 年 4 月

独立行政法人 情報処理推進機構

1 はじめに

本プロジェクトの開発作業部分は、平成 15 年度において「タイムスタンプ・プロトコルに関する技術調査」の成果であるタイムスタンプのテストケースを自動検証できるようにするため、平成 14 年度成の成果である「PKI 相互運用テストスイート」に各種機能を追加実装するものである。これによって完成したテストスイートを「TSP 相互運用テストスイート」と呼ぶ。

本書では、この PKI 相互運用テストスイート (TSP Interoperability Test Suite) の設計概要を説明する。

2 背景

タイムスタンプ・プロトコルに関する技術調査より、現在ほとんどのテストサイトで運用されている TSA やクライアントは、RFC 3161 及び関連する規定に十分に準拠しているとはいえないことがわかった。こうした仕様への準拠性の欠如は、アプリケーション間の相互運用性を阻害する要因となる。

タイムスタンプ・プロトコルを利用した様々なアプリケーション開発が活性化し、タイムスタンプが広く利用されるようになるためには、そのアプリケーションが RFC 3161 を始めとする規定に準拠し、相互運用性が十分に確保されていることを確認できるツールが必要であると考えられる。

そこで本プロジェクトでは、タイムスタンプ・プロトコルの相互運用性を確保するために、TSP テストスイートを設計した。一般に、TSP のテストは以下の立場の人々によって利用されると考えられる。

1. タイムスタンプサーバー (TA や TSA など) の実装者
2. TSP を利用したクライアントアプリケーションの実装者

本テストスイートは、主に 2 に該当する人々を主な対象としてつくられている。テストスイートを広く公開することで、RFC 3161 に準拠し、相互運用性の確保された TSP アプリケーション開発を促進することを目的とする。

3 設計方針

テストスイートへの要求を、以下にまとめる。

- TSA の動作をシミュレートでき、TSP over HTTP をサポートすること
- テストケースに基づいて、以下のタイムスタンプ・レスポンス及びトークンを自由に生成できること
 - RFC 3161 をはじめとする仕様に準拠するもの
 - RFC 3161 をはじめとする仕様に準拠しないもの
- テストケースを自由に閲覧・編集できる機能を有すること
- 将来のタイムスタンプ・プロトコルの拡張・仕様変更に極力対応できること

テストスイートは、RFC 3161 をはじめとする仕様に準拠したレスポンスやトークンを生成することで、テスト対象とするクライアントが正常に通信・検証などの処理を行えるかどうかを評価することができる。

さらに、これらの仕様に準拠しないものをあえて生成することで、不適切なデータに対して、適切なエラーメッセージを示すことができるかどうかを評価することもできる。一般に、TSA は適切なデータを生成することを目的として設計されており、任意に不適切なデータを生成することは困難である。本プロジェクトでは、不適切なデータに対する処理や例外的な処理も、相互運用性を確保する上で重要と考える。

テストスイートは、PKIX TSP Interoperability Testing と同様に、RFC 3161 の中で MAY、SHALL、SHOULD、MUST もしくは REQUIRED として明示された全ての項目を表現できる仕組みとなっている。さらに、将来 RFC 3161 が拡張されたり、TSP に関する新たな仕様が発行されても、極力それらの仕様を吸収できるように汎用的な設計とした。

4 構成要素と動作概要

本テストスイートのタイムスタンプ・プロトコル検証機能は、以下の要素から構成される。

- テスト DB
- テスト管理 CGI
- テストドライバ
 - TSA レスポンダシミュレータ CGI
 - TSR/TST 生成プログラム
 - 証明書生成プログラム
 - TSQ 設定ファイル生成プログラム
- テスト対象クライアント

テスト DB とは、テストケース及びテスト結果を格納するための DB である。テスト管理 cgi とは、Web ブラウザを用いて、テスト DB に納められているテストケースの閲覧・編集及び、テスト結果の閲覧・分析を行う機能を有する cgi プログラムである。

テストドライバとは、テストを実行するために必要な各種プログラム群の総称をいう。また、テスト対象クライアントとは、テストスイートを用いて相互運用性評価を行う対象となる TSP クライアントを指す。

テストスイートのユーザ（テスト実施者）が操作するのは、テスト対象クライアントと、テスト管理 CGI のみである。以下に、テストスイートの全体構成を示す。

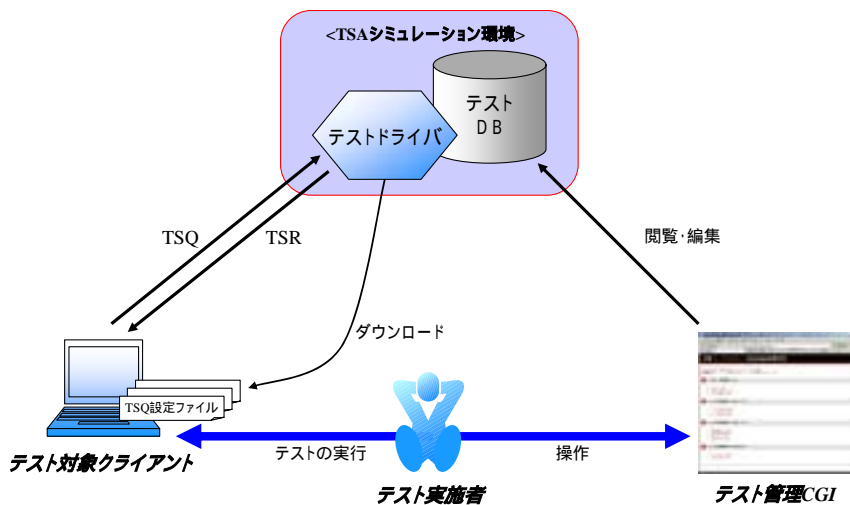


図1 テストスイートの動作概要 (TSR テスト)

1つのテストは、以下の順序で実行される。

まず、テスト対象クライアントは、TSQ 設定ファイル生成プログラムによって生成された TSQ 設定ファイルに基づいて、TSA レスポンダシミュレータに対して TSQ を送付する。従って、テスト対象クライアントは本テストスイートで定める TSQ 設定ファイルの書式を理解できなくてはならない。このため、任意の TSP クライアント（例えば OpenTSA の ts コマンドなど）をテスト対象クライアントとして評価する場合には、その TSP クライアント専用に、何らかのラップ用プログラムをつくる必要がある。

テストドライバは、テスト対象クライアントから送られてきたこの TSQ に対し、テストケースで規定された通りの TSR を作成し、これを返送する。最後にテスト対象クライアントがこの TSR を受信し、検証を行う。

テスト対象クライアントが示した検証結果が、テストケースの期待した通りのものであったなら、そのテスト結果は「成功(OK)」となり、そうでない場合は「失敗(NG)」となる。テスト結果や、テスト対象クライアントの標準出力・標準エラー出力などは、テスト DB に格納することもできる。

以上が、最も一般的なテストの流れである。このように、テスト対象クライアントが TSR を検証するテストを、TSR テストと呼ぶ。テストケースによっては、TSR を用いずに、テスト対象クライアントが直接 TST ファイルを検証するものもある。これを、TST テストと呼ぶ。TST テストの構成を、以下に示す。

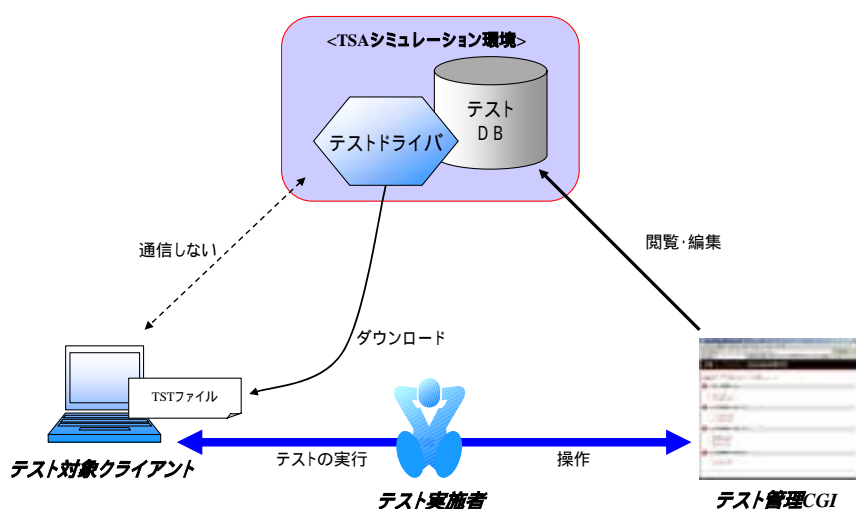


図2 テストスイートの動作概要 (TST テスト)

さらに、テストケースによっては、TST に含まれる ordering フラグや accuracy の値によって、2 つの TST 間の前後関係を比較する必要があるものも考えられる。これを、Accuracy&Ordering テストと呼ぶ。
Accuracy&Ordering テストの構成を、以下に示す。

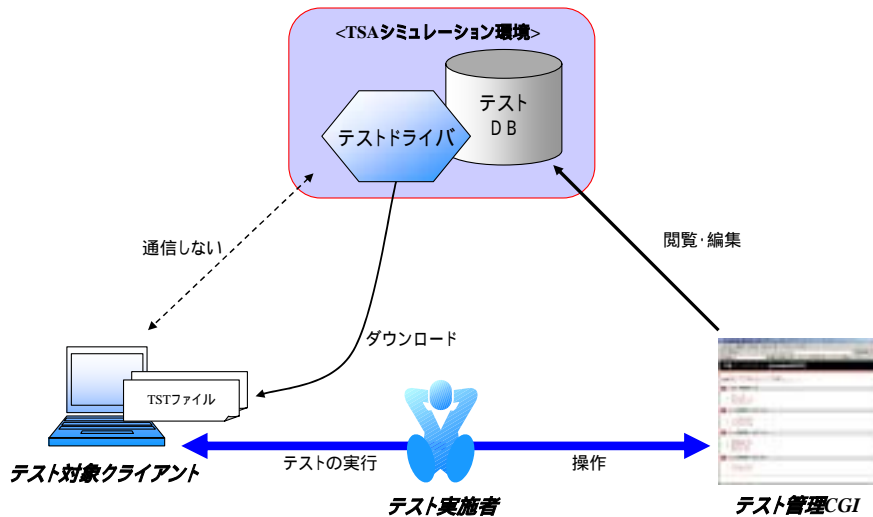


図3 テストスイートの動作概要 (Accuracy&Ordering テスト)

TSR、TST 及び Accuracy&Ordering テストの特徴を以下にまとめる。

表1 TSR テスト・TST テスト・Accuracy&Ordering テストの比較

テスト種別	TSP トランザクション	テスト対象クライアントに与えるもの	テスト対象クライアントの出力
TSR	有り	TSQ 設定ファイル	TSR の検証結果
TST	無し	TST ファイル (1 つ)	TST の検証結果
accuracy	無し	TST ファイル (2 つ)	TST の前後関係

5 テストドライバ

テストドライバは、以下のプログラムから構成される。

- TSA レスポンダシミュレータ CGI
- TSR/TST 生成プログラム
- 証明書生成プログラム
- TSQ 設定ファイル生成プログラム

TSQ 設定ファイル生成プログラムは、データベースに納められているテストケースに応じ、以下の形式の TSQ 設定ファイルを出力する。

```
#TimeStampRequest
TimeStampRequest.version={INTEGER}
TimeStampRequest.hashAlgorithm={AlgorithmIdentifier}
TimeStampRequest.hashedException={Hash}
[TimeStampRequest.reqPolicy={OID}]
[TimeStampRequest.nonce=hex:{INTEGER}]
[TimeStampRequest.certReq={TRUE|FALSE}]
[TimeStampRequest.extensions={extension}]
<EOF>
```

なお、[]はオプション項目、{a|b}は選択項目とする。また、シャープ記号で始まる行はコメント行である。

以上の形式の設定ファイルを用いることにより、[TSP]に準拠した TSQ だけではなく、準拠しない TSQ も任意に生成することができる。例えば、`TimeStampRequest.version` を [TSP] で定義された 1 (v1) ではない値に設定することも可能である。

続いて、テストドライバを構成する以下の 4 つのプログラムについて説明する。

- TSA レスポンダシミュレータ CGI
- TSR/TST 生成プログラム
- 証明書生成プログラム

TSA レスポンダシミュレータ CGI 以降のプログラムが協調して動作することで、TSA サーバーを模倣し、TSR テストを実行することができる。また、TST 生成プログラム以降のプログラムが動作することで、TST テスト及び

Accuracy&Ordering テストを実行するために必要な TST ファイルを生成することができる。

以下に、各プログラムと、タイムスタンプ・プロトコルにおけるレイヤとの対応関係を示す。

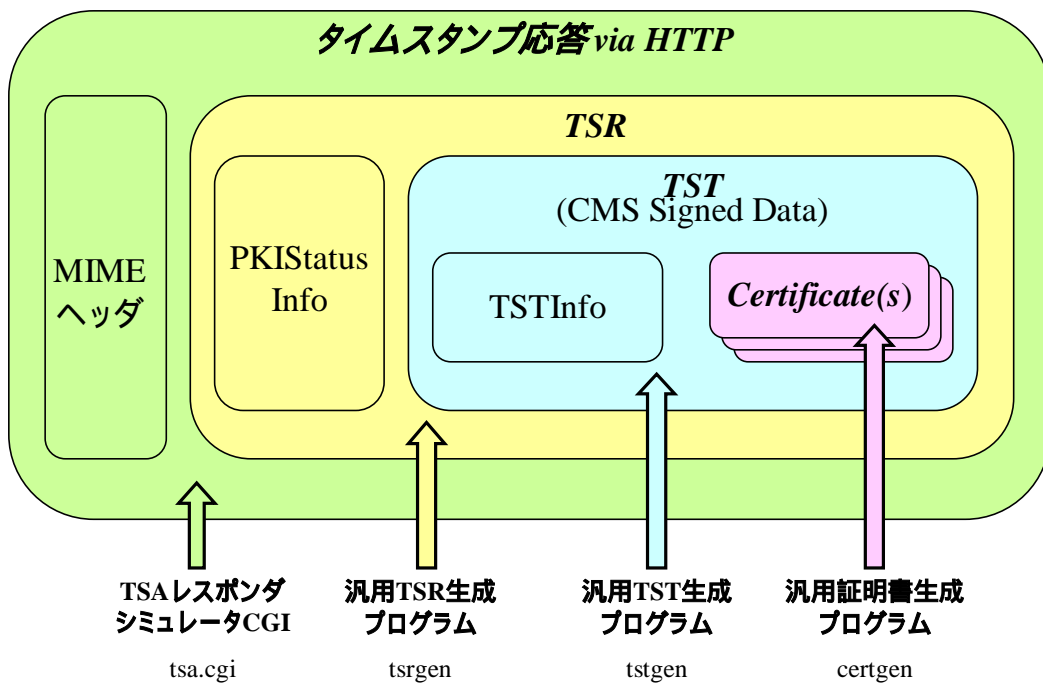


図4 プロトコル・レイヤとテストドライバとの関係

テストドライバが TSR もしくは TST を生成する際には、まずデータベースから以下の形式の設定ファイルを生成し、これを元に TSR/TST ファイルを作成する。

```
#TimeStampResp
TimeStampResp.status={PKIStatus}
#PKIStatusInfo
[PKIStatusInfo.statusString={PKIFreeText}]
[PKIStatusInfo.failInfo={PKIFailureInfo}]
#TSTInfo
[
TSTInfo.version=dec:{INTEGER}
TSTInfo.policy={OID}
```

```

[TSTInfo.messageImprint.AUTO=TRUE]
TSTInfo.messageImprint.hashAlgorithm={AlgorithmIdentifier}
}
TSTInfo.messageImprint.hashedException={Hash}
TSTInfo.serialNumber={dec|hex}:{INTEGER}
TSTInfo.genTime={GeneralizedTime}
[TSTInfo.accuracy={Accuracy}]
[TSTInfo.ordering=TRUE]
[TSTInfo.nonce.AUTO=TRUE]
[TSTInfo.nonce=hex:{INTEGER}]
[TSTInfo.tsa={directoryName}]
*[TSTInfo.extensions={extension}]
:
]
#CMS SignedData(rfc2634)
[ContentInfo.contentType={OID}]
[EncapsulatedContentInfo.eContentType={OID}]
[ESSCertID.certHash={Hash}]
[ESSCertID.issuerSerial.issuer={GeneralName}]
[ESSCertID.issuerSerial.serialNumber={dec|hex}:{INTEGER}]
}]
[#MIME.Content-Type={MIME header}]
[#SigningCertificate]
[#SigningCertificate.certs={certIDs}]
[#SigningCertificate.signer_keypair={keypairIDs}]
<EOF>

```

なお、[]はオプション項目、{a|b}は選択項目とする。また、シャープ記号で始まる行はコメント行であり、コロン記号は直前の行の繰り返しを意味する。

以上の形式の設定ファイルを用いることにより、[TSP]に準拠した TSR/TST だけではなく、準拠しない TSR/TST も任意に生成することができる。例えば、ContentInfo.contentType を[TSP]で定義された id-signedData ではない値に設定することも可能である。

6 データベース

データベース(テストDB)には、テストを実行するために必要な情報(テストケース情報)と、テストの実行結果に関する情報を納めることができる。本項では、テストケース情報の設計内容を説明する。

データベースでは、テストケースに関するテーブルとして、以下に示す16種類を定義している。

表2 テーブル一覧

テーブル名	table name	格納される情報
証明書テーブル	cert	証明書の内容
証明書値テーブル	cert_value	証明書の各フィールド名と値
鍵ペアテーブル	keypair	生成された鍵ペアとハッシュ値など
テスト結果テーブル	test_result	テストの結果
テスト対象テーブル	test_object	テスト対象
テストスイート情報	ts_info	リポジトリとなるホストのディレクトリツリー全体
TSR テストケーステーブル	tsr_testcase	TSR テストケース(on-line テスト用)
TST テストケーステーブル	tst_testcase	TST テストケース(off-line テスト用)
TSQ テーブル	tsq	TSQ の ID
TSQ 値テーブル	tsq_value	TSQ の各フィールド名と値
TSR テーブル	tsr	TSR の ID
TSR 値テーブル	tsr_value	TSR の各フィールド名と値
HTTP ヘッダテーブル	http_header	HTTP ヘッダー(MIME-Type 等)
TST テーブル	tst	TST の内容
TSTInfo テーブル	tstinfo	TSTInfo の ID
TSTInfo 値テーブル	tstinfo_value	TSTInfo の各フィールド名と値

テーブル間の参照関係を示した ER 図を、以下に示す。tsq_value テーブル及び tsr_value テーブルには、前項で示した TSQ 及び TSR 設定ファイルでそれぞれ示される各フィールドの内容が登録される。

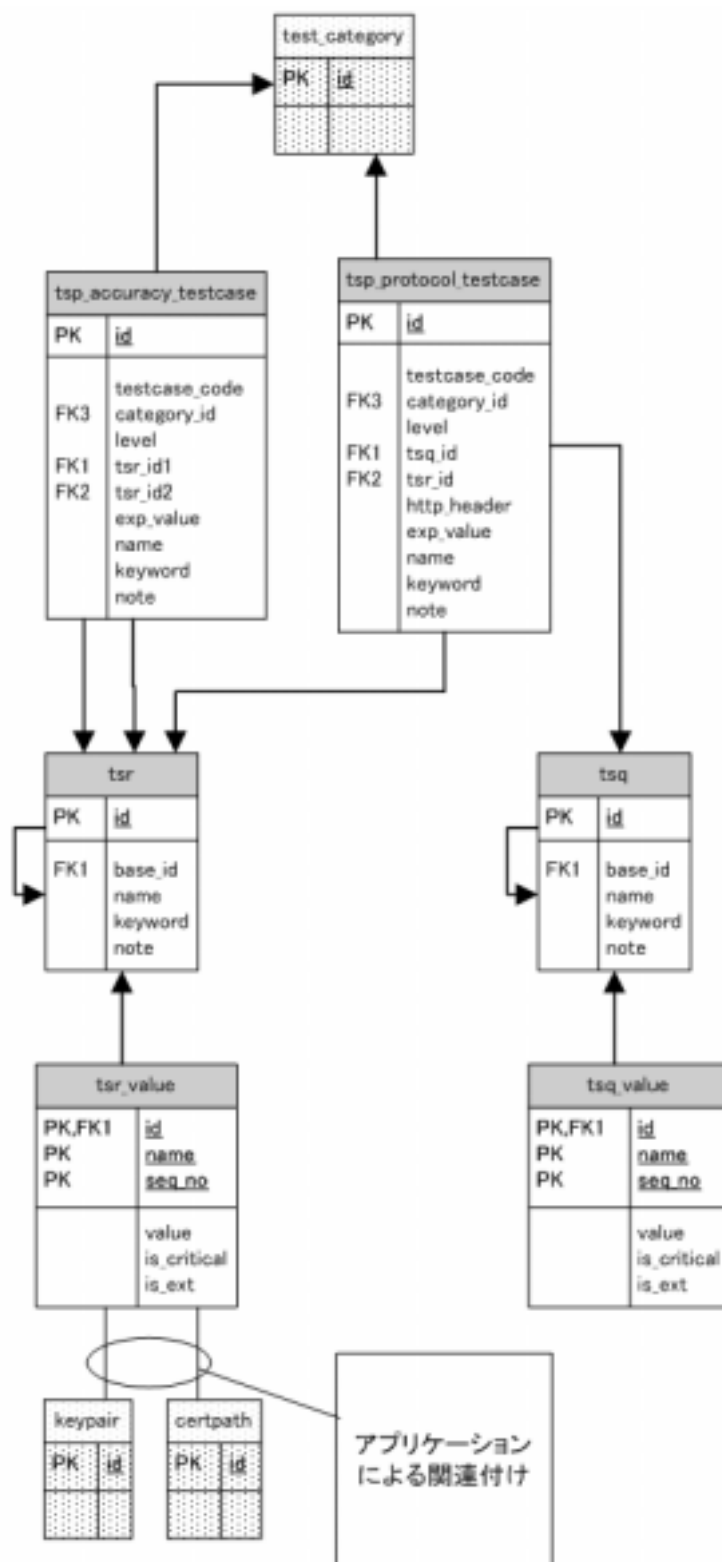


図5 テストケース関連テーブルのER図

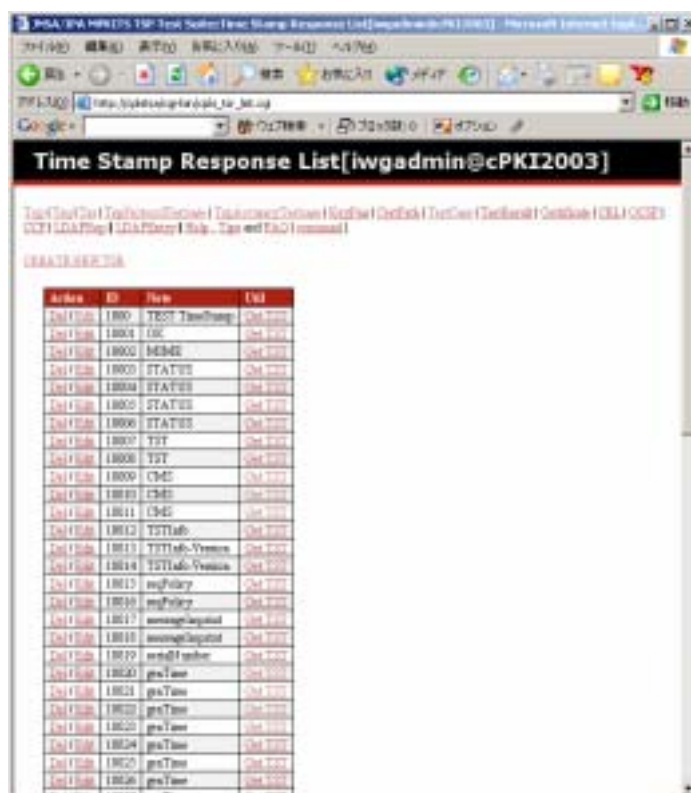
7 テスト管理 CGI

テスト(DB)管理 CGI とは、テストデータベースを閲覧・編集するツールである。この CGI を利用することで、テストを実行するために必要な情報（テストケース情報）をテスト実施者に正しく伝え、またテスト内容をカスタマイズ・修正したり、将来の標準化の変更に合わせて更新することができる。

テスト管理 CGI は、以下の URL を Web ブラウザから閲覧することで利用できる。

http://cpkitsa/cgi-bin/cpki_index.cgi

なおここで、cpkitsa とはテスト管理 CGI がインストールされたサーバーのホスト名を示す。テスト管理 CGI のサンプル画面を以下に示す。



Action	ID	Name	URL
TEST	1800	TEST TimeStamp	204.202.204.202
EXEC	1801	EXEC	204.202.204.202
MEME	1802	MEME	204.202.204.202
STATUS	1803	STATUS	204.202.204.202
STATUS	1804	STATUS	204.202.204.202
STATUS	1805	STATUS	204.202.204.202
STATUS	1806	STATUS	204.202.204.202
TST	1807	TST	204.202.204.202
TST	1808	TST	204.202.204.202
CME	1809	CME	204.202.204.202
CME	1810	CME	204.202.204.202
CME	1811	CME	204.202.204.202
TSTTab	1812	TSTTab	204.202.204.202
TSTTab-Verison	1813	TSTTab-Verison	204.202.204.202
TSTTab-Verison	1814	TSTTab-Verison	204.202.204.202
regPolicy	1815	regPolicy	204.202.204.202
regPolicy	1816	regPolicy	204.202.204.202
messageRegistat	1817	messageRegistat	204.202.204.202
messageRegistat	1818	messageRegistat	204.202.204.202
emailAdmin	1819	emailAdmin	204.202.204.202
proTime	1820	proTime	204.202.204.202
proTime	1821	proTime	204.202.204.202
proTime	1822	proTime	204.202.204.202
proTime	1823	proTime	204.202.204.202
proTime	1824	proTime	204.202.204.202
proTime	1825	proTime	204.202.204.202
proTime	1826	proTime	204.202.204.202

図6 テスト管理 CGI のサンプル

以上