経済産業省補助事業

平成１５年度情報セキュリティ対策推進事業

（電子商取引（ＥＣ）技術基盤の相互運用性に関する調査研究）

PKI の国際的相互接続実証実験報告書

平成１６年３月

（財）日本情報処理開発協会

— 目 次 —

1 目的

近年、世界の経済社会の幅広い分野において情報技術（デジタル技術）を高度に活用する動きが急速に進展しつつある。その中において、公開鍵認証基盤（Public Key Infrastructure:以下 PKI という）の整備は、各国で加速的に進められており、インターネット上で電子商取引を安全に且つ確実に実現する技術として、世界の経済社会に多大な貢献をもたらすものと期待されている。

しかしながら、PKI を使用した国際間相互認証に関する分野において、特にアジア地域においては現在 PKI を構築されつつあるが、PKI の構築方法は各国の事情に応じて構築されることから、統一されていないのが現状である。

そこで、平成 13 年度に引き続き、平成 14 年度には、韓国、シンガポール、チャイニーズ台北及び日本の間で、PKI コンポーネント間相互接続実験及び、PKI アプリケーションに関する実験を実施し、平成 13 年度の成果の活用及び課題の解決を行った。ここで得た成果を、アジアにおける推奨仕様として、他の関連する国際組織へ提案及び、普及活動を行っている。

しかし、アジア地域全体で実運用するためには、策定した仕様のガイドライン化、実証実験により顕在化した技術的な課題の解決、法制度・運用調査、アプリケーション層への展開などの問題が残されている。

一方、本格的なアジア圏における電子商取引を実現するためには、アジア諸国／地域において相互認証基盤の整備が急務である。上記テーマについて相互認証に関する標準を作成し、それら相互認証の標準に対する有効性を検証する実験が必要とされている。

本実証実験は、アジア圏における相互認証基盤の普及及びグローバルな電子商取引市場の創出を目指すものであり、韓国・シンガポール・チャイニーズ台北及びタイとの間で、認証局間相互接続及びそのテストに関する標準、証明書検証に関する標準及びアプリケーションインターフェースに関する標準に対し、実証実験を通して検証するものである。

特に今年度は、認証局間相互接続テストのガイドライン化、パス検証テスト効率化のためのツール整備、アプリケーションインターフェース仕様の範囲拡大及び認証局間相互接続におけるタイとの検証を目的に実施する。

2 プロジェクト概要

2.1　期間

平成 15 年 9 月 24 日～平成 16 年 3 月 7 日まで

2.2　体制

本実証実験の参加国／地域と推進組織を「表 2.1　体制」に示す。

表 2.1　体制

参加国／地域	組織
日本	日本 PKI フォーラム
韓国	Korea PKI Forum
シンガポール	PKI Forum Singapore
チャイニーズ台北	Chinese Taipei PKI Forum
タイ	Government Information Technology Services（以降 GITS と表記）

2.3　活動概要

顕在化した技術的な課題の解決を含む相互接続基盤の整備にあたり、以下の活動を実施した。

- 電子署名に関連する法律、政省令及び認証局の認可・認定に関わる文献調査を行い、国／地域による差分を抽出した。
- 実アプリケーションの実現について、ターゲットとして PAA（Pan Asia e-commerce Alliance）を適用するに当たっての法制度面、技術面、運用面での課題を調査した。実証実験による検証等、課題解決のための方策について考察を行った。
- 日本、韓国、シンガポール、チャイニーズ台北、タイの認証局間で相互接続が可能となる、認証局間相互接続及びそのテストに関する標準、証明書検証に関する標準、アプリケーションインターフェースに関する標準案を作成した。
- 各標準案について Korea PKI Forum、PKI Forum Singapore、Chinese Taipei PKI Forum、及び日本 PKI フォーラムの参加団体のメンバを含む各国／地域技術者と調整した。
- 認証局間相互接続及びそのテストに関する標準案について GITS、及び日本 PKI フォーラムの参加団体のメンバを含む各国技術者と調整した。
- 各標準案の有効性を、実証実験環境にて検証した。
- 実証実験の成果を標準案にフィードバックし、標準を作成した。

本報告書では、活動の概要と実証実験の結果を記述する。策定した標準は付録として添付する。

3 法制度・運用方式調査・分析

電子署名関連法令及び認証局の認可・認定に関する調査分析に関する活動内容概要について以下に記述する。

3.1　調査作業

国／地域による差分を抽出することを目的として、電子署名に関連する法律、政省令及び認証局の認可・認定に関わる文献調査を行った。

3.1.1 調査対象／手順

下記に示す各国／地域の文献につき、法的特徴及び電子認証局の認可・認定基準を明確にする目的で調査分析を行った。タイの電子商取引法を新たに調査対象とするとともに改定された文献を新たな調査対象とした。

表 3.1　調査を行った文献一覧

対象	国/地域	文献	改定/新規	頁数	開示
法制度分析調査	日本	電子署名及び認証業務に関する法律		17	公開
		電子署名及び認証業務に関する法律施行令		4	公開
		電子署名及び認証業務に関する法律施行規則	改定	20	公開
		電子署名及び認証業務に関する法律に基づく特定認証業務の認定に係る指針	改定	17	公開
		電子署名及び認証業務に関する法律に基づく指定調査機関等に関する省令		11	公開
		電子署名及び認証業務に関する法律に基づく指定調査機関の調査に関する方針	改定	9	公開
		特定認証業務の認定に係る調査表 (V3.1)	改定	34	公開
	韓国	電子署名法		12	公開
		電子署名法施行令		4	公開
		電子署名法施行規則		18	公開
		電子署名認証業務指針	改定	14	公開
		認定認証局の施設及び装置等に関する規程	改定	20	公開
		認定認証局の保護措置に関する規程		15	公開
		代理人等を通じた身元確認方法及び手順		4	公開
	シンガポ	電子商取引法 1999 年改訂版		35	公開
		電子取引（認証機関）規則		31	公開

	ール	認証局セキュリティガイドライン (V2.0)	改定	30	公開
	チャイニーズ台北	電子署名法		5	公開
		電子署名法施行細則	新規	5	公開
		外国認証局許可方法	新規	4	公開
		認証実務作業基準明記事項		8	公開
	香港チャイナ	電子交易条例		25	公開
		認定認証局実施規程（付録 CPS[1]記述要領）	改定	33	公開
	タイ	電子商取引法	新規	18	公開

3.2 調査結果

分析の結果、各国／地域の法制度及び認可・認定制度の差異が以下に記述するように明確になった。

3.2.1 法的面

各国／地域の法制度の概要について以下に記述する。

(1) 日本

電子署名の適用領域について規定していない。また利用者の真偽の確認に関する情報の適正な使用を規定している。

電子署名とは、署名者の特定及び改変の検証ができるものであると規定しているが、署名者の承認を示すものであるとまでは規定していない。このことは、印鑑あるいは人の手による署名と同一の考え方であり、電子署名に何らの差別を与えてはいない。

法上は技術中立的な記載となっている（法第 2 条第 1 項）が、電子署名及び認証業務に関する法律に基づく特定認証業務の認定に係る指針においては、署名者の特定及び改変の検証が可能な技術は非対称暗号技術だけであるため、当該技術を対象にした規定を行っている。

本人が電子署名を行った電磁的記録は、真正に成立したものと推定するとされている（法第 3 条）。このことは、認証事業者は必ずしも特定認証業務の認定を取得する必要は無いことを意味している。他方、契約自由の原則により、法が何らの制限も認定認証事業者以外の認証事業者と利用者の間に与えていない。

本人による電子署名が行われていれば、真正に成立したものと推定するとしている。ただし、本人だけが署名を行える状況を保持できるように、署名に必要な符号及び物件を適正に管理する必要がある。

利用者の義務の規定はなく、認定要件として、認証事業者に対して利用申込者への説明義務を負わせている。

[1] CERTIFICATION PRACTICE STATEMENT：認証業務運用規程。本報告書中では以降 CPS と記述する。

(2) 韓国

電子署名の適用領域について規定していない。また個人情報の保護を規定したうえ、加入者及び利用者の保護を規定している。

電子署名とは、署名者の特定が可能であり、当該電子文書に署名したことを明示するのに利用するために当該電子文書に添付又は論理的に結合した電磁的形態の情報との定義を規定しているが、署名者の承認を示すものであるとまでは規定していない。

さらに、一定の水準を規定し、それを満たすものを「認可電子署名」と定義している。また、署名の検証が可能な者を加入者及び利用者としている。

法上は技術中立的な記載となっているが、電子署名認証業務指針においては、非対称暗号技術を対象にした規程を行っている。（実証実験報告書 標準作成編 付録 4 法制度及び認証局の認可・認定に関わる調査報告書 参照）

法令が署名を必要とする場合、認可電子署名があれば、これを充足していると規定している。また、認可電子署名以外の電子署名の効力は当事者間の約定に基づくと規定している。（法第 3 条第 3 項）

署名の検証が可能な者を加入者及び利用者とし、加入者の電子署名生成情報が紛失･毀損若しくは盗難･流出し、又は毀損の危険を認めたときには、認可認証局に通知するとともに、利用者へ対しても告知する義務があるとしている。

外国の認証局との相互接続に関しては、外国と協定を締結した場合に認可電子署名又は認可証明書と同等の効力を認めている。

(3) シンガポール

商業的な観点からの妥当性の解釈を目的とし、政府サービスまでを含めた電子署名を規定している。また利用者に関する情報の秘密の保障を規定している。

電子署名及びデジタル署名を定義している。電子署名の定義として、認証または承認の意図で実行又は採用するものであるとしている。次にデジタル署名は非対称暗号技術及びハッシュ関数を使用した電子署名であると規定したうえで、署名者の公開鍵に対応する秘密鍵で変換されるかどうか及び変換後に変更されたかどうかを検証できるものであると規定している。

電子署名及びデジタル署名の定義を併記しているが、法上では非対称暗号技術が前提となるデジタル署名を用いた証明書について効力をみとめている。

当事者間で電子記録を生成、送信、受信、記憶、または別途処理することに関わるとき、法の一部の条項は協定により変えられるとしている。（法第 5 条）

法令が署名を必要とする場合、電子署名があれば、これを充足していると規定している。また一定の基準を満たす認定認証局において発行される証明書を安全な電子署名と定義し、安全な電子署名がなければ電子署名の確実性と整合性に関連するどのような推定もしないとしている（法第 18 条）。

外国の認証局との相互接続に関しては、推奨される信頼限度が、証明書に記載される等の条件が満たされれば外国の認証局が認められるとしている。

(4) チャイニーズ台北

電子文書に法的効果を与えており、電子文書の真贋を判断するものとして電子署名を規定している。電子文書の適用領域について規定していない。また個人情報の保護を規定している。

電子署名及びデジタル署名を定義している。電子署名を署名者の身分／資格及び電子文書の真贋の確認に用いられるものと規定しているが、署名者の承認を示すものであるとまでは規定していない。

電子署名及びデジタル署名の定義を併記しているが、法上では非対称暗号技術が前提となるデジタル署名を用いた証明書について効力をみとめている。

認証局は認証業務運用規程を作成し主管機関の審査決定後に公布しなければならない。このことにより、法の規定範囲において、電子署名に関して効力等に当事者間の契約により変更できる余地を残していない。

法令が署名を必要とする場合、相手の同意を得た上で、電子署名があれば、これを充足していると規定している。

外国の認証局との相互接続に関しては、許可を受けることにより、外国の認証局が発行する証明書に国内と同等の効力を認めている。

(5) 香港チャイナ

電子署名の適用領域を商業のみとは規定していない。また情報の守秘義務を規定している。

電子署名及びデジタル署名を定義しており、シンガポールとほぼ同様の定義となっている。

電子署名の定義として、認証または承認の意図で実行又は採用するものであるとしている。次にデジタル署名は非対称暗号技術及びハッシュ関数を使用した電子署名であると規定したうえで、署名者の公開鍵に対応する秘密鍵で変換されるかどうか及び変換後に変更されたかどうかを検証できるものであると規定している。

電子署名及びデジタル署名の定義を併記しているが、法上では非対称暗号技術が前提となるデジタル署名を用いた証明書について効力をみとめている。

デジタル署名が認可証明書によって裏付けられているか、あるいはその証明書の有効性（効力）の範囲内で生成したものであれば、その場合に限り、そのデジタル署名は要件を満たしていると規定されている（法第 6 条）。法は、認可された認証局以外から発行された証明書に何ら効果を認めていないため、何らの制限もまた認可認証事業者以外の認証事業者と利用者の間に与えていない。

外国の認証局との相互接続に関しては、認可認証局と匹敵する認可を外国で受

けていること等を条件とする。

(6) タイ

UNCITRAL[2]電子署名モデル法第 6 条に則した条文をもって信頼できる電子署名を規定している。即ち、例えば、以下の事項に合致する電子署名は信頼すべき電子署名であると見做されるものとしている。

- 署名生成データが、それらが使用される文脈の中で、署名者に関連付けられ、かつ他の何者にも関連付けられていないこと。
- 署名生成データが、電子署名を生成する時に、署名者の制御下にあったこと、かつ他の何者の制御下にもなかったこと。
- 電子署名の生成時後になされた電子署名への変更が検知可能であること。かつ
- 電子署名に関する法的要件の目的が情報の完全性に関する保証を与えることである場合、その情報に署名時後になされた変更が検知可能であること。

また、UNCITRAL 電子署名モデル法第 12 条に則した条文をもって、外国で発行された電子証明書及び外国でなされた電子署名の法的効力を規定している。即ち、例えば、電子証明書または電子署名は、以下の事項に関わりなく、法的に有効と見做されるものとしている。

- 電子証明書が発行される場所または電子署名が生成または使用される場所。または
- 電子証明書発行者または署名者の事業場所の位置

[2] UNITED NATIONS COMMISSION ON INTERNATIONAL TRADE LAW：国連国際商取引法委員会。本報告書中では以降 UNCITRL と記述する。

3.2.2 制度面

各国／地域認証局の認可･認定制度について、次の点からその概要を記述する。

- 監督官庁
- 準拠法
- 制度の位置付け
- 認可･認定の指定事項
- 認可･認定で要求される報告
- 指定保管情報
- 記録の保管期間
- 認可･認定の更新
- 廃業手続き

(1) 日本

(a) 監督官庁

総務省、法務省、経済産業省

(b) 準拠法

「電子署名及び認証業務に関する法律」第 6 条第 1 項、第 11 条

(c) 制度の位置付け

認定（ボランタリ評価）制度

(d) 認可･認定の指定事項

- 指定アルゴリズムの使用[RSA 方式(1024bit 以上)、RSA-PSS 方式(1024bit 以上)、ECDSA 方式(1024bit 以上)、DSA 方式(1024bit 以上)]
- 肩書き等の認定証明書における属性証明の部分は認定対象外である旨の表示

(e) 認可･認定で要求される報告

- 危殆化と災害復旧

(f) 指定保管情報

個人情報の管理に関する記録、加入者向け重要事項説明に関する記録、申請書、申請書添付資料･提示証明書等の写し、審査員名と決定日、申請拒否理由、加入者証明書及びその作成に関する記録、認証局秘密鍵及びその作成に関する記録（使用範囲･生成/保存･暗号モジュールの変更記録･活性化/非活性化･廃棄）、加入者秘密鍵の作成及び廃棄に関する記録と加入者受領書、失効申請書及びその判断に関する記録、失効決定者と決定日、失効申請拒否理由、失効情報及びその作成に関する記録、CPS 及びその変更に関する記録、事務取扱要領及びその変更に関する記録、組織図･指揮命令系統及びその変更に関する記録、外部委託契約、監査報告書（不定期に実施される監査の記録･定期的に実施される監査の記録･是正処置報告書）、認証業務設備室の入退記録（日時･場所･入退者･操作記録･警報記録）、認証業務設備へのアクセス記録（ネットワーク不正アクセス･

不正動作)、操作員毎の権限管理記録、認定に基づく設備の維持管理記録、事故障害記録（不正侵入･停止・不正操作)、書類廃棄記録

(g) 記録の保管期間

証明書の有効期間の終了から 10 年

(h) 認可･認定の更新

- 認定期間は 1 年
- 認定期限 30 日前までの更新申請

(i) 廃業手続き

- 主務大臣への通報（期限指定無し）
- 加入者への通報（60 日前）
- 全証明書の失効

(2) 韓国

(a) 監督官庁

情報通信部

(b) 準拠法

「電子署名法」第 8 条、第 18 条 3、第 15 条 1 項

「電子署名法律施行令」第 2 条

「電子署名法律施行規則」第 13 条 2 第 3 項、第 13 条 4

(c) 制度の位置付け

認定（ボランタリ評価）制度

(d) 認可･認定の指定事項

- KISA[3]の試験運営に合格した認証局が認定証明書を取得できる
- 指定アルゴリズムの使用[RSAKCDSA (1024bit 以上)、KCDSA (1024bit 以上)、楕円曲線方式(160bit 以上)、SHA-1 (160bit 以上)、SEED (128bit 以上)]
- 24 時間警備の実施
- FIPS[4]140-1(または 140-2)level 3

(e) 認可･認定で要求される報告

- 毎 6 ケ月の準拠性監査報告（登録局）
- 施設および装置設備の変更
- 15 日前までの変更届
- 15 日以内の譲渡･合併届

[3] KOREA INFORMATION SECURITY AGENCY：韓国情報保護振興院。本報告書中では以降 KISA と記述する。

[4] FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION：米国連邦政府情報処理標準。本報告書中では以降 FIPS と記述する。

(f) 指定保管情報

証明書情報、証明書業務情報、施設･設備管理情報、危機管理情報、バックアップ、監査記録

(g) 記録の保管期間

証明書及び証明書業務記録の保管期間は 10 年、ログ･運用記録は 2 年以上

(h) 認可･認定の更新

記載無し

(i) 廃業手続き

- 加入者及び長官への通報（60 日前）

(3) シンガポール

(a) 監督官庁

情報通信開発庁

(b) 準拠法

「電子取引（認証局規則）」序第 6 条

(c) 制度の位置付け

認定（ボランタリ評価）制度

(d) 認可･認定の指定事項

- 指定アルゴリズムの使用[IEEE[5]1363 準拠]
- FIPS140-1
- ISO15408

(e) 認可･認定で要求される報告

- 準拠性監査報告
- 事故

(f) 指定保管情報

認証局公開鍵、証明書情報、証明書業務情報

(g) 記録の保管期間

7 年以上

(h) 認可･認定の更新

- 認定期間は 1 年
- 3 ケ月前までの更新申請

(i) 廃業手続き

- 3 ケ月前までの CCA[6]への通報

[5] INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS,INC : 米国電気電子学会。本報告書中では以降 IEEE と記述する。

[6] CONTROLLER OF CERTIFICATION AUTHORITIES：シンガポールの認証局監督機関。本報告書中では以降 CCA と記述する。

・ 2 ケ月前までの加入者への通報
・ 広告(2 ケ月前)

⑷ チャイニーズ台北

(a) 監督官庁

経済部

(b) 準拠法

「電子署名法」第 11 条第 2 項

(c) 制度の位置付け

認可（ライセンス）制度

(d) 認可･認定の指定事項

・ 審査された CPS の書類番号の記載

(e) 認可･認定で要求される報告

記載無し

(f) 指定保管情報

記述無し

(g) 記録の保管期間

記載無し

(h) 認可･認定の更新

記載無し

(i) 廃業手続き

・ 経済部への通報（30 日前）
・ 加入者への通知（30 日前）
・ 他の認証局への業務引き継ぎ
・ 必要に応じて全証明書の失効

⑸ 香港チャイナ

(a) 監督官庁

情報技術サービス部

(b) 準拠法

「電子商取引条例(Cap. 553) 」第 33 条

(c) 制度の位置付け

認定（ボランタリ評価）制度

(d) 認可･認定の指定事項

・ CPS の提出
・ 認定外証明書の署名用認証局秘密鍵の分離
・ 独占禁止及び公正な比較広告
・ CPS 管理委員会の設置

・ ISO 15782-1 または FIPS140-1

(e) 認可･認定で要求される報告

・ 危殆化と災害復旧の通報
・ 3 就業日以内責任者異動の通報
・ 毎 6 ケ月の業務報告
・ 毎 12 ケ月の準拠性監査報告
・ CPS の変更通知及び重大な影響を与える事故通報

(f) 指定保管情報

契約書、CPS、証明書の発行等管理文書、運用記録（鍵生成設備へのアクセス･鍵と証明書のライフサイクル業務･鍵危殆化等事故･暗号装置の設置･認定証明書のサービス業務･CRL のサービス業務･認定認証局鍵ペアの生成･加入者鍵ペアの生成･施設管理）、発行証明書、認定証明書発行･失効･更新トランザクション記録、本人確認資料

(g) 記録の保管期間

記録の保管期間は少なくとも 7 年または長官が指定するこれより短い期間

(h) 認可･認定の更新

・ 認定期間は 1 年
・ 3 ヶ月以内の準拠性監査の実施と実施後 4 週間以内の提出要

(i) 廃業手続き

・ 長官への通報（90 日前）
・ 加入者への通知（60 日前）
・ 広告（60 日前連続 3 日間）
・ 全証明書の失効

⑹ タイ

現在、認可･認定制度を整備中である。

3.2.3 仕様面

各国／地域認証局の認可･認定仕様について、次の点からその概要を記述する。

- 特徴
- 鍵の使用目的
- 加入者秘密鍵の認証局保管
- 個人の存在確認書類
- 組織の存在確認書類
- 申請者への交付方法
- 代理人への交付方法
- 個人情報開示に係る加入者の同意取り付け

なお、電子署名に関連する法律、政省令及び認証局の認可・認定に関わる文献に記述の無い部分は、例示として、その国の代表的な認可・認定認証局の CPS から引用するものとし、具体的には、シンガポールの Netrust（Netrust 社の認証局)、チャイニーズ台北の TWCA（Taiwan-CA.COM 社の認証局)、及び香港チャイナの HKPCA（香港郵政の認証局）の CPS を参考にした。

(1) 日本

(a) 特徴

「電子署名及び認証業務に関する法律施行規則」の目次構成に基づいている

(b) 鍵の使用目的

- 加入者証明書への署名
- 認定認証局間の相互認証証明書への署名
- 日本政府ブリッジ CA との相互認証証明書への署名
- 自己署名
- リンク証明書への署名
- 捜査員向け証明書への署名
- CRL への署名
- リポジトリへの署名

(c) 加入者秘密鍵の認証局保管

不可

(d) 個人の存在確認書類

発行から 5 年未満の有効な証明書がある電子署名または次に示すもの

- 住民票の写し
- 戸籍の謄本または抄本
- 外国人登録法による登録原票記載事項証明書
- 上記に準ずるもの

の提出に加え、次のいずれか 1 つ以上の方法で行う

(i) 次のいずれか 1 つ以上の証明書等の提示を求める方法

- パスポート
- 運転免許証、船員手帳、海技免状、小型船舶操縦免許証、猟銃/空気銃所持許可証、戦傷病者手帳、宅地建物取引主任者証、電気工事士免状、無線従事者免許証、認定電気工事従事者認定証、特殊電気工事資格者認定証、耐空検査員の証、航空従事者技能証明書、運航管理者技能検定合格証明書、動力車操縦者運転免許証、教習資格認定証、検定合格証
- 外国人登録証明書
- 住民基本台帳カード
- 官公庁/独立行政法人/特殊法人の写真付き職員証明書

(ii) 申請書に押印した印鑑の印鑑登録証明書の提出を求める方法

(iii) 次のいずれかの証明書等による名あて人認証を伴う郵送サービスを用いる方法

- (i)と同等な方法
- 健康保険、国民健康保険、船員保険等の被保険者証、共済組合員証、国民年金手帳、国民年金、厚生年金保険若しくは船員保険に係る年金証書又は共済年金、恩給等の証書のいずれか 2 つ以上
- 上記のいずれか 1 つ以上と学生証、会社の身分証明書又は(i)以外の公的機関が発行した資格証明書であって写真をはり付けたもののいずれか 1 つ以上

(iv)（ⅲ）と同等な方法で主務大臣が認めたもの

(e) 組織の存在確認書類

商業登記規則による(電子署名法の適用外)

(f) 申請者への交付方法

(i) オンサイト

申請者が、規則 5 条 1 項に示す書類を提出し、規則 5 条 1 項 1 号に示す本人確認用書類を提示すると、調査票 2201 に基づき書類の真正性及び本人であることが確認され証明書が交付される

(ii) オフサイト

- 名あて人認証を行う郵便サービス

申請者が、規則 5 条 1 項と規則 5 条 1 項 2 号に示す書類を郵送すると、局留めの案内はがきが送られてくるので、それと規則 5 条 1 項 3 号に示す本人確認用書類を郵便局に提示すると、本人であることが確認され証明書が交付される

(g) 代理人への交付方法

可

(h) 個人情報開示に係る加入者の同意取り付け
要求されている

⑵ 韓国

(a) 特徴
- RFC[7]2527 を参考にしている
- システム及び装置の機能要件を指定している

(b) 鍵の使用目的
記述無し

(c) 加入者秘密鍵の認証局保管
記述無し

(d) 個人の存在確認書類

(i) 住民登録発行対象者の場合
- 住民登録証明書
- ただし書として、国家機関/地方公共団体/学校が発行した存在が確認できる証明書/書類

(ii) 住民登録発行非対象者の場合
- 国家機関/地方公共団体/学校が発行した存在が確認できる証明または
- 住民登録証明書謄本と法定代理人の a)の証明書

(iii) 在外国民の場合
- パスポート または
- 在外国民証明書

(iv) 外国人の場合
- 登録外国人記録票
- 但書として、パスポート または 身分証明書

(e) 組織の存在確認書類

(i) 法人の場合
- 非訟事件手続法による法人登記簿謄本または商業登記簿謄本
- 法人税法による事業者登録証明書
- 所得税法による納税番号が付与された文書または写し
- 付加価値税法による事業者登録証明書及び固有番号を付与された文書または写し

(ii) 法人以外の団体
- 代表者の存在確認書類
- 規則 13-2 条 1 項 3 号ただし書の団体の場合は 納税番号または固有番号

[7] REQUEST FOR COMMENTS：IETF（Internet Engineering Task Force）が発行する規格。本報告書中では以降 RFC と記述する。

が付与された文書または写し

(f) 申請者への交付方法

(i) オンサイト

申請者が、規則 13-3 条に示す書類を提出すると、規則 13-2 条に示す書類の真正性が確認され証明書が交付される

(ii) オフサイト

・ オンライン申請（電子金融取引に限る）

申請者の存在認証規則 6 条 2 項に示す情報（勘定、勘定暗証番号、口座暗証番号、住民登録番号、ワンタイムパスワードまたは本人しか知り得ぬ前述以外の 2 つ以上の情報）または存在認証規則 6 条 3 項に示す既に発行された電子署名の真正性が確認されると、証明書が交付される

(g) 代理人への交付方法

法人のみ可

(h) 個人情報開示に係る加入者の同意取り付け

記述無し

(3) シンガポール

(a) 特徴

・ BS[8]7799 と RFC2527 を参考にしている

(b) 鍵の使用目的

・ 署名用途と暗号用途は兼ねないこと

(c) 加入者秘密鍵の認証局保管

記述無し

(d) 個人の存在確認書類

CCA が認定した CPS に従う

Netrust の場合

・ シンガポール国民 ID カード（NRIC）

・ パスポート

(e) 組織の存在確認書類

CCA が認定した CPS に従う

Netrust の場合

・ 代表者の委任状

・ 企業登録（RCB）

・ ドメイン名

[8] BRITISH STANDARD：英国標準。本報告書中では以降 BS と記述する。

(f) 申請者への交付方法

(i) オンサイト

秘密鍵を保有している申請者が、CCA が認定した CPS に従って、書類を提出または提示すると、書類の真正性が確認され証明書が交付される

(ii) オフサイト

推奨されていない

(g) 代理人への交付方法

可

(h) 個人情報開示に係る加入者の同意取り付け

要求されている

⑷ チャイニーズ台北

(a) 特徴

・ RFC2527 に基づいている

(b) 鍵の使用目的

記述無し

(c) 加入者秘密鍵の認証局保管

認証局の判断による

(d) 個人の存在確認書類

TWCA の場合

・ レベル 3: 国民 ID カード
・ レベル 4: 財政的な証明書
・ 外国人: パスポート

(e) 組織の存在確認書類

TWCA の場合

・ レベル 3: 企業許可証明書
・ レベル 4: ドメイン名

(f) 申請者への交付方法

(i) オンサイト

TWCA の場合

申請者が、経済部が許可した CPS に従って、書類を提出または提示すると、書類の真正性及び本人であることが確認され証明書が交付される

(ii) オフサイト

TWCA の場合

不可

(g) 代理人への交付方法

TWCA の場合

法人のみ可

(h) 個人情報開示に係る加入者の同意取り付け

記述無し

⑸ 香港チャイナ

(a) 特徴

・ BS7799 を参考とし RFC2527 に基づいている

(b) 鍵の使用目的

記述無し

(c) 加入者秘密鍵の認証局保管

加入者の同意を必要とする

(d) 個人の存在確認書類

HKSAR[9]が認定した CPS に従う

・ 香港 ID カード（HKID）

・ パスポート

(e) 組織の存在確認書類

HKSAR が認定した CPS に従う

・ 会社設立契約書

・ 法人登録証明書

(f) 申請者への交付方法

(i) オンサイト

HKPCA の場合

申請者が、HKSAR が認定した CPS に従って、書類及び香港 ID カード（HKID）を提出または提示すると、書類の真正性及び本人であることが確認され証明書が交付される

(ii) オフサイト

HKPCA の場合

不可

(g) 代理人への交付方法

HKPCA の場合

不可

(h) 個人情報開示に係る加入者の同意取り付け

要求されている

⑹ タイ

現在、認可･認定制度を整備中である。

[9] GOVERNMENT OF THE HONG KONG SPECIAL ADMINISTRATIVE REGION：香港特別行政区。本報告書中では以降 HKSAR と記述する。

4 PAA に関する調査

4.1　調査目的

実アプリケーションの実現について、ターゲットとして PAA（Pan Asia e-commerce Alliance）を適用するに当たっての法制度面、技術面、運用面での課題について調査を行った。また、実証実験による検証等、課題解決のための方策について考察を行った。

4.2　調査対象

PAA の技術仕様、運用及び法制度に関するヒアリング調査を行った。調査対象を「表 4.1　調査対象一覧」に示す。

表 4.1　調査対象一覧　(順不同・敬称略)

調査対象	対象者	調査期間
Tradelink	Mr. Peter Stokes Senior Vice President Buisiness Development	平成 15 年 11 月 19 日 PAA 国際会議（上海）
KTNET	Ms. Kim Chaemee Manager Development Team	平成 15 年 11 月 19 日 PAA 国際会議（上海）
TEDI	日本電子貿易サービス 鍛冶俊彦 TEDI アドバンストネットワーク 渡辺浩吉	平成 15 年 11 月 27 日 個別訪問（東京）
Tradevan	Ms. Alicia Say Project Manager	平成 16 年 2 月 11 日 個別訪問（韓国ソウル）
CrimsonLogic	Mr. Ser Tee Hong Manager Business Systems	平成 16 年 2 月 11 日 個別訪問（韓国ソウル）

また、PAA に関する調査結果をもとに PAA と連携した実証実験に向けたパイロットプロジェクトに関する国内調整及び海外調整を行い、実施計画のための考察を行った。

国内調整に関する調整対象を「表 4.2　国内調整対象」に示す。

表 4.2　国内調整対象 (順不同・敬称略)

調査対象	対象者
財務省	関税局担当者
日本商工会議所	国際部　星川 孝宜氏
東京商工会議所	経営・福利厚生支援部　柳本　満生氏

また、海外調整に関する調整対象を「表 4.3　海外調整対象」に示す。

表 4.3　海外調整対象

調査対象	対象者
iDA	Mr. Jansen CHUA Assistant Director E-Business Transformation Ms. Evelyn Ong E-Business Transformation
CrimsonLogic	Mr. Ser Tee Hong Manager Business Systems Mr. Chuang Shyne Song Business Development Director
KISA	Dr. Chuhwan Park, Ph.D Director Electronic Transaction Security & Data Protection Division Mr. Seok-Lae Lee Electronic Transaction Security & Data Protection Division
KTNET	Ms. Kim Chaemee Manager Development Team Mr. Min, Cheol Hong eTrade Certification Authority / Team Manager

4.3　調査内容

(1) 技術面に関する調査

PAA のサービス向けに証明書を発行する認証局について以下の点のヒアリングを行い、問題点の抽出と「Asia PKI Interoperability Guideline」の適用可能箇所の選定を行った。

- 認証局の自己署名証明書及び EE 証明書、CRL/ARL に関する証明書プロファイルの実態
- 認証局間の相互接続の実態

(2) 運用面に関する調査

PAA サービス体系を整理し、以下の点のヒアリングを行い、PKI 活用に関する問題点の抽出と「Asia PKI Interoperability Guideline」の適用可能箇所の選定を行った。

- 認証局が発行する証明書の適用範囲
- 電子署名、暗号化等証明書の利用方法
- 署名検証者におけるパス検証に関する実態

(3) 法及び認定制度面に関する調査

国際的なサービスを支えるための法制度及び認証局に関する認定制度について PAA の現状を調査した。

(4) PAA 連携実験に向けたパイロットプロジェクトに関する国内調整

電子原産地証明書を実用化するために必要となる要件について国内調整を実施し、以下の内容について討議を行った。

- 原産地証明書の発行状況について
- 原産地証明書の真正性の確保の現状について
- 電子原産地証明書のメリットについて
- 電子原産地証明書の実現方法について
- 原産地証明書の FTA への対応状況について

(5) PAA 連携実験に向けたパイロットプロジェクトに関する海外調整

電子原産地証明書を実用化するために必要となる要件について海外調整を実施し、以下の内容について討議を行った。

- アジア PKI としての今後の推進戦略について
- 電子原産地証明書パイロットプロジェクトの構成について
- 電子ドキュメントに必要となる要件（船荷証券、原産地証明）について

4.4 調査結果

以下に調査結果の概要を述べる。

4.4.1 技術面の調査結果

(1) 証明書プロファイル

証明書プロファイルについては、2 ヵ国の認証局（CA1、CA2 とする）を対象とした調査を実施した。

それぞれの認証局から認証局の自己署名証明書及び EE 証明書をサンプルとして入手し比較を行った。CRL については入手することができなかったため、調査対象としなかった。また、CA1 については、自己署名証明書を持たない運用を行っているため、自己署名証明書は入手できていない。

PAA が定める証明書プロファイルの基本項目の設定を「表 4.4　PAA 証明書プロファイルの基本項目」に示す。

表 4.4　PAA 証明書プロファイルの基本項目

PAA プロファイル要件必須項目	CA 自己署名証明書	EE 証明書
バージョン	○	○
シリアル番号	○	○
署名アルゴリズム	○	○
発行者	○	○
有効期間	○	○

サブジェクト	○	○
公開キーアルゴリズム	○	○
機関キー識別子		○
サブジェクトキー識別子	○	○
キー使用方法		○
証明書ポリシ		○
基本制限	○	
CRL 配布ポイント		○

CA 自己署名証明書に関して比較を行った CA2、およびアジア PKI のプロファイル共、PAA のプロファイル要件（必須項目 : 9 項目）を満たしていると言える。EE 証明書に関しては、必須項目 12 項目の内未定義なものが、CA1 のプロファイルに 1 項目、CA2 のプロファイルに関しては 3 項目存在する。また、必須項目で定義内容に不足のある項目は CA1 とアジア PKI プロファイルにそれぞれ 1 項目存在した。該当する項目を「表 4.5　EE 証明書プロファイル要件の未達成項目」に示す。

表 4.5　EE 証明書プロファイル要件の未達成項目

項目	要件	未達成プロファイル
サブジェクトキー識別子	SHA1 アルゴリズムの 160bit の公開鍵ハッシュ値	・CA2 のプロファイルは未定義
証明書ポリシ	「policyID」を含むこと	・CA1 のプロファイルは未定義 ・CA2 のプロファイルは未定義
CRL 配布ポイント	「ディレクトリ名」か「URI」を含むこと	・CA2 のプロファイルは未定義
キー使用方法	Identification Certificate (デジタル署名) : digitalSignature, (nonRequdiation) Secure E-Mail Certificate (暗号化、デジタル署名) : keyEncipherment, dataEncipherment	・CA1 プロファイル dataEncipherment 未定義 ・PKI プロファイル keyEncipherment 未定義 dataEncipherment 未定義

これらの未達成項目や不足項目に関しては、要件を満たすために新しい証明書プロファイルを導入する必要がある。

結果を以下にまとめる。

1. CA 自己署名証明書、EE 証明書双方の「発行者」「サブジェクト」において、すべての証明書が要件を満たしているものの、強く推奨されている UTF8String にてエンコーディングされている証明書はアジア PKI の証明書のみである。要件に準拠するのであれば CA1 や CA2 のプロファイルはエンコーディ

ングの種別をアジア PKI における DN マッチングに関する考察等を基にしたUTF8String に変更することが望ましい。

2. CRL 配布ポイント、証明書ポリシが無いことは、パス検証を行う際に致命的である。PAA プロファイルでは必須項目となっているため、それを厳密に守った運用とする必要がある。（EE 証明書）

3. キー使用方法として PAA プロファイル要件では署名用証明書と暗号用証明書を同時に設定することを必須としているが、署名用証明書と暗号用証明書は分ける必要があるという考え方もあり、要件に対して検討が必要と考えられる。（EE 証明書）

4. CA 自己署名証明書の「キー使用方法」「CRL 配布ポイント」について PAA プロファイル要件では任意となっている。しかし、アジア PKI のプロファイルではこの項目を必須項目としており、PAA のプロファイル要件についても検討が必要と考えられる。

5. 以上のことから、「Asia PKI Interoperability Guideline」をパス検証を可能とするための証明書プロファイル、或いは CRL/ARL プロファイルへ適用することで PAA のサービスのセキュリティを向上させることが可能であると考えられる。

⑵ 認証局間の相互接続の実態

PAA メンバは、認定された CA から PAA 用の CA 自己署名した公開鍵証明書を受領し、これを PAA が定める TrustList に登録することで認証局の相互接続を実現している。

PAA 間で交換される全てのメッセージは、TrustList に登録された公開鍵証明書に基づいて検証される。

また、CA の自己署名した公開鍵証明書の代わりに、PAA メンバの公開鍵証明書を直接 TrustList へ登録することも許されている。

4.4.2 運用面の調査結果

(1) 認証局が発行する証明書の適用範囲

PAA サービスの概要を「図 4.1　PAA サービスの概要」に示す。

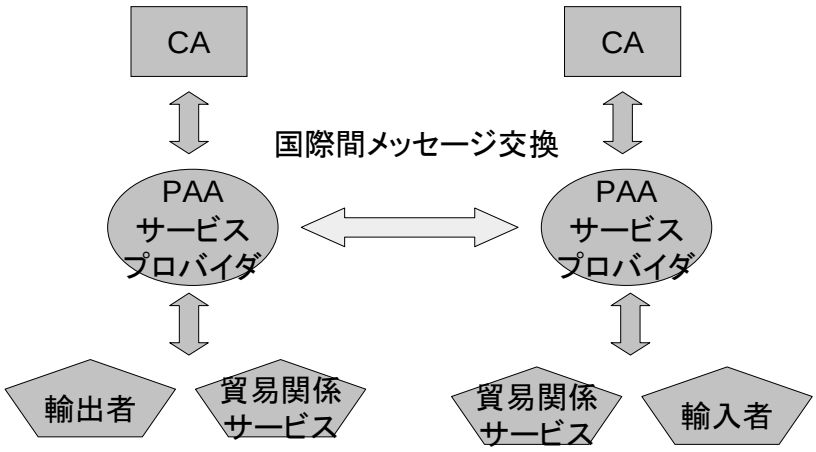


図 4.1　PAA サービスの概要

まず、輸入者と輸出者は、それぞれの国の PAA サービスプロバイダと利用契約を締結し、PAA サービスを利用できる状態になることが必要である。

PAA においては、各利用者と PAA サービスプロバイダ間のメッセージ交換方法については、セキュアであることを要求するが、それを実現するための具体的な方法については、一切規定していない。

各国の認証局は、PAA サービスプロバイダに対しては証明書を発行する必要があるが、エンドユーザである輸出者及び輸入者にたいしては証明書を発行することは義務付けられていない。

従って、PAA サービスプロバイダ間はセキュリティを確保したメッセージ交換を行っているが、PAA サービスプロバイダ・輸入者及び輸出者間では必ずしもセキュリティを確保したメッセージ交換が行われているとは限らない。

(2) 電子署名、暗号化等証明書の利用方法

PAA サービスを実現するシステムの概要を「図 4.2　PAA システムの概要」に示す。

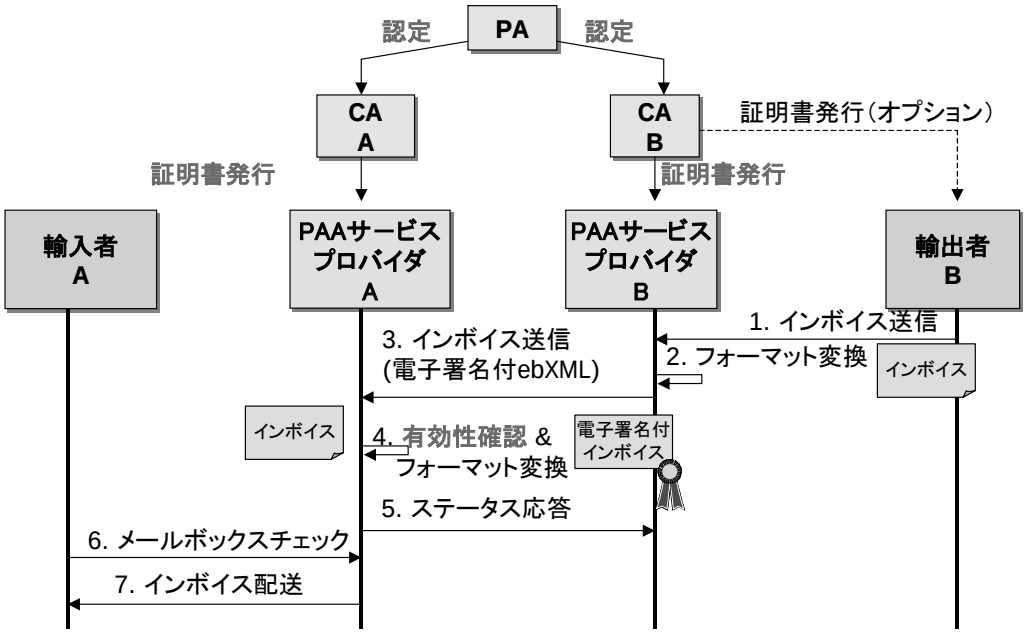


図 4.2　PAA システムの概要

PAA においては、特定の文書の内容を安全に交換できるようにするために、特定の文書、たとえば、インボイス[10]、の PAA フォーマットを定めている。

利用者からそれぞれの PAA サービスプロバイダが規定する方法で受け取ったインボイスは、PAA サービスプロバイダによって、PAA フォーマットに変換される。

PAA フォーマットに変換されたインボイスの内容は、PAA が定める「信頼できる国際メッセージ交換」によって、相手方の PAA サービスプロバイダに送信される。このメッセージには、PAA サービスプロバイダによって電子署名が行われている。

PAA プロバイダから受信したもう一方の PAA プロバイダは、電子署名を検証し、PAA フォーマットのインボイスをそのプロバイダ固有、あるいは、利用者が要望するフォーマットに変換して、利用者に配送することになる。

PAA サービスプロバイダと輸入者・輸出者の間のメッセージ交換については、輸入者及び輸出者が証明書を利用していないため、電子署名が行われていないのが現状である。

[10] 送り状：売主が買主宛に作成する約定品の出荷案内書、物品明細書、価格計算書、代金、請求書を兼ねた商用書類

(3) 署名検証者におけるパス検証に関する実態

証明書プロファイルに関する「4.4.1(1)証明書プロファイル」でも述べているが、CRL の運用がなされていない認証局があると思われる。また、証明書の有効性を検証するために必要な項目が欠落している証明書を発行している認証局があると思われる。

このような状況の中、電子署名の検証を PAA サービスプロバイダが行ったとしても厳密な意味で有効性を検証することは不可能であると考えられる。PAA における電子署名の有効性を高めるためには、「Asia PKI Interoperability Guideline」の証明書プロファイル及び CRL プロファイルを適用し、証明書検証におけるパス構築及びパス検証を厳密に行うことを可能とする基盤整備が必要である。

4.4.3 法及び認定制度面の調査結果

(1) 電子署名の法的拘束力の確保

PAA においては、電子署名の法的拘束力を確保するために、PAA サービスに関わるすべての利用者、サービスプロバイダが PAA Club Agreement によって拘束されることが同意され、さらに、PAA Club Agreement においては、PAA が 認定した CA が発行した PAA 用の公開鍵証明書によって検証された電子署名の記録は、決定的証拠 conclusive evidence とすることが合意されている。法的枠組みに関して「図 4.3　電子署名に関わる法的枠組み」に示す。

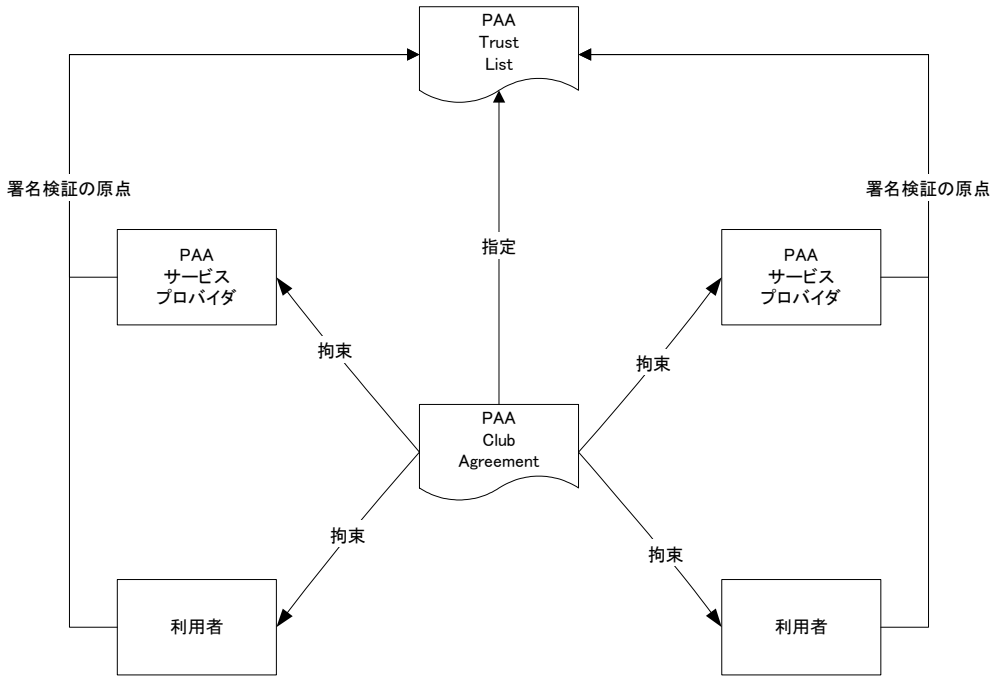


図 4.3　電子署名に関わる法的枠組み

(2) PAA　CA の認定

電子署名ついてのセキュリティを確保することは、インターネットを利用した国際間の電子メッセージ交換をサービスの中心に据えている PAA にとっては、最大の課題である。多くの PAA メンバが、CA に関わる事業を子会社や関連会社として保有しているため、PAA としての CA 会社を設立するのではなく、セキュリティレベルを統一したうえで CA を認定し、公開鍵証明書の発行は各 CA に任せるという形態を選択した。認定方式に関して「図 4.4　PAA　CA の認定方式」に示す。

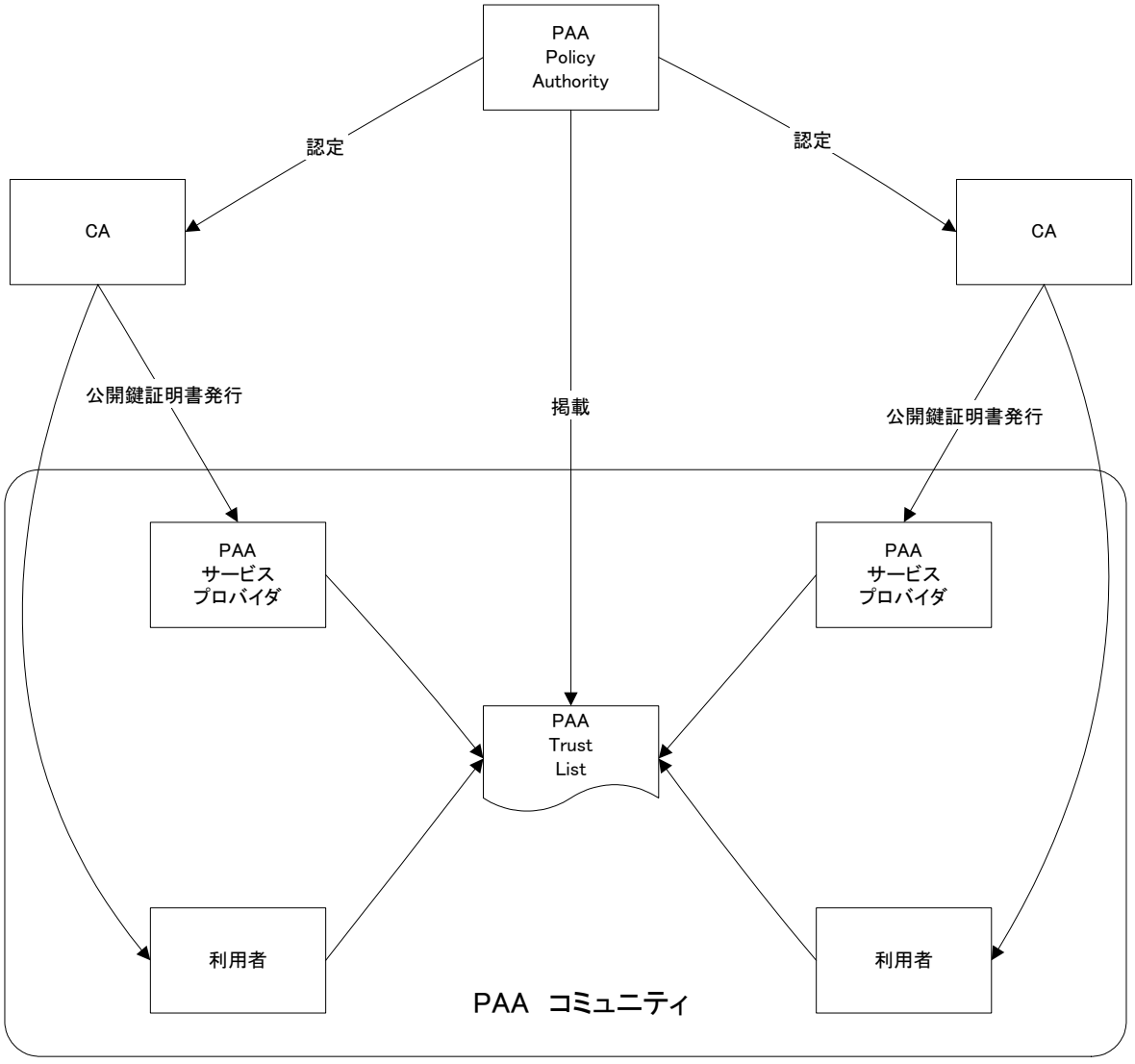


図 4.4　PAA　CA の認定方式

(3) Certificate Policy

原則として、各国の認定認証業務と同等のセキュリティレベルを確保していることが認定されるための基準となる。

発行対象として、以下のものが指定されている。

- 自然人
- 法人
- NPO
- 政府機関

また、検証者(Relying Party)としては、すべての個人、組織としている。

この Certificate Policy 策定にあたっては、PAA メンバ各国の特定に認定認証に関わるセキュリティレベルの調査の上、PAA の事業目的に合うように設定された。

(4) PAA Certificate Authority

PAA Certificate Authority の役割は、PAA が認定した Trust Certificates の管理を実施することである。

PAA によって認定された CA は、認定された自己署名した公開鍵証明書を PAA Certificate Authority に渡し、Trust List において公開される。PAA Certificate Authority は、Trust List に掲載される Certificate の信頼を各 CA の CPS が PAA Certificate Authority に反しないことの検査、各 CA の運用が CPS のとおりに行われていることの外部監査機関による報告書に基づき、PAA 用の公開鍵証明書の認定を行う。

認定された証明書により検証された電子署名の法律的な効果については、Club Agreement において規定されている。

4.4.4 パイロットプロジェクトの実施計画に向けた考察

現在の PAA サービスは、PKI による国際間メッセージ交換であり、アジア PKI フォーラムが目指す、国際間の PKI 相互運用性の確保と大きく重なるところがある。ほとんどの PAA メンバは、貿易手続きにおける税関とのゲートウェイサービスを提供しており、半官半民の性格をもった企業である。PAA の仕様は、アジアにおける標準 PKI 仕様にも大きく影響を与えるものであり、アジア PKI フォーラムが定めるガイドラインと連携できていることが望ましい。

表 4.6　アジア PKI フォーラムと PAA の特性

	アジア PKI フォーラムの強み	アジア PKI フォーラムの弱み
PAA の強み	政府の協力 国際的な知名度 該当分野における専門家の確保	実用パイロットの構築 電子取引全体をバーする契約面の整備 実用アプリケーションの提供
PAA の弱み	国際間 PKI の相互接続性の技術ガイドラインの確立	収益性の確保

	利用者拡大時の検証方法の確立	

現在の PAA における公開鍵証明書の利用は、サービスプロバイダにのみにしか適用されておらずエンドユーザの公開鍵証明書は利用されていないので、公開鍵証明書の検証は簡易的な方法で代替が可能である。しかし、近い将来は、エンドユーザの公開鍵証明書の検証も必要となり、その場合には、現状以上の検証方法が必要となる。PAA におけるアジア PKI ガイドラインの適用箇所に関して図 4.5　PAA におけるアジア PKI ガイドライン適用領域に示す。

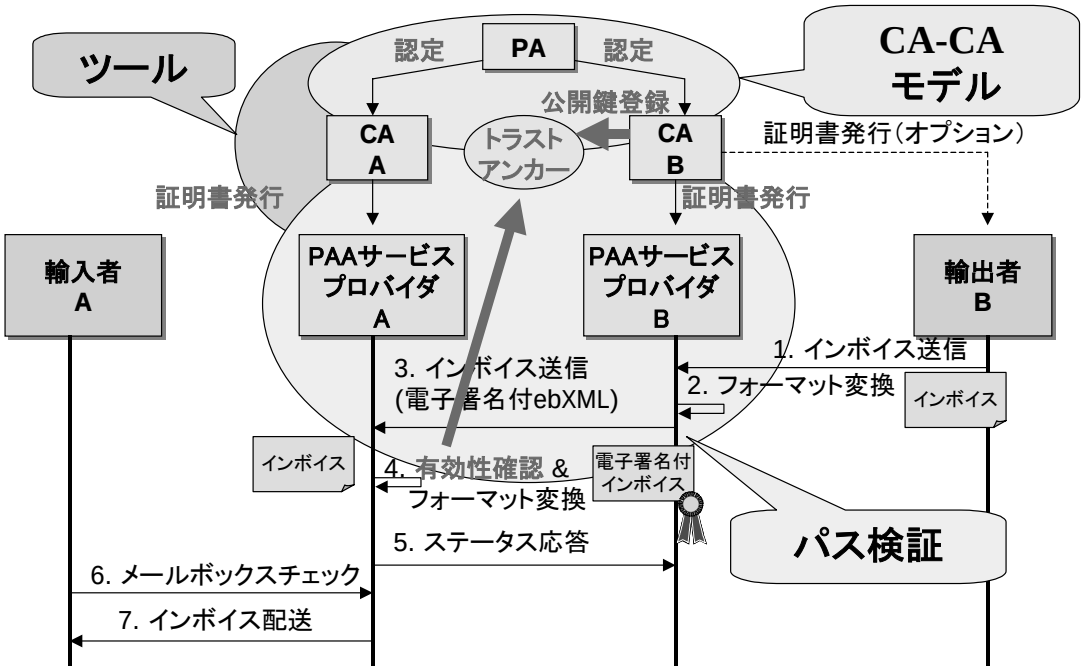


図 4.5　PAA におけるアジア PKI ガイドライン適用領域

まずは、PAA において利用されている公開鍵証明書について、その証明書プロファイルで国際間の相互運用が可能かどうかをチェックする必要があり、次に、電子署名は、証明書に基づいて正しく検証できて初めて、署名としての効果を発揮するものであるので、PAA のサービスプロバイダが実施する証明書の検証方法についても、以下の点でアジア PKI ガイドラインが貢献できる要素がある。

- 認証局の相互接続の形態
- 証明書及び失効リストのプロファイル
- 証明書の検証項目と検証方法
- 署名者及び署名検証者におけるアプリケーションと PKI コンポーネントとのインターフェース

また、現在の PAA は、各プロバイダが認証局のルート証明書を保存した「トラストリスト」を共有することによりトラストアンカーを設定しているが、このトラ

ストアンカーについての安全なあり方についても以下の点でアジア PKI ガイドラインが貢献できると思われる。

- 認証局の相互接続の形態に関する提言及び支援
- 法及び認定制度に関する支援

PAA の立場からは、アジア PKI フォーラムとの連携メリットにはどのようなものが考えられるであろうか。

PAA は、すでに国をまたがる異なる認証機関を利用して相互認証をすでに実現している。その面から見れば、アジア PKI ガイドラインの採用は、当面のことだけを考えれば、かえって負担が増えるだけで、目に見えるメリットを提供することは難しい。

しかし、これから、タイ、スリランカ、インドネシア、あるいは、その他の地域からも PAA メンバが加わり、その地域ごとに、認証機関が加わってきた場合、現在のような少数を前提とした、メンバにおける電子署名検証の確認方法、認証機関の認定方法では、煩雑さ、精度の低下が危惧される。

また、現在は、サービスプロバイダの電子署名のみを前提としたサービスが提供されているが、近い将来、エンドユーザ、すなわち、輸出者、輸入者、あるいは貿易関係のサービス提供者が署名を行う文書を取り込むことが必要となる場面が想定される。たとえば、「電子船荷証券」（B/L）や「原産地証明書」（C/O）なのである。原産地証明は、2 カ国間における FTA（貿易自由協定）の進展に伴い、貿易手続きのスピードアップを実現するために、その需要が大きく拡大することが想定される。

PAA に対してアジア PKI ガイドラインを適用した場合の効果について「図 4.6 PAA におけるアジア PKI ガイドラインの利用効果」に示す。

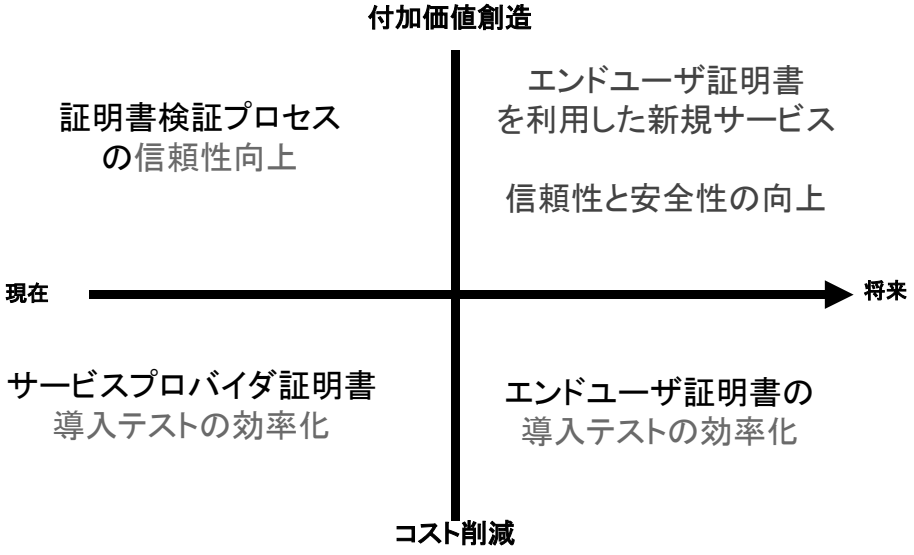


図 4.6　PAA におけるアジア PKI ガイドラインの利用効果

5 標準作成

5.1　概要

本標準作成作業は、日本、韓国、シンガポール、チャイニーズ台北、香港チャイナ、タイの認証局間で相互接続が可能となる、認証局間相互接続及びそのテストに関する標準、証明書検証に関する標準、アプリケーションインターフェースに関する標準を策定する。

認証局間相互接続及びそのテストに関する標準では、電子署名に関連する法律、政省令及び認証局の認可・認定に関わる文献調査を行い、国／地域による差分を抽出する。各国／地域の法律、政省令などに変更が合った場合のはその反映を含む。

アプリケーションインターフェースに関する標準では、実アプリケーションの実現について、ターゲットとして PAA（Pan Asia e-commerce Alliance）を適用するに当たっての法制度面、技術面、運用面での課題を調査する。実証実験による検証等、課題解決のための方策について考察を行う。

各標準作成では、標準案を作成し、日本 PKI フォーラムに参加している団体のメンバを含む技術者のレビューを受ける。日本、韓国、シンガポール、チャイニーズ台北、日本で構成される JKSTIWG メンバ及びタイの認証局関係者に提出してレビューを受けるとともに、調整後の標準案、コメントを基に標準案作成の担当者を含むメンバで検討し、認証局間相互接続及びそのテストに関する標準、証明書検証に関する標準、アプリケーションインターフェースに関する標準を作成することとする。

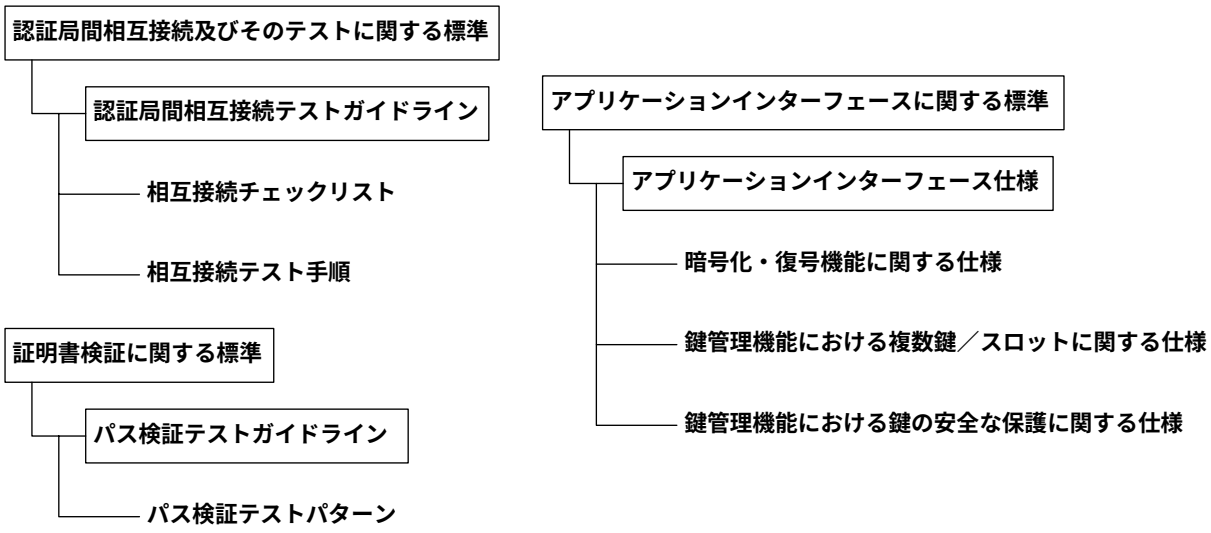


図 5.1 標準書の構成

5.2　認証局間相互接続及びそのテストに関する標準

相互接続基盤の整備に際し、認証局間相互接続及びそのテストに関する標準を定める。相互接続チェックリスト及び相互接続手順を含む。

5.2.1 認証局間相互接続テストガイドライン

(1) 認証局間相互接続テストガイドラインの内容

(a) 相互接続チェックリスト（付録 1−1 3 章）

本標準は以下のチェック項目及びそのリファレンスを含む。

- CA-CA モデル
- コンポーネント間インターフェース
- 証明書・失効リストプロファイル
- LDAP プロファイル

以下にチェックリストを示す。

(i) CA−CA モデルの事前確認事項

(ｲ) 相互接続先認証局に関する基本情報の確認

以下に示す基本情報を事前確認する。

- 相手認証局名称、住所、電話番号、FAX 番号、
- 相手側の窓口となる人の名前及び連絡先

(ﾛ) 相互接続モデルに関する確認事項

IWG では 2 種類の相互接続モデル(CC モデルと CR モデル)を定義しており、どちらかの相互接続モデルを選択することを推奨している。

(ﾊ) 認証局ソフトウェアの機能要件に関する確認事項(CC モデルの場合)

CC を選択する場合には以下の機能が必須である。

- PKCS#10 の発行及び受け付け
- crossCertificatePair の作成及び配布

(ﾆ) 相互接続先の認証局の確認事項

CC モデルにおける相互接続先の認証局としてルート認証局を推奨する。

(ﾎ) CRL 配布方法の確認

CRL の配布方法はそれぞれの PKI ドメインのポリシによって異なるため、下記のいずれかであることを確認する。

- CA は 1 つの完全 CRL を発行する
- CA は分割 CRL を発行する
- CA 1 つの CRL と 1 つの ARL を発行する
- CA は分割された CRL と ARL を発行する

(ii) コンポーネント間インターフェースの事前確認事項(共通)

(ｲ) 証明書および CRL のプロファイル

証明書および CRL のプロファイルは X.509 および RFC3280 に従っていることを推奨する。

(ﾛ) 証明書検証アルゴリズム

証明書検証アルゴリズムが RFC3280 または X.509 に従っていることを推奨する。

(ﾊ) 証明書検証者

証明書は EE または VA によって検証される。

(iii) 認証局間インターフェースの事前確認事項

(ｲ) 相互認証証明書リクエストフォーマット

PKCS#10 version1.0[11]

(ﾛ) Fingerprint の送付方法

電子メールまたは FAX[12]

(ﾊ) 相互認証証明書リクエストの確認方法(Proof of Possession)

特に確認方法は規定しないが、確認は行わなければならない。

(iv) CA-EE 間インターフェースの事前確認事項

(ｲ) EE への証明書配布フォーマット

PKCS#12 (秘密鍵を含む)[13]

(v) EE(VA)-リポジトリ間インターフェースの事前確認事項

(ｲ) リポジトリへのアクセスプロトコル

LDAP を推奨する。

(vi) ルート認証局証明書プロファイルの事前確認事項

(ｲ) Version (必須)

いくつかの拡張領域を利用しているので、バージョンは 3 とする。但し

[11] 拡張領域を含まないこと

[12] これは実証実験仕様であり、実際に相互接続を行う場合には手渡しまたは書留などで送付するのが望ましい。

[13] これは実証実験仕様であり、実際には秘密鍵を誰が生成するかはその PKI ドメインによって異なる。認証局が EE の秘密鍵を生成する場合には秘密鍵を安全に配布する手段が必要になる。

ルート認証局に関しては V1 でもよい。その場合には認証局は同じ subject に同じ DN を持つ異なる鍵を持つ証明書を発行してはならない。

(ﾛ) Serial Number (必須)

20 オクテット以下の一意な正整数

(ﾊ) Signature (必須)

署名アルゴリズムは sha1WithRSAEncryption を利用する。

(ﾆ) Issuer (必須)

この領域には空でない Distinguished Name を記載しなければならない。本プロファイルでは本領域を UTF8 でエンコードすることを推奨している。しかし、レガシーシステムへの相互運用性確保のために PrintableString の利用も許容する。また、リポジトリとして LDAP を推奨していることから、記載される Distinguished Name はその DIT に対応していなければならない。

(ﾎ) Validity (必須)

この領域には証明書の有効期間を記載する。この有効期間は様々なポリシによって異なるが、認証局は認証局証明書の有効期限を越えた証明書を発行してはならない。

(ﾍ) Subject (必須)

この領域には空でない Distinguished Name を記載しなければならない。本プロファイルでは本領域を UTF8 でエンコードすることを推奨している。しかし、レガシーシステムへの相互運用性確保のために PrintableString の利用も許容する。また、リポジトリとして LDAP を推奨していることから、記載される Distinguished Name はその DIT に対応していなければならない。ルート認証局証明書の本領域は Issuer と同じでなければならない。

(ﾄ) subjectPublicKeyInfo (必須)

RSA アルゴリズム(鍵長は 2,048bit 以上)を利用する。

(ﾁ) authorityKeyIdentifier (任意)

ルート認証局証明書ではこの領域は通常では必要ない。この領域を利用する場合には RFC3280 に示される最初の計算方法(160bit SHA-1 hash)を用いること。

(ﾘ) subjectKeyIdentifier (必須)

RFC3280 に示される最初の計算方法(160bit SHA-1 hash)を用いること。

(ﾇ) keyUsage (任意, Critical)

本領域を利用する場合には、keyCertSign、cRLSign を含む。

(ﾙ) Certificate Policies (任意)

本領域を利用する場合には policyID は必ず含むこと。

(ﾌ) subjectAltName (任意)

証明書の持ち主に関する情報として電子メールアドレスや漢字やハングル文字などを含めたい場合には本領域を利用する。

(ﾜ) issuerAltName (任意, Non-Critical)

証明書の発行者情報として電子メールアドレスや漢字やハングル文字などを含めたい場合には本領域を利用する。

(ｶ) basicConstraints (必須, Critical)

ルート認証局証明書であるので本領域は必ず TRUE とする。

(ﾖ) cRLDistributionPoints (任意, Non-Critical)

本領域を利用する場合には、distributionPoint.fullName を DN または URI で必ず含むこと。

(vii) 相互認証証明書プロファイルの事前確認事項

(ｲ) Version (必須)

いくつかの拡張領域を利用しているので、バージョンは 3 とする。

(ﾛ) Serial Number (必須)

20 オクテット以下の一意な正整数

(ﾊ) Signature (必須)

署名アルゴリズムは sha1WithRSAEncryption を利用する。

(ﾆ) Issuer (必須)

この領域には空でない Distinguished Name を記載しなければならない。本プロファイルでは本領域を UTF8 でエンコードすることを推奨している。しかし、レガシーシステムへの相互運用性確保のために PrintableString

の利用も許容する。また、リポジトリとして LDAP を推奨していることから、記載される Distinguished Name はその DIT に対応していなければならない。

(ホ) Validity (必須)

この領域には証明書の有効期間を記載する。この有効期間は様々なポリシによって異なるが、認証局は認証局証明書の有効期限を越えた証明書を発行してはならない。

(ヘ) Subject (必須)

この領域には空でない Distinguished Name を記載しなければならない。本プロファイルでは本領域を UTF8 でエンコードすることを推奨している。しかし、レガシーシステムへの相互運用性確保のために PrintableString の利用も許容する。また、リポジトリとして LDAP を推奨していることから、記載される Distinguished Name はその DIT に対応していなければならない。

(ト) subjectPublicKeyInfo (必須)

RSA アルゴリズム(鍵長は 2,048bit 以上)を利用する。

(チ) authorityKeyIdentifier (必須)

RFC3280 に示される最初の計算方法(160bit SHA-1 hash)を用いること。

(リ) subjectKeyIdentifier (必須)

RFC3280 に示される最初の計算方法(160bit SHA-1 hash)を用いること。

(ヌ) keyUsage (必須, Critical)

keyCertSign、cRLSign を含む。

(ル) Certificate Policies (必須, Critical または non-Critical)

policyID は必ず含むこと。

(ヲ) policyMappings (必須)

それぞれのポリシ OID を必ず含むこと。

(ワ) subjectAltName (任意)

証明書の持ち主に関する情報として電子メールアドレスや漢字やハングル文字などを含めたい場合には本領域を利用する。

(ｶ) issuerAltName (任意)

証明書の発行者情報として電子メールアドレスや漢字やハングル文字などを含めたい場合には本領域を利用する。

(ﾖ) basicConstraints (必須, Critical)

本領域は必ず TRUE とする。

(ﾀ) nameConstraints (任意, Critical)

本領域を利用する場合には、それぞれのドメインで利用を認める名前空間を記載する。

(ﾚ) policyConstraints (任意, Critical)

この領域は証明書検証におけるポリシ検証を制限するために利用する。

(ｿ) cRLDistributionPoints (必須)

distributionPoint.fullName を DN または URI で必ず含むこと。

(viii) 中間認証局証明書プロファイルの事前確認事項

(ｲ) Version (必須)

いくつかの拡張領域を利用しているので、バージョンは 3 とする。

(ﾛ) Serial Number (必須)

20 オクテット以下の一意な正整数

(ﾊ) Signature (必須)

署名アルゴリズムは sha1WithRSAEncryption を利用する。

(ﾆ) Issuer (必須)

この領域には空でない Distinguished Name を記載しなければならない。本プロファイルでは本領域を UTF8 でエンコードすることを推奨している。しかし、レガシーシステムへの相互運用性確保のために PrintableString の利用も許容する。また、リポジトリとして LDAP を推奨していることから、記載される Distinguished Name はその DIT に対応していなければならない。

(ﾎ) Validity (必須)

この領域には証明書の有効期間を記載する。この有効期間は様々なポリ

シによって異なるが、認証局は認証局証明書の有効期限を越えた証明書を発行してはならない。

(ﾍ) Subject (必須)

この領域には空でない Distinguished Name を記載しなければならない。本プロファイルでは本領域を UTF8 でエンコードすることを推奨している。しかし、レガシーシステムへの相互運用性確保のために PrintableString の利用も許容する。また、リポジトリとして LDAP を推奨していることから、記載される Distinguished Name はその DIT に対応していなければならない。ルート認証局証明書の本領域は Issuer と同じでなければならない。

(ﾄ) subjectPublicKeyInfo (必須)

RSA アルゴリズム(鍵長は 2,048bit 以上)を利用する。

(ﾁ) authorityKeyIdentifier (必須)

RFC3280 に示される最初の計算方法(160bit SHA-1 hash)を用いること。

(ﾘ) subjectKeyIdentifier (必須)

RFC3280 に示される最初の計算方法(160bit SHA-1 hash)を用いること。

(ﾇ) keyUsage (必須, Critical)

keyCertSign、cRLSign を含む。

(ﾙ) certificatePolicies (必須, Critical または non-Critical)

policyID は必ず含むこと。

(ﾌ) subjectAltName (任意)

証明書の持ち主に関する情報として電子メールアドレスや漢字やハングル文字などを含めたい場合には本領域を利用する。

(ﾜ) issuerAltName (任意)

証明書の発行者情報として電子メールアドレスや漢字やハングル文字などを含めたい場合には本領域を利用する。

(ｶ) basicConstraints (必須, Critical)

本領域は必ず TRUE とする。

(ヨ) cRLDistributionPoints (必須)

distributionPoint.fullName を DN または URI で必ず含むこと。

(ix) EE 証明書プロファイルの事前確認事項

(イ) Version (必須)

いくつかの拡張領域を利用しているので、バージョンは 3 とする。

(ロ) Serial Number (必須)

20 オクテット以下の一意な正整数とする。

(ハ) Signature (必須)

署名アルゴリズムは sha1WithRSAEncryption を利用する。

(ニ) Issuer (必須)

この領域には空でない Distinguished Name を記載しなければならない。本プロファイルでは本領域を UTF8 でエンコードすることを推奨している。しかし、レガシーシステムへの相互運用性確保のために PrintableString の利用も許容する。また、リポジトリとして LDAP を推奨していることから、記載される Distinguished Name はその DIT に対応していなければならない。

(ホ) Validity (必須)

この領域には証明書の有効期間を記載する。この有効期間は様々なポリシによって異なるが、認証局は認証局証明書の有効期限を越えた証明書を発行してはならない。

(ヘ) Subject (必須)

この領域には空でない Distinguished Name を記載しなければならない。本プロファイルでは本領域を UTF8 でエンコードすることを推奨している。しかし、レガシーシステムへの相互運用性確保のために PrintableString の利用も許容する。また、リポジトリとして LDAP を推奨していることから、記載される Distinguished Name はその DIT に対応していなければならない。ルート認証局証明書の本領域は Issuer と同じでなければならない。

(ト) subjectPublicKeyInfo (必須)

RSA アルゴリズム(鍵長は 1,024bit 以上)を利用する。

(ﾁ) authorityKeyIdentifier (必須)
RFC3280 に示される最初の計算方法(160bit SHA-1 hash)を用いること。

(ﾘ) subjectKeyIdentifier (必須)
RFC3280 に示される最初の計算方法(160bit SHA-1 hash)を用いること。

(ﾇ) keyUsage (必須, Critical)
利用用途に合わせた適切な値を記載すること。但し、keyCertSign と cRLSign を含めてはならない。

(ﾙ) certificatePolicies (必須, Critical または non-Critical)
policyID は必ず含むこと。

(ﾌ) subjectAltName (任意, Non-Critical)
証明書の持ち主に関する情報として電子メールアドレスや漢字やハングル文字などを含めたい場合には本領域を利用する。

(ﾜ) issuerAltName (任意, Non-Critical)
証明書の発行者情報として電子メールアドレスや漢字やハングル文字などを含めたい場合には本領域を利用する。

(ｶ) basicConstraints (任意, Critical)
EE 証明書には本領域を記載しないことを推奨する。

(ﾖ) cRLDistributionPoints (必須, Non-Critical)
distributionPoint.fullName を DN または URI で必ず含むこと。

(x) 証明書失効リストプロファイルの事前確認事項
(ｲ) version (必須)
バージョンは 2 とする。

(ﾛ) signature (必須)
署名アルゴリズムは sha1WithRSAEncryption を利用する。

(ﾊ) issuer (必須)
この領域には空でない Distinguished Name を記載しなければならない。本プロファイルでは本領域を UTF8 でエンコードすることを推奨している。しかし、レガシーシステムへの相互運用性確保のために PrintableString

の利用も許容する。また、リポジトリとして LDAP を推奨していることから、記載される Distinguished Name はその DIT に対応していなければならない。

(ニ) thisUpdate (必須)
UTCTIME 形式で記載すること。

(ホ) nextUpdate (必須)
UTCTIME 形式で記載すること。

(ヘ) revokedCertificates (必須)
(実験の進行に従って、)失効した証明書についての情報を記載すること。

(ト) ReasonCode (必須)
認証局は意味のある理由を本領域に記載することを推奨する。

(チ) invalidityDate (任意)
UTCTIME 形式で記載すること。

(リ) authorityKeyIdentifier (必須)
RFC3280 に示される最初の計算方法(160bit SHA-1 hash)を用いること。

(ヌ) issuerAltName (任意)
証明書失効リストの発行者情報として電子メールアドレスや漢字やハングル文字などを含めたい場合には本領域を利用する。

(ル) cRLNumber (必須, Non-Critical)
20 オクテット以下の一意な正整数

(ヲ) deltaCRLIndicator (任意, Critical)
デルタ CRL を利用する場合には本領域を利用する。

(ワ) issuingDistributionPoint
本領域は証明書失効リストの配布方法に従ってプロファイルを決定する。
(カ) freshestCRL (任意, Non-Critical)
デルタ CRL を利用する場合には本領域を利用する。

(xi) DIT に関する事前確認事項

証明書に記載されている DN が DIT と対応していること。

(xii) Schema に関する事前確認事項

(ｲ) CA

pkiCA または certificationAuthority を利用すること。

(ﾛ) EE

pkiUser または inetOrgPerson を利用すること

(ﾊ) cRLDP

もし CA エントリ以外で証明書失効リストを配布する場合には cRLDistributionPoint を利用すること。

(ﾆ) Referral

Referral を利用すること。

(b) 相互接続テスト手順

本標準は以下の内容を含む。

- 実験環境(付録 1-1 4.3 節)
- CC モデルにおける認証局間相互接続テスト手順(付録 1-1 4.4 節)
- CR モデルにおける認証局間相互接続テスト手順(付録 1-1 4.5 節)

以下に実験環境についての概要とテスト手順を示す。

(i) テスト環境

テストに必要な PKI ドメイン内のコンポーネントとして、認証局 (CA)、検証局 (VA)、ディレクトリサーバ及びアプリケーションサーバを定義する。

テストを実施する PKI ドメインはポート番号 389 の LDAPv3 をサポートするリポジトリを提供するべきである。

(ii) CC モデルにおける認証局間相互接続テスト手順

(ｲ) Step 1: 証明書発行 - 準備作業(Certificate issuance - Before trying each test flow)

利用者の認証局は以下の証明書を自 PKI ドメイン内で発行する:

- 自己署名証明書
- エンドエンティティ証明書
- 検証局証明書

(ﾛ) Step 2: 証明書発行 - 相互認証(Certificate issuance - Cross certification)

利用者の認証局とパートナー認証局は互いに相互認証証明書を発行しあう。CC 関係構築のために以下の手順を実施する:

- パートナー認証局へ相互認証証明書の発行を要求する
- 相互認証証明書を発行する (issuedByThisCA)
- 相互認証証明書ペアを作成する
- 相互認証証明書ペアをリポジトリに格納する

(ﾊ) Step 3: 証明書失効と検証(Certificate revocation and validation)

証明書の失効について検証するフェーズとして以下の手順を行う実施する:

- エンドエンティティ証明書を失効する
- 証明書失効リスト(CRL)をリポジトリに格納する
- エンドエンティティ証明書を検証する
- 相互認証証明書を失効する
- 証明書失効リスト(ARL)をリポジトリに格納する
- 相互認証証明書ペアをリポジトリから削除する
- エンドエンティティ証明書を検証する

(ﾆ) Step 4: 証明書更新(Certificate update)

"証明書更新"は既存の証明書に記載された値 (鍵情報を除く)を変更するために、新しい証明書を発行するオペレーションである。新しい証明書を発行する前に、既存の証明書を失効するか否かは任意である。証明書更新のテストは以下の手順を実施する:

- 相互認証証明書を発行する (issuedByThisCA)
- 相互認証証明書ペアを作成する
- 相互認証証明書ペアをリポジトリに格納する
- エンドエンティティ証明書を検証する

(ﾎ) Step 5: 証明書再発行(Certificate renewal)

“証明書再発行”は既存の証明書の有効期間を延長するために、新しい証明書を発行するオペレーションである。証明書再発行のテストは以下の手順を実施する:

- 相互認証証明書を発行する (issuedByThisCA)
- 相互認証証明書ペアを作成する
- 相互認証証明書ペアをリポジトリに格納する
- エンドエンティティ証明書を検証する

(iii) CR モデルにおける認証局間相互接続テスト手順

(ｲ) Step 1: 証明書発行 - 準備作業(Certificate issuance - Before trying each test flow)

利用者の認証局は以下の証明書を自 PKI ドメイン内で発行する:

- 自己署名証明書
- エンドエンティティ証明書
- 検証局証明書

(ﾛ) Step 2: 証明書発行 - 相互承認(Certificate issuance - Cross recognition)

CR(相互承認)関係を構築するために､相互承認相手の PKI ドメインのトラストアンカ(例えばルート認証局)の証明書を入手し、利用者のトラストポイントとして登録しなければならない。相互承認相手も同様である。

(ﾊ) Step 3: 証明書失効と検証(Certificate revocation and validation)

証明書の失効について検証するフェーズとして以下の手順を行う実施する:

- エンドエンティティ証明書を失効する
- 証明書失効リスト(CRL)をリポジトリに格納する
- エンドエンティティ証明書を検証する

5.3 証明書検証に関する標準

認証局の設計担当者及びソフトウェア開発者が、アプリケーションにおける証明書検証方法の設計作業において必要となる検証要件及び評価作業におけるテスト項目を検討する際に使用するパス検証に関する標準を定める。前年度作成した標準に対しテスト検証テストパターンの追加を行うと共に、パス検証テストツールを整備することにより実験を効率的に行う方法について追加する。

5.3.1 パス検証テストガイドライン

(1) パス検証テストガイドラインの内容

パス検証テストガイドラインはパス検証テストパターン、テスト項目、パス検証テストツールおよびパス検証テスト項目選択ワークシートの説明により構成されている。詳細については「付録 2-1 パス検証テストガイドライン」に示す。

(a) パス検証テストパターン

本標準では、ポリシの異なる複数のドメインが相互接続する際に確認すべきテスト要件とテスト項目を定義した。まず想定するテストモデルを定義し、次に各テストモデルに必要なテスト要件と、それに対応するテスト項目を定義し

ている。(付録 2-1 2 章)

(i) テストフレームワーク

(ｲ) テストの基本設計

対象とするトラストモデルやテスト項目の概要などの基本設計について述べている。(付録 2-1 2.1.1 節)

(ﾛ) テストフレームワークの前提条件

テスト項目が対象とするトラストモデルのトポロジに関する前提条件や署名鍵、扱う拡張領域などの前提条件について述べている。(付録 2-1 2.1.2 節)

(ﾊ) テスト環境

実験者が準備すべきトラストモデルに基づく認証局の構成、また署名検証者側が準備する事項について述べている。(付録 2-1 2.1.3 節)

(ﾆ) テスト項目の表記規約

テスト項目の表記法について述べている。(付録 2-1 2.1.4 節)

(ﾎ) パス検証ガイドラインの使用法

パス検証ガイドラインの使用法について、実験対象のトラストモデルの決定と使用プロファイルの定義を必要とすることを具体的に述べている。(付録 2-1 2.1.5 節)

(ii) テストモデルおよびテストの要件

PKI ドメインの分類やパス検証の要件について述べている。(付録 2-1 2.2 節)

(ｲ) 様々な PKI ドメインの分類

本ガイドラインのテスト項目範疇を超えた PKI ドメインにおけるトラストモデルを網羅的に分析、紹介している。(付録 2-1 2.2.1 節)

(ﾛ) パス検証の要件

全てのパス検証テスト項目毎に、X.509 4th Edition、RFC3280 や IWG プロファイル[14]などの標準のうち、どの要件を対象としたテストであるかを述べている。(付録 2-1 2.2.2 節)

(iii) テストケースの前提条件

パス検証テストガイドラインで用いられるエンティティ構成、トラストモデル毎のプロファイル定義と、認証パス検証の初期値について述べている。(付録 2-1 2.3 節)

[14]日本、韓国、シンガポール、チャイニーズ台北、香港チャイナ及びタイで合意した認証局間相互接続のための証明書/CRL プロファイル。http://www.japanpkiforum.jp/shiryou/IWG_2002/Appendix.pdfの Appendix 1. IWG Recommended Profiles に相当する。本報告書中では以降 IWG プロファイルと表記する。

(ｲ) ベースモデルの構成エンティティ、使用プロファイル、初期値
(ﾛ) 階層モデル(SH)の構成エンティティ、使用プロファイル、初期値
(ﾊ) 相互認証モデル(CC)の構成エンティティ、使用プロファイル、初期値
(ﾆ) 相互承認モデル(CR)の構成エンティティ、使用プロファイル、初期値

(b) テスト項目

トラストモデル毎にテスト項目をまとめた表を示している。(付録 2-1 3 章)
パス検証ガイドラインで含まれるテストケースを以下に示す。

- 正常系テストケース
- 発行者名と主体者名との DN マッチングテストケース
- 有効期限をチェックするテストケース
- 署名をチェックするテストケース
- 証明書の失効に関するテストケース
- 発行者と主体者の鍵識別子の一致に関するテストケース
- 基本制約拡張に関するテストケース
- 鍵使用目的拡張に関するテストケース
- 証明書ポリシ拡張に関するテストケース
- ポリシ制約拡張に関するテストケース
- ポリシマッピング拡張に関するテストケース
- 名前制約に関するテストケース
- cRLDistributionPoints, issuingDistributionPoint 拡張のテストケース
- UTF8 CJK 文字セットに関するテストケース
- 未知のプライベート拡張に関するテストケース
- CRL エントリ拡張に関するテストケース

(i) ベースモデルのテスト項目（付録 2-1 3. 1 節）
(ii) 階層モデルのテスト項目（付録 2-1 3.2 節）
(iii) 相互認証モデルのテスト項目（付録 2-1 3.3 節）
(iv) 相互承認モデルのテスト項目（付録 2-1 3.4 節）

(c) IWG テストツール

IWG テストツールはパス検証テストガイドラインの実証実験に用いることが可能な統合テスト環境であり、1 台の Linux マシンで複数の仮想認証局、仮想リポジトリを提供が可能であり、テストに必要なデータを一括生成することが可能である。(付録 2-1 4 章)

(i) IWG テストツールの概要

IWG テストツールの概要および構成について述べている。(付録 2-1 4.1 節)

(ii) テスト項目の設計

IWG テストツールを用いたテスト項目の設計手順について述べている。(付録 2-1 4.2 節)

(iii) ツールを用いたテストの実施

IWG テストツールを用いたパス検証テストの実施手順について述べている。(付録 2-1 4.3 節)

(iv) 環境構築

IWG テストツールの環境構築方法および、必要なハードウェア、ソフトウェア要件について述べている。(付録 2-1 4.4 節)

(d) テスト項目選択ワークシート

テスト項目選択ワークシートとは、パス検証テストガイドラインで述べられているテスト項目において、実験国間で必要なテスト項目を抽出するのを支援するウェブブラウザで参照可能なオンラインコンテンツである。キーワードに基づき関係の無いテスト項目を非表示状態にしたり、関心のテスト項目のみを表示状態にすることが可能である。(付録 2-1 5 章)

(i) テスト項目の表示、非表示の切り替え

(ii) テスト項目の切り替えに用いる用いるキーワードについて

5.4　アプリケーションインターフェースに関する標準

相互接続基盤の整備に際し、アプリケーションに関する標準としてアプリケーションインターフェース仕様を定める。

5.4.1 アプリケーションインターフェース仕様

(1) アプリケーションインターフェース仕様の内容

本標準は、平成 14 年度の本事業で作成した署名用トークンインターフェース仕様について、改善及び必要となる機能を拡張したものである。平成 14 年度に策定した標準と今年度に策定した標準の比較を「表 5.1　平成 14 年度策定した標準との比較」に示す。

表 5.1　平成 14 年度策定した標準との比較

	機能	平成 14 年度	平成 15 年度
1	署名付与・検証	PKCS#1 暗号化形式の署名データフォーマット	SHA-1 形式のハッシュ値を作成し、PKCS#1 形式で暗号化する署名データフォーマット
2	複数鍵／スロット	－	複数鍵／スロットが存在する環境へ対応
3	鍵保護	－	暗号化で使用する共通鍵の安全な送信方法に対応
4	暗号化・復号	－	データの暗号化・復号に対応

5	初期化ファイルの格納場所	c:¥windows¥system32¥	c:¥program files¥iwg¥
6	初期化ファイルの内容	PKCS#11 ライブラリの格納場所を記述。	複数のスロット、各スロットが使用する PKCS#11 ライブラリ名、PKCS#11 ライブラリの格納場所を記述。

本標準が想定するアプリケーションモデルにおいて、「表 5.1　平成 14 年度策定した標準との比較」の全ての項目を定義する。本標準の詳細は「付録 3　アプリケーションインターフェース仕様」に示す。

アプリケーションモデルは平成 14 年度と同様であり、その概要を「図 5.2　アプリケーションモデルの概要」に示す。

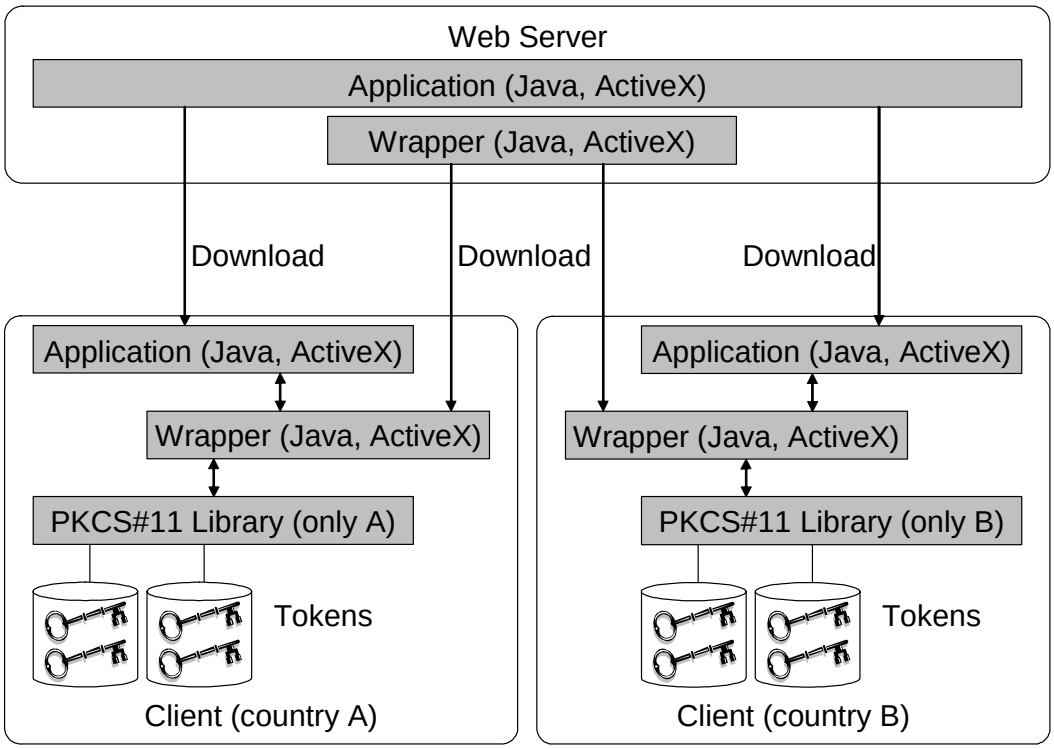


図 5.2　アプリケーションモデルの概要

本環境では、Web Server 上のアプリケーションとセットで Wrapper と呼ばれるモジュール（Java Applet or ActiveX）が置かれ、クライアントからの要求に応じてクライアントへダウンロードされる。

クライアントにおいては、PKCS#11 ライブラリが予めインストールされており、その PKCS#11 ライブラリが管理するトークンに秘密鍵と証明書が格納されている。ここで、トークン及びトークンに格納される秘密鍵と証明書は複数存在

することが可能である。クライアントは Web ブラウザを使用して Web Server 上のアプリケーションへアクセスし、必要に応じて Wrapper をダウンロードし、Wrapper を介してアプリケーションと PKCS#11 ライブラリが連携する。

機能については次のようになる。まず、Web Server がアプリケーションサービスを提供する。クライアントにおいては、Web Server へアクセスしている状態で、PKCS#11 ライブラリの機能を呼び出す際に Web Server から Wrapper をダウンロードし、PKCS#11 ライブラリの処理結果をアプリケーションへ渡す。Wrapper とは、各国／地域の機能的な差異及び PKCS#11 ライブラリとアプリケーションの開発言語の差異を吸収するための機能として導入している。

PKCS#11 ライブラリとトークンは各国／地域が利用可能なものを使用するが、Wrapper はアプリケーションサービス提供者が共通なモジュールとして提供する。

(a) 使用する PKCS#11 関数

今年度の仕様で使用する関数の一覧を「表 5.2　使用する関数一覧」に示す。

表 5.2　使用する関数一覧

関数名	機能説明
C_Initialize	PKCS#11 ライブラリの環境を初期化する。
C_Finalize	PKCS#11 ライブラリの終了処理として、PKCS#11 が使用した全てのリソースを開放する。
C_GetFunctionList	PKCS#11 ライブラリの関数のエントリアドレスのリストを取得する。
C_GetSlotList	スロットを識別する ID のリスト又はスロットの数を取得する。
C_GetSlotInfo	スロットに関する情報を取得する。
C_GetTokenInfo	指定したスロットに対応するトークンの情報を取得する。
C_GetMechanismList	トークンでサポートされているメカニズムタイプのリスト、又はメカニズムタイプの数を獲得する。
C_OpenSession	プロセスとトークンの間でセションを開設する。
C_CloseSession	トークンとプロセス間のセションを閉じる。
C_Login	トークンへログインする。
C_Logout	トークンから利用者をログアウトする。
C_FindObjectsInit	テンプレートで指定した属性と一致するオブジェクトを検索するための環境を初期化する。
C_FindObjects	C_FindObjectsInit 関数で設定した検索条件と一致するトークンオブジェクトおよび、セションオブジェクトを検索する。
C_FindObjectsFinal	オブジェクトの検索を終了する。
C_GetAttributeValue	オブジェクトの属性値を獲得する。

C_CreateObject	オブジェクトを新規に登録する。
C_DestroyObject	オブジェクトを破棄する。
C_GenerateKey	共通鍵を生成する。
C_SignInit	署名操作の初期化を行う。
C_Sign	シングルパートデータの署名処理を行う。
C_VerifyInit	署名検証操作の初期化を行う。
C_Verify	シングルパートデータの署名検証操作を行う。
C_EncryptInit	暗号化処理の初期化を行う。
C_Encrypt	シングルパートデータを暗号化する。
C_DecryptInit	復号処理の初期化を行う。
C_Decrypt	シングルパートデータを復号する。
C_WrapKey	鍵をラップする。
C_UnwrapKey	ラップされた鍵をアンラップする。

(b) 使用する関数で必要となるパラメータ

(i) PKCS#11 ライブラリのファイル名

PKCS#11 ライブラリのファイル名は、Wrapper がクライアントに格納されている PKCS11.ini ファイルをロードすることで解決を図る。

(ii) メカニズム

使用するメカニズムを「表 5.3　メカニズムリスト」に示す。

表 5.3　メカニズムリスト

メカニズム	説明
CKM_RSA_PKCS	署名値を作成する際、ハッシュ値を暗号化するために使用する。
CKM_DES3_KEY_GEN	暗号化及び鍵のラップ処理を行う際、共通鍵を作成するために使用する。
CKM_DES3_CBC	暗号化及び鍵のラップ処理で使用する。
CKM_SHA_1	署名値を作成する際、ハッシュ値を作成するために使用する。

(iii) アルゴリズム

使用するアルゴリズムを「表 5.4　アルゴリズムリスト」に示す。

表 5.4　アルゴリズムリスト

アルゴリズム	説明
CKA_SIGN	署名付与に関する属性。
CKA_VERIFY	署名検証に関する属性。
CKA_ENCRYPT	暗号化に関する属性。
CKA_DECRYPT	復号に関する属性。
CKA_WRAP	鍵のラップに関する属性
CKA_UNWRAP	鍵のアンラップに関する属性

(iv) 暗号アルゴリズム及び鍵長

秘密鍵及び公開鍵は RSA アルゴリズムで鍵長 1024 ビットとする。

(c) その他

(i) pkcs11.ini ファイルのフォーマット

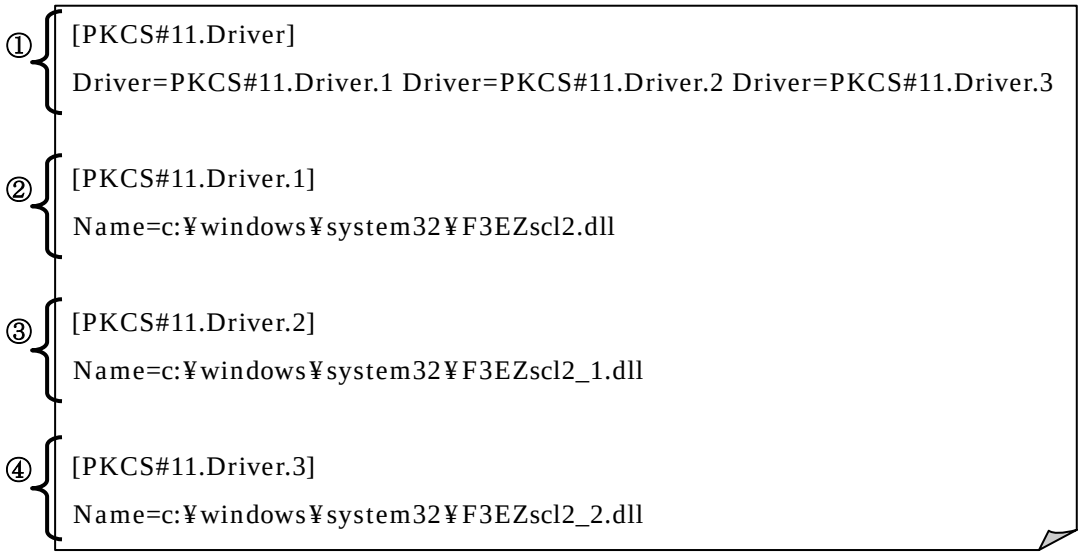


図 5.3　pkcs11.ini ファイルのフォーマット

pkcs11.ini ファイルでは、クライアントに複数存在するスロットと各スロットがどの PKCS#11 ライブラリを使用するのかをアプリケーションに認識させる役割を果たす。

「図 5.3　pkcs11.ini ファイルのフォーマット」の①の項では、クライアントに存在するスロットの数をリストとして記載する。②から④の項では、各スロットが使用する PKCS#11 ライブラリの格納場所とファイル名を記載する。

この pkcs11.ini ファイルをアプリケーションが読み込むことで各国／地域クライアント環境の差異を吸収する。

(ii) pkcs11.ini ファイルの格納場所

pkcs11.ini ファイルは、C:¥Program Files¥iwg¥フォルダに格納する。

6 開発作業

以下に示す 2 つの機能を実現するソフトウェアを開発する。

- アプリケーションインターフェース確認機能
- パス検証テストツール

6.1 アプリケーションインターフェース確認機能

6.1.1 概要説明

トークンに格納された公開鍵証明書及び秘密鍵を利用可能とするための機能等を具備するソフトウェアとして、アプリケーションインターフェース確認機能を開発する。

なお、今回の開発では、平成 14 年度に開発した署名用トークン確認機能、共通インターフェース機能を活用する。

6.1.2 全体構成

本ソフトウェアは、Java アプレットとして提供される。

利用者は、クライアントに JavaVM（Java Runtime Environment）をインストールし、WWW ブラウザ上から本機能を利用する。

アプリケーションの全体構成を「図 6.1 アプリケーションの全体構成」に示す。

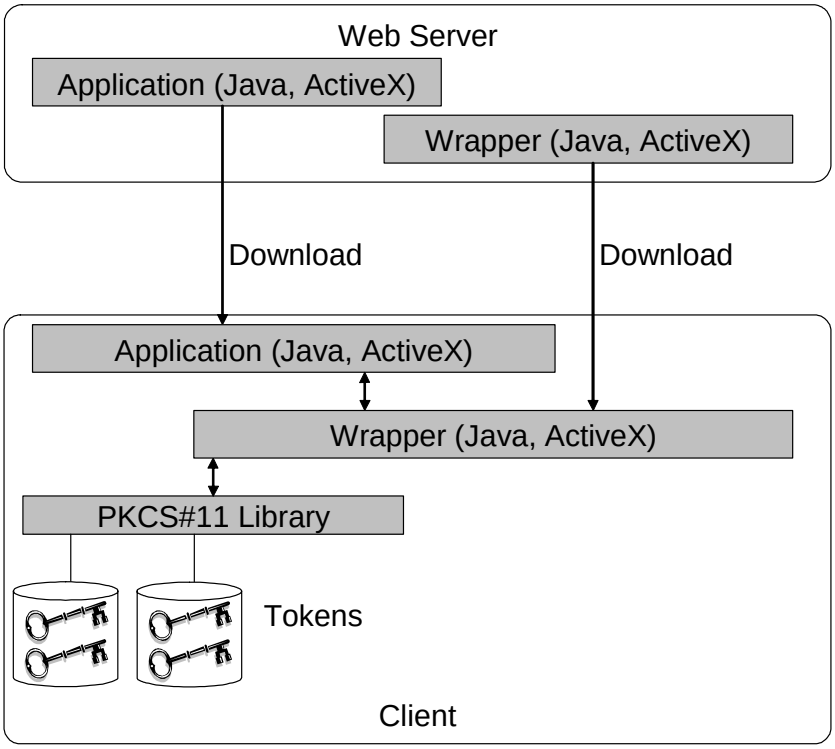


図6.1 アプリケーションの全体構成

また、クライアントについて、日本におけるクライアントの構成を「図 6.2 日本

のクライアント構成」に示す。

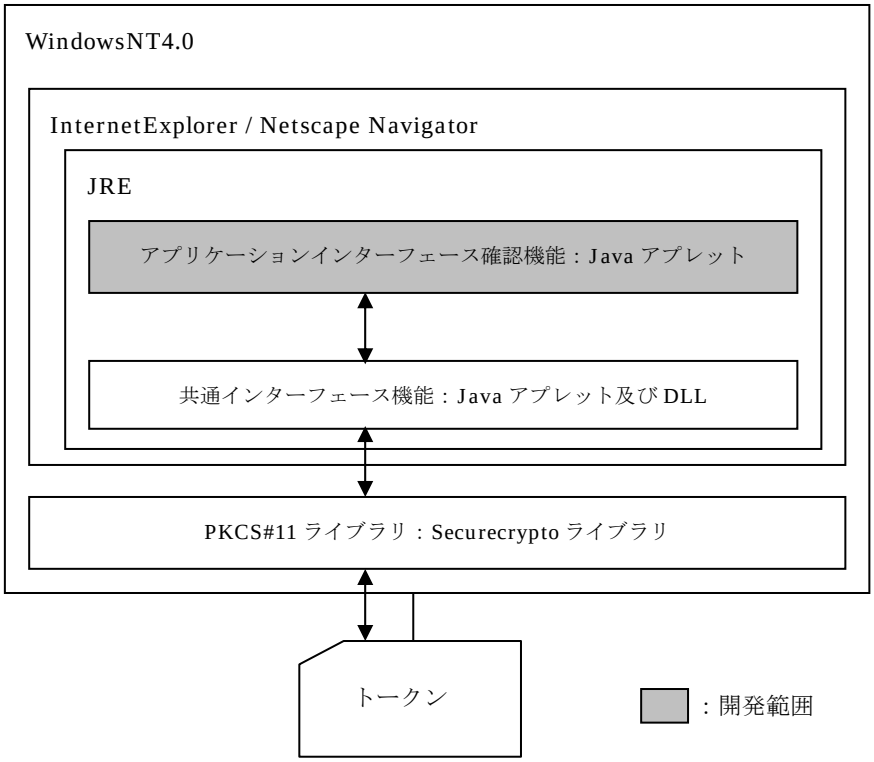


図6.2 日本のクライアント構成

「図 6.1 アプリケーションの全体構成」との対比では、アプリケーションインターフェース確認機能が Application に対応し、共通インターフェース機能が Wrapper に対応する。

クライアント上の WindowsNT4.0 上で PKCS#11 ライブラリが動作し、その管理下にトークンがある。

アプリケーションとして、アプリケーションインターフェース確認機能が Java アプレットとして動作し、アプリケーションインターフェース確認機能と PKCS#11 ライブラリの間で両者の連携を行う共通インターフェース機能がある。

アプリケーションインターフェース確認機能は、鍵管理、暗号化・復号を行うアプリケーションとして Java で開発される。

また、共通インターフェース機能は、平成 14 年度に開発したアプリケーションを活用する。

なお、PKCS#11 ライブラリとしては、富士通株式会社製の PKCS#11 ライブラリである Securecrypto ライブラリ V1.0 を用いることを前提とする。

6.2　パス検証テストツール

6.2.1 概要説明

パス検証テストガイドラインに則ったテストを支援するソフトウェアとして、以下のシステムを開発する。

(1) パス検証テストデータローダー

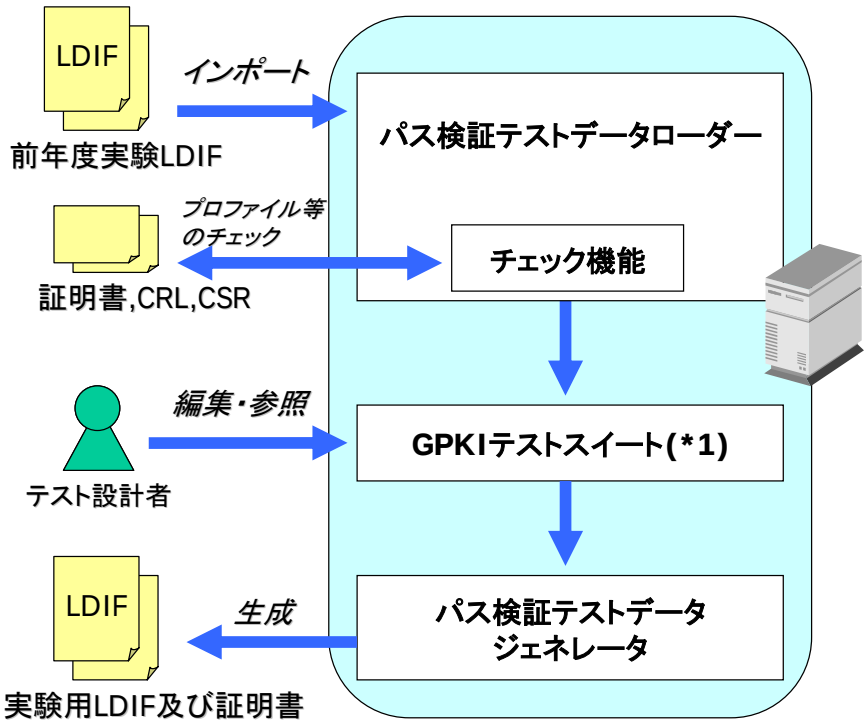
リポジトリの汎用データフォーマットである LDIF ファイルを入力としてテストケースを管理するデータベースへ登録するコマンドラインプログラムである。これにより、前年度実験で利用したデータの再利用や改変が容易可能となる。なお、証明書、CRL および証明書発行要求のプロファイル準拠性のチェック機能を含む。

(2) パス検証テストデータジェネレータ

データベースで管理されているテスト情報に基づき、汎用リポジトリデータフォーマットである LDIF を生成するためのコマンドラインプログラムである。これによりテストに必要な鍵ペア、証明書、証明書失効リストなどの情報を含んだ LDIF ファイルを一括して自動生成することができる。

6.2.2 全体構成

本ソフトウェアの全体構成を「図 6.3　パス検証テストツールの全体構成」に示す。



(*1)情報処理振興事業協会 平成14年度「電子政府情報セキュリティ技術開発事業 電子政府情報セキュリティ相互運用支援技術の開発(GPKI相互運用フレームワークの開発)」の成果物

図 6.3　パス検証テストツールの全体構成

6.2.3 他システムとの関連

情報処理振興事業協会 平成 14 年度「電子政府情報セキュリティ技術開発事業 - 電子政府情報セキュリティ相互運用支援技術の開発(GPKI 相互運用フレームワークの開発)」の成果物の一部である「PKI 相互運用テストスイート」(以降「GPKI テストスイート」と呼ぶ)に含まれているテストデータ管理データベース、証明書生成管理や LDAP リポジトリ管理などの機能を利用し、動作する。

7 実験環境

本実験環境構築作業では、日本国内に実際に運用している環境を想定した模擬環境を構築するため、以下の実験環境構築作業を実施した。

- ネットワーク環境構築作業
- サーバ環境構築作業
- クライアント環境構築作業

実証実験環境構成は、「表 7.1　実証実験環境構成図」に示す。

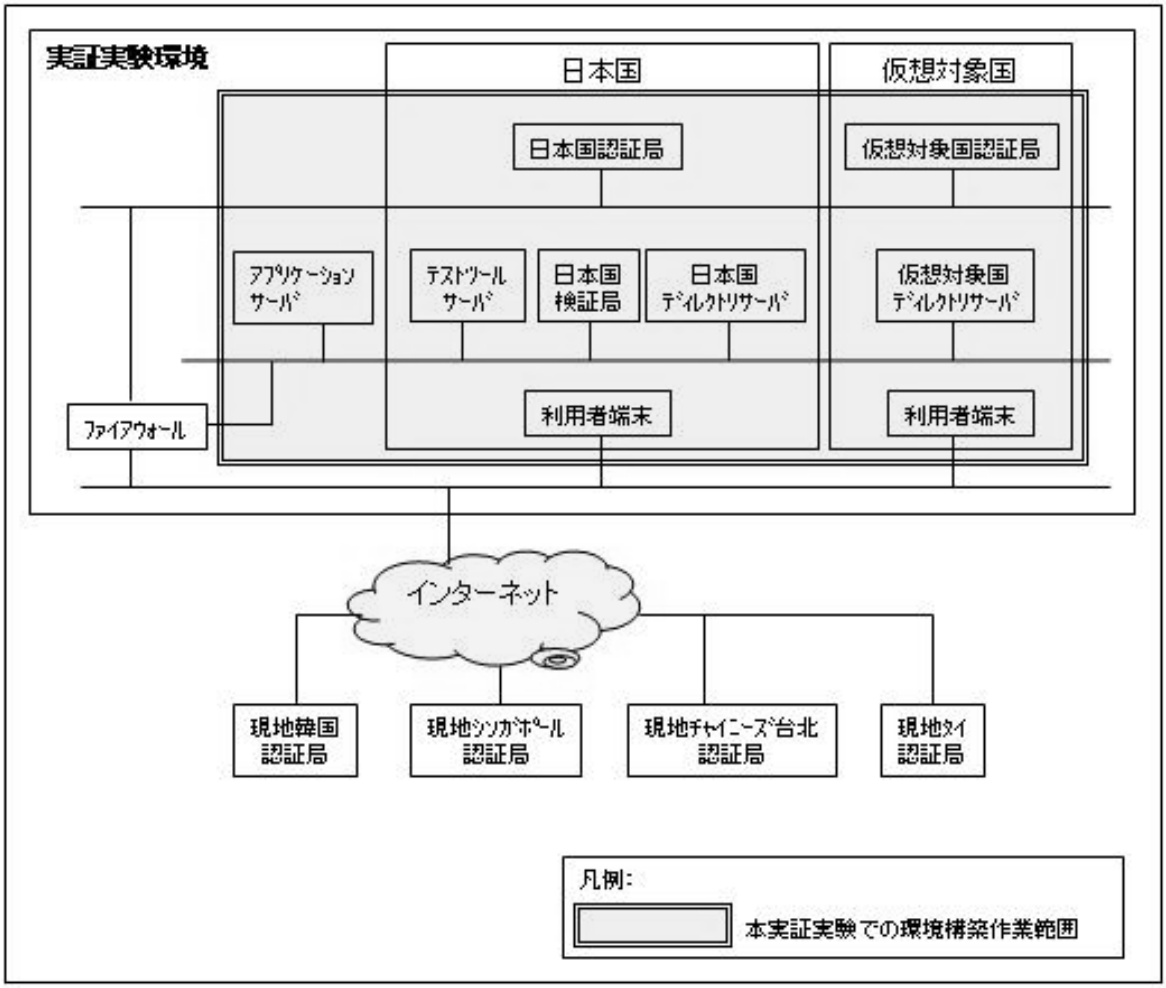


図 7.1　実証実験環境構成図

(1) ネットワーク環境構築作業

日本国ディレクトリサーバ、仮想対象国ディレクトリサーバ、日本国認証局、仮想対象国認証局、日本検証局、テストツールサーバ、アプリケーションサーバ

及び利用者端末間を、ファイアウォールを介して適切なアクセス制御のもとで接続した。

(2) サーバ環境構築作業

実証実験環境に以下のサーバを設置し、ネットワークに接続する。

- アプリケーションサーバ
- 日本検証局
- テストツールサーバ
- 日本国ディレクトリサーバ
- 日本国認証局
- 仮想対象国ディレクトリサーバ
- 仮想対象国認証局

(3) クライアント環境構築作業

実証実験環境に WWW ブラウザが動作する利用者端末を設置した。

8 実験項目

作成した標準案の有効性を検証するため、「表 8.1　実験単位一覧」に示す実験単位で実証実験を実施する。

表 8.1　実験単位一覧

大項目	中項目	小項目
(1)現地タイ認証局との認証局間相互接続テストガイドラインの実証実験	(1-1)相互接続に係わる証明書発行	(1-1-1)自己署名証明書の発行
		(1-1-2)自己署名証明書のリポジトリへの格納
		(1-1-3)エンドエンティティ証明書の発行
		(1-1-4)エンドエンティティ証明書のリポジトリへの登録
		(1-1-5)証明書検証サーバ証明書の発行
		(1-1-6)証明書要求の発行
		(1-1-7)相互認証証明書の発行
		(1-1-8)相互認証証明書ペアの作成
		(1-1-9)相互認証証明書ペアのリポジトリへの格納
		(1-1-10)エンドエンティティ証明書の検証
	(1-2)相互接続に係わる証明書失効	(1-2-1)エンドエンティティ証明書の失効と CRL の発行
		(1-2-2)CRL のリポジトリへの格納
		(1-2-3)エンドエンティティ証明書のリポジトリからの削除
		(1-2-4)エンドエンティティ証明書の検証
		(1-2-5)相互認証証明書の失効と ARL の発行
		(1-2-6) ARL のリポジトリへの格納
		(1-2-7) 相互認証証明書ペアのリポジトリからの削除
		(1-2-8) エンドエンティティ証明書の検証
	(1-3)相互接続に係わる証明書更新	(1-3-1)相互認証証明書の更新
		(1-3-2)相互認証証明書ペアの作成

		(1-3-3)相互認証証明書ペアのリポジトリへの格納
		(1-3-4)エンドエンティティ証明書の検証
	(1-4)相互接続に係わる証明書再発行	(1-4-1)相互認証証明書の再発行
		(1-4-2)相互認証証明書ペアの作成
		(1-4-3)相互認証証明書ペアのリポジトリへの格納
		(1-4-4)エンドエンティティ証明書の検証
(2)仮想対象国認証局との認証局間相互接続テストガイドラインの実証実験	(2-1)CC モデルにおける認証局間相互接続テストガイドラインの検証	-
	(2-2)CR モデルにおける認証局間相互接続テストガイドラインの検証	-
(3)パス検証テストガイドラインの実証実験	(3-1)Interconnection Model–Strict Hierarchy	-
(4)アプリケーションインターフェース仕様の実証実験	(4-1)複数鍵／スロット処理検証	-
	(4-2)暗号化・鍵保護処理検証	-
	(4-3)署名付与・検証処理検証(正常系)	-
	(4-4)署名付与・検証処理検証(異常系)	-

9章以降で「表 8.1　実験単位一覧」に示した実験単位ごとに実証実験の詳細を記述する。

実験単位ごとの対象国／地域を「表 8.2　実証実験対象国／地域」に示す。

表 8.2　実証実験対象国／地域

実験単位大項目	対象国／地域
(1)現地タイ認証局との認証局間相互接続テストガイドラインの実証実験	日本
	タイ
(2)仮想対象国認証局との認証局間相互接続テストガイドラインの実証実験	日本
(3)パス検証テストガイドラインの実証実験	日本
	チャイニーズ台北
(4)アプリケーションインターフェース仕様の実証実験	日本
	韓国
	シンガポール
	チャイニーズ台北

9 現地タイ認証局との認証局間相互接続テストガイドラインの実証実験

9.1　相互認証に係わる証明書発行

(1) 目的

本実験単位は、相互接続に係わる証明書発行を通しての「認証局間相互接続テストガイドライン」の有効性の検証を目的とする。

利用者に、認証局における証明書発行作業のテスト、及び発行された証明書の検証テストを「認証局間相互接続テストガイドライン」に従って実施してもらい、「認証局間相互接続テストガイドライン」が、利用者にとって、テスト実施に必要な事前チェックの要件やテスト手順、及び実施結果の正確な確認方法についての必要十分な内容であることを確認する。さらに「認証局間相互接続テストガイドライン」に従ったテスト実施で期待した結果が得られることでテスト手順や期待値の妥当性を確認する。これらにより「認証局間相互接続テストガイドライン」の有効性の確認とする。

(2) 実験内容

現地タイ認証局との間で、相互接続モデルとして CC モデルを選択し、「認証局間相互接続テストガイドライン」の「Step 1: 証明書発行 - 準備作業(Certificate issuance - Before trying each test flow)」及び「Step 2: 証明書発行 - 相互認証(Certificate issuance - Cross certification)」の認証局に係わるテスト項目を実施する。

実験内容の概要を「図 9.1　現地タイ認証局との証明書発行テスト」に示す。

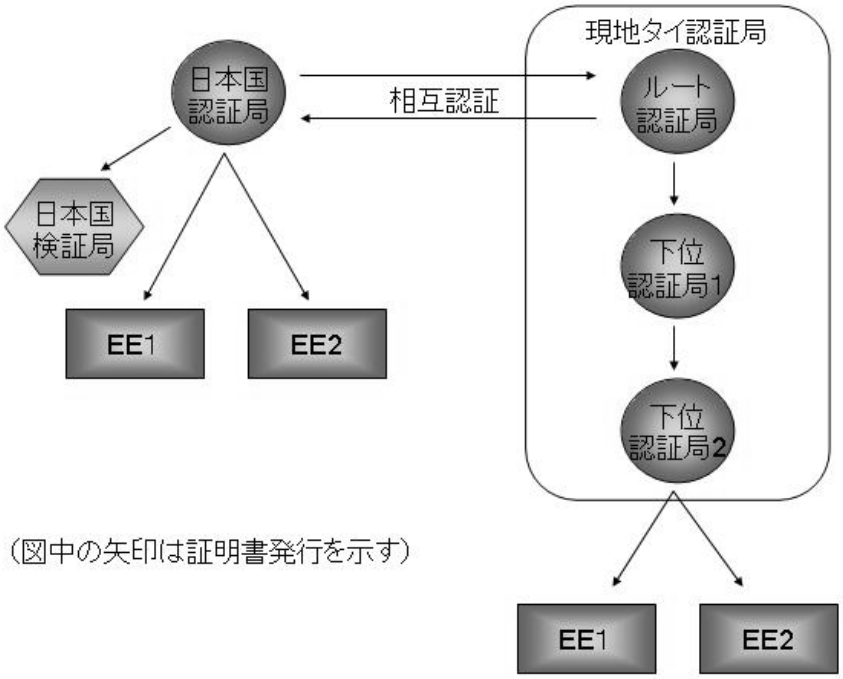


図 9.1　現地タイ認証局との証明書発行テスト

実施するテスト項目を「表 9.1　現地タイ認証局との証明書発行テスト項目一覧」に示す。表中のテスト ID の列の StepN-N(‘N’は数字を表す)という表記は「認証局間相互接続テストガイドライン」に記載されたテスト項目を識別する ID である。以下、本報告書中ではこのテスト ID で、「認証局間相互接続テストガイドライン」中のテスト項目を識別する。

表 9.1　現地タイ認証局との証明書発行テスト項目一覧

No.	実験単位小項目	ガイドライン内のテスト項目	
		テスト ID	内容
1	(1-1-1)自己署名証明書の発行	Step 1-1	Issue a CA's self-signed certificate
2	(1-1-2)自己署名証明書のリポジトリへの格納	Step 1-2	Store the self-signed certificate in repository.
3	(1-1-3)エンドエンティティ証明書の発行	Step 1-3	Issue EE certificates
4	(1-1-4)エンドエンティティ証明書のリポジトリへの登録	Step 1-4	[optional] Store EE certificates in repository.
5	(1-1-5)証明書検証サーバ証明書の発行	Step 1-5	[optional] Issue a VA certificate.
6	(1-1-6)証明書要求の発行	Step 2-1	Request Cross certification to the partner CA.
7	(1-1-7)相互認証証明書の発行	Step 2-2	Issue a cross certificate (issuedByThisCA)

8	(1-1-8)相互認証証明書ペアの作成	Step 2-3	Make a cross certificate pair.
9	(1-1-9)相互認証証明書ペアのリポジトリへの格納	Step 2-4	Store the cross certificate pair into the repository
10	(1-1-10)エンドエンティティ証明書の検証	Step 2-5	Verify the valid EE certificates.

(3) 実験方法

「表 9.1　現地タイ認証局との証明書発行テスト項目一覧」に示した項目について、「認証局間相互接続テストガイドライン」の定める手順に従ってテストを実施する。

日本国認証局は、認証局のオペレーションである Step 1-1 から Step2-4 までを実施し、現地タイ認証局との間で相互認証関係を構築する。

日本国検証局を用いて Step 2-5 を実施する。Step 2-5 では Step 1-3 で発行したエンドエンティティ証明書を検証する。実験にあたっては、利用者端末及び日本国検証局の設定が、「表 9.2　証明書発行テストにおける証明書検証モジュールへのインプット情報」を満たす設定となっていることを確認する。

表 9.2　証明書発行テストにおける証明書検証モジュールへのインプット情報

No.	インプット情報	設定
1	受け入れ可能なポリシ情報 (user-initial-policy-set)	設定しない
2	トラストアンカ情報 (trust anchor information)	日本国認証局の自己署名証明書を設定
3	ポリシマッピングを許可するか否かの情報 (initial-policy-mapping-inhibit)	ポリシマッピングを許可
4	証明書パス中にユーザが受け入れ可能なポリシが入っていることを要求するか否かの情報 (initial-explicit-policy)	要求しない
5	any-policy を許可するか否かの情報 (initial-any-policy-inhibit)	許可する

エンドエンティティ証明書検証の実験結果期待値を「表 9.3　Step2-5 の実験結果期待値」に示す。

表 9.3　Step2-5の実験結果期待値

No	テスト ID		期待値		備考
			結果	Return Code	
1	Step2-5	Step2-5-1	Valid	0	「Return Code=0」は証明書が有効であることを示す。

(4) 実験結果

「認証局間相互接続テストガイドライン」の定めるテスト項目の実施結果を「表 9.4　現地タイ認証局との証明書発行テスト実施結果」に示す。

表 9.4　現地タイ認証局との証明書発行テスト実施結果

テスト ID		実施結果	
		1 回目	2 回目
Step1-1	Step1-1-1	問題なし	－
	Step1-1-2	問題なし	－
	Step1-1-3	問題なし	－
Step1-2	Step 1-2-1	問題なし	－
	Step 1-2-2	問題なし	－
	Step 1-2-3	問題なし	－
	Step 1-2-4	問題なし	－
	Step1-2-5	問題なし	－
Step1-3	Step1-3-1	問題なし	－
	Step1-3-2	問題なし	－
	Step1-3-3	問題なし	－
Step1-4	Step1-4-1	問題なし	－
	Step1-4-2	問題なし	－
	Step1-4-3	問題なし	－
	Step1-4-4	問題なし	－
	Step1-4-5	問題なし	－
Step1-5	Step1-5-1	問題なし	－
	Step1-5-2	問題なし	－
	Step1-5-3	問題なし	－
Step2-1	Step2-1-1	問題なし	－
	Step 2-1-2	問題なし	－
	Step 2-1-3	問題なし	－
	Step 2-1-4	問題なし	－
	Step 2-1-5	問題なし	－
	Step 2-1-6	問題なし	－
Step2-2	Step 2-2-1	問題なし	－
	Step 2-2-2	問題なし	－
	Step 2-2-3	問題なし	－
	Step 2-2-4	問題なし	－
	Step 2-2-5	問題なし	－
	Step 2-2-6	問題なし	－
	Step 2-2-7	問題なし	－
	Step 2-2-8	問題なし	－
	Step 2-2-9	問題なし	－

	Step 2-2-10	問題なし	ー
Step2-3	Step 2-3-1	受け取った相互認証証明書のチェックで subjectKeyIdentifier の値が日本国認証局で生成したものと一致していない問題を検出。 現地タイ認証局が設定変更し、問題なしとなった。	問題なし
	Step 2-3-2	問題なし	問題なし
	Step 2-3-3	問題なし	問題なし
	Step 2-3-4	問題なし	問題なし
Step2-4	Step 2-4-1	問題なし	問題なし
	Step 2-4-2	問題なし	問題なし
	Step 2-4-3	問題なし	問題なし
	Step 2-4-4	問題なし	問題なし
	Step 2-4-5	問題なし	問題なし
	Step 2-4-6	問題なし	問題なし
	Step 2-5-1	タイの証明書を日本国検証局が処理できない問題が発生した。原因は、現地タイ認証局が発行した証明書の ASN.1 構造が間違っていたことである。現地タイ認証局が ASN.1 構造を修正して Step2-3-1 より再実施した。	問題なし

日本国検証局のログ情報を解析し、エンドエンティティ証明書検証結果を評価した。結果を「表 9.5　現地タイ認証局との証明書発行テストにおけるエンドエンティティ証明書検証結果」に示す。

表 9.5　現地タイ認証局との証明書発行テストにおけるエンドエンティティ証明書検証結果

No.	テスト ID	期待値		検証結果	
		結果	Return Code	結果	Return Code
1	Step 2-5-1	Valid	0	Valid	0

(5) 評価及び考察

本実験単位では、「表 9.4　現地タイ認証局との証明書発行テスト実施結果」に

示す通りすべてのテスト項目について期待した結果を得ることができた。

ただし、現地タイ認証局から、エンドエンティティ証明書の発行(テスト ID:Step1-3-2)及び相互認証証明書発行(テスト ID:Step2-2-5)にあたって、「認証局間相互接続テストガイドライン」で必須としているエンドエンティティ証明書の certificatePolicies と、相互認証証明書の policyMapping を設定しないで実験を進めたいとする提案があった。相互認証においてセキュリティ要件を確保するためには両フィールドは重要である旨を補足説明し、現地タイ認証局の理解を得て実験は両フィールドを設定しての実施となったが、これは「認証局間相互接続テストガイドライン」に両フィールドについてより詳細な解説が必要か否かの検討材料となった。タイ側は実験用のポリシ ID を取得するのが難しいとの理由から両フィールドを未使用としたいと考えたとのことで、本質的に両フィールドの使用に反対しているものではない。この経緯が明らかになったので、「認証局間相互接続テストガイドライン」が現状両フィールドを設定必須としていることで情報としては足りていると判断し、追記は行わないこととした。

また、相互認証証明書の受け取り(テスト ID:Step2-3-1)で subjectKeyIdentifier の値が現地タイ認証局と日本国認証局で異なる生成方法を採用していたため一致しないことが検出された。この問題は、過去の実証実験でも同一原因の問題が発生している。それが再度発生したことで、認証局間相互接続時に注意すべき点として「認証局間相互接続テストガイドライン」により詳細な説明を追加する必要があると分析し、テスト手順の相互認証証明書の内容チェックで keyID の値の一致を確認するよう明記するフィードバックを行った。この件に関しては「14.2.1(1)keyID の計算方法について」でより詳細な考察を行っているので参照されたい。

エンドエンティティ証明書の検証の項目(テスト ID: Step2-5-1)で ASN.1 エンコードの問題が検出された。この件に関しても「14.2.1(3)証明書の ASN.1 構造及びエンコーディング方法に関する考察」でより詳細な考察を行っているので参照されたい。

これらの結果から、利用者が証明書発行に関するテストを行う際に「認証局間相互接続テストガイドライン」を利用することで、技術的問題が正しく検出された上で期待した結果を得られたと判断され、テストは有効に実施されたと言える。よって「認証局間相互接続テストガイドライン」は利用者にとって有効であると考える。

9.2 相互認証に係わる証明書失効

(1) 目的

本実験単位は、相互接続に係わる証明書失効を通しての「認証局間相互接続テストガイドライン」の有効性の検証を目的とする。

利用者に、認証局における証明書失効作業のテストを「認証局間相互接続テス

トガイドライン」に従って実施してもらい、「認証局間相互接続テストガイドライン」が、利用者にとって、テスト実施に必要な事前チェックの要件やテスト手順、及び実施結果の正確な確認方法についての必要十分な内容であることを確認する。さらに「認証局間相互接続テストガイドライン」に従ったテスト実施で期待した結果が得られることでテスト手順や期待値の妥当性を確認する。これらにより「認証局間相互接続テストガイドライン」の有効性の確認とする。

(2) 実験内容

現地タイ認証局との間で、相互接続モデルとして CC モデルを選択し、「認証局間相互接続テストガイドライン」の「Step 3: 証明書失効と検証(Certificate revocation and validation)」の認証局に係わるテスト項目を実施する。実験内容の概要を「図 9.2　現地タイ認証局との証明書失効テスト」に示す。

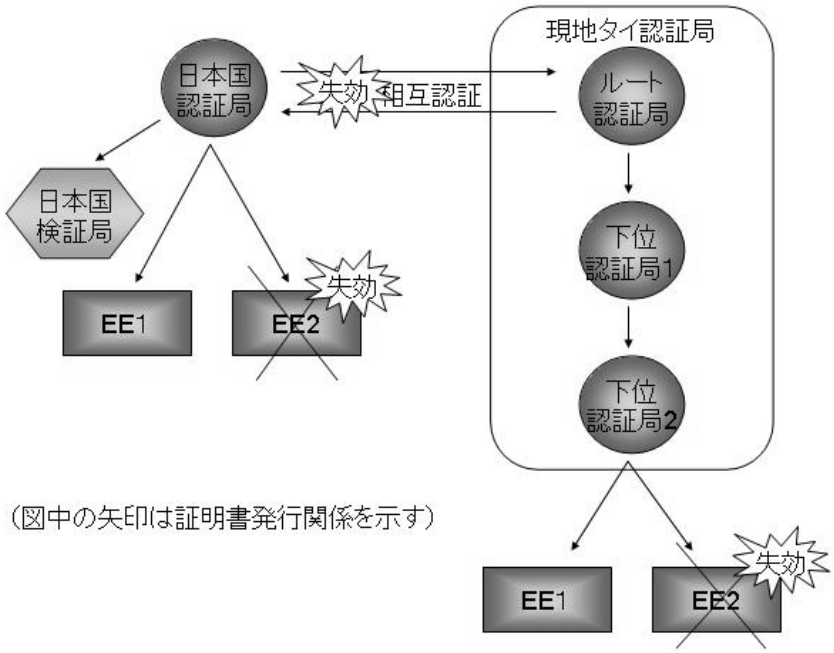


図 9.2　現地タイ認証局との証明書失効テスト

実施するテスト項目を「表 9.6　現地タイ認証局との証明書失効テスト項目一覧」に示す。

表 9.6　現地タイ認証局との証明書失効テスト項目一覧

No.	実験単位小項目	ガイドライン内のテスト項目	
		テスト ID	内容
1	(1-2-1)エンドエンティティ証明書の失効と CRL の発行	Step 3-1	Revoke an EE certificate.
2	(1-2-2)CRL のリポジトリへの格納	Step 3-2	Store the CRL in the repository.

3	(1-2-3)エンドエンティティ証明書のリポジトリからの削除	Step 3-3	[optional]Delete the EE certificate from the repository.
4	(1-2-4)エンドエンティティ証明書の検証	Step 3-4	[optional] Verify the valid EE certificates.
5	(1-2-5)相互認証証明書の失効と ARL の発行	Step 3-5	Revoke the cross certificate.
6	(1-2-6) ARL のリポジトリへの格納	Step 3-6	Store the ARL(or CRL) in the repository.
7	(1-2-7)相互認証証明書ペアのリポジトリからの削除	Step 3-7	Delete the cross certificate pair from the repository.
8	(1-2-8)エンドエンティティ証明書の検証	Step 3-8	[optional] Verify the EE certificate.

(3) 実験方法

「表 9.6　現地タイ認証局との証明書失効テスト項目一覧」に示した項目について、「認証局間相互接続テストガイドライン」の定める手順に従ってテストを実施する。

日本国認証局は、認証局のオペレーションである Step 3-1 から Step 3-3 を実施しエンドエンティティ証明書を失効する。日本国検証局による Step3-4 実施後、Step3-5 から Step3-7 までを実施し、現地タイ認証局との間の相互認証関係を解消する。

日本国検証局を用いて Step3-4 及び Step3-8 を実施する。Step3-4 及び Step3-8 では Step 1-3 で発行したエンドエンティティ証明書を検証する。実験にあたっては、利用者端末及び日本国検証局の設定が、「表 9.7　証明書失効テストにおける証明書検証モジュールへのインプット情報」を満たす設定となっていることを確認する。

表 9.7　証明書失効テストにおける証明書検証モジュールへのインプット情報

No.	インプット情報	設定
1	受け入れ可能なポリシ情報 (user-initial-policy-set)	設定しない
2	トラストアンカ情報 (trust anchor information)	日本国認証局の自己署名証明書を設定
3	ポリシマッピングを許可するか否かの情報 (initial-policy-mapping-inhibit)	ポリシマッピングを許可
4	証明書パス中にユーザが受け入れ可能なポリシが入っていることを要求するか否かの情報 (initial-explicit-policy)	要求しない
5	any-policy を許可するか否かの情報 (initial-any-policy-inhibit)	許可する

エンドエンティティ証明書検証の実験結果期待値を「表 9.8　Step3-4 及び Step3-8 の実験結果期待値」に示す。

表 9.8　Step3-4及びStep3-8の実験結果期待値

No	テスト ID		期待値		備考
			結果	Return Code	
1	Step3-4	Step3-4-1	Valid	0	「Return Code=0」は証明書が有効であることを示す。
2		Step3-4-2	Invalid	203	「Return Code=203」は証明書パスを構築する証明書のいずれかが失効していることを示す。
3	Step3-8	Step3-8-1	Invalid	101	「Return Code=101」は証明書パスが存在しないことを示す。

Step 3-3 が完了した時点で、アプリケーション利用者の立場から認証局間相互接続ガイドラインの有効性を検証するため、PKI を適用したアプリケーションを使用して電子署名の付与及びパス検証のテストを行った。

PKI を適用したアプリケーションとしては e-Procurement に関する業務システムを用い、図 9.3　業務アプリケーションの処理フロー」に示す一連の業務フローのうち電子署名の付与・検証を行う(2)から(5)の処理について、各国端末を使用して実行する。

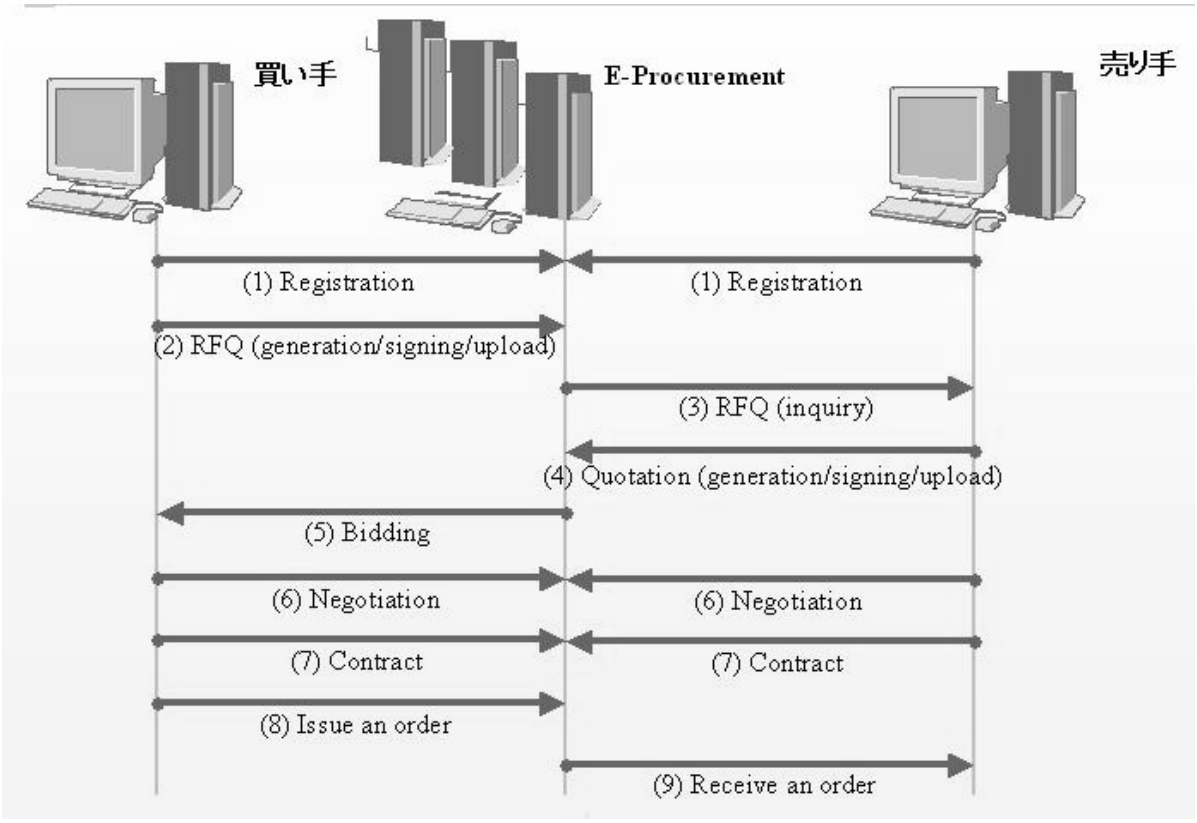


図 9.3　業務アプリケーションの処理フロー

この処理フローを用い、買い手と売り手を入れ替えた下記のテストケースについてテストを行う。

表 9.9　証明書失効テストにおける業務フローテストケース一覧

No.	内　　容	パターン
1	買い手側・正常系証明書、売り手側・正常系証明書を使用し、業務フローに沿って一連の処理を行う。検証モデルは EE モデルとする。	1)買い手：JP-EE 売り手：TH-EE 2)買い手：TH-EE 売り手：JP-EE
2	買い手側・正常系証明書、売り手側・失効系証明書を使用し、業務フローに沿って一連の処理を行う。検証モデルは EE モデルとする。	1)買い手：JP-EE 売り手：TH-EE 2)買い手：TH-EE 売り手：JP-EE
3	買い手側・失効系証明書、売り手側・正常系証明書を使用し、業務フローに沿って一連の処理を行う。検証モデルは EE モデルとする。	1)買い手：JP-EE 売り手：TH-EE 2)買い手：TH-EE 売り手：JP-EE

(4) 実験結果

「認証局間相互接続テストガイドライン」の定めるテスト項目の実施結果を「表 9.10　現地タイ認証局との証明書失効テスト実施結果」に示す。

表 9.10　現地タイ認証局との証明書失効テスト実施結果

テスト ID		実施結果
Step3-1	Step3-1-1	問題なし
	Step3-1-2	問題なし
	Step3-1-3	問題なし
	Step3-1-4	問題なし
Step3-2	Step3-2-1	問題なし
	Step3-2-2	問題なし
	Step3-2-3	問題なし
	Step3-2-4	問題なし
	Step3-2-5	問題なし
Step3-3	Step3-3-1	問題なし
	Step3-3-2	問題なし
Step3-4	Step3-4-1	問題なし
	Step3-4-2	問題なし
Step3-5	Step3-5-1	問題なし
	Step3-5-2	問題なし
	Step3-5-3	問題なし
	Step3-5-4	問題なし
Step3-6	Step3-6-1	問題なし
	Step3-6-2	問題なし
	Step3-6-3	問題なし
	Step3-6-4	問題なし
	Step3-6-5	問題なし
Step3-7	Step3-7-1	問題なし
	Step3-7-2	問題なし
Step3-8	Step3-8-1	問題なし

国際間調達システムを使用したアプリケーションテストの結果を表 9.11　証明書失効テストにおけるアプリケーションテストの結果一覧に示す。

表 9.11　証明書失効テストにおけるアプリケーションテストの結果一覧

no	パターン	買い手	売り手	期待値	結果	判定
1	正常系(1)	日本・正常証明書	現地タイ・正常証明書	○	○	○
2	正常系(2)	現地タイ・正常証明書	日本・正常証明書	○	○	○
3	失効系 1(1)	日本・正常証明書	現地タイ・失効証明書	×	×	○
4	失効系 1(2)	現地タイ・正常証明書	日本・失効証明書	×	×	○
5	失効系 2(1)	日本・失効証明書	現地タイ・正常証明書	×	×	○
6	失効系 2(2)	現地タイ・失効証明書	日本・正常証明書	×	×	○

日本国検証局のログ情報を解析し、エンドエンティティ証明書検証結果を評価

した。結果を「表 9.12　現地タイ認証局との証明書失効テストにおけるエンドエンティティ証明書検証結果」に示す。

表 9.12　現地タイ認証局との証明書失効テストにおけるエンドエンティティ証明書検証結果

No.	テスト ID	期待値		検証結果	
		結果	Return Code	結果	Return Code
1	Step3-4-1	Valid	0	Valid	0
2	Step3-4-2	Invalid	203	Invalid	203
3	Step3-8-1	Invalid	101	Invalid	101

(5) 評価及び考察

本実験単位では、「表 9.10　現地タイ認証局との証明書失効テスト実施結果」に示す通りすべてのテスト項目について期待した結果を得ることができた。

これらの結果から、利用者が認証局における証明書失効作業のテストを行う際に「認証局間相互接続テストガイドライン」を利用することで期待した結果を得られたと判断され、「認証局間相互接続テストガイドライン」は利用者にとって有効であると考える。

9.3　相互認証に係わる証明書更新

(1) 目的

本実験単位は、相互接続に係わる証明書の更新を通しての「認証局間相互接続テストガイドライン」の有効性の検証を目的とする。

利用者に、認証局における証明書更新作業のテストを「認証局間相互接続テストガイドライン」に従って実施してもらい、「認証局間相互接続テストガイドライン」が、利用者にとって、テスト実施に必要な事前チェックの要件やテスト手順、及び実施結果の正確な確認方法についての必要十分な内容であることを確認する。さらに「認証局間相互接続テストガイドライン」に従ったテスト実施で期待した結果が得られることでテスト手順や期待値の妥当性を確認する。これらにより「認証局間相互接続テストガイドライン」の有効性の確認とする。

(2) 実験内容

現地タイ認証局との間で、相互接続モデルとして CC モデルを選択し、「認証局間相互接続テストガイドライン」の「Step 4: 証明書更新(Certificate update)」の認証局に係わるテスト項目を実施する。

実験内容の概要を「図 9.4　現地タイ認証局との証明書更新テスト」に示す。

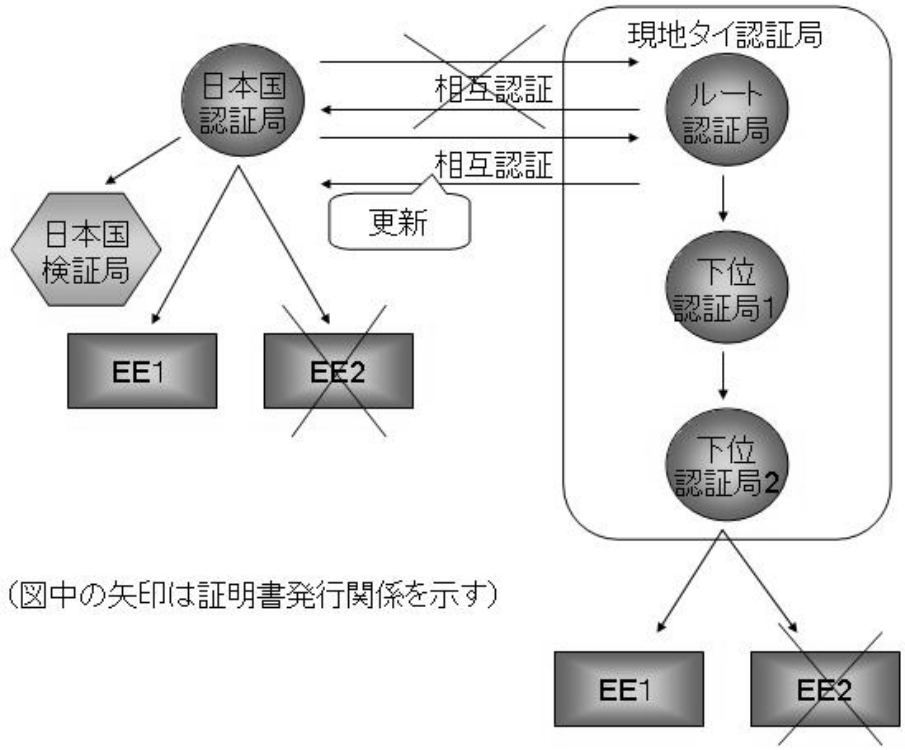


図 9.4　現地タイ認証局との証明書更新テスト

証明書更新とは、すでに発行された相互認証証明書を一旦失効し、発行し直す処理として「認証局間相互接続テストガイドライン」で定義している。すでに発行された相互認証証明書の失効は必須の手順とは定められておらずテスト実施者の選択に任されている。本実験単位では、一旦失効するものとし、相互認証証明書の失効は「9.2　相互認証に係わる証明書失効」で実施した状態を引き継ぐ。

実施するテスト項目を「表 9.13　現地タイ認証局との証明書更新テスト項目一覧」に示す。

表 9.13　現地タイ認証局との証明書更新テスト項目一覧

No.	実験単位小項目	ガイドライン内のテスト項目	
		テスト ID	内容
1	-	Step 4-1	Request Cross certification to the partner CA.
2	(1-3-1)相互認証証明書の更新	Step 4-2	Issue a cross certificate (issuedByThisCA)
3	(1-3-2)相互認証証明書ペアの作成	Step 4-3	Make a cross certificate pair.
4	(1-3-3)相互認証証明書ペアのリポジトリへの格納	Step 4-4	Store the cross certificate pair into the repository
5	(1-3-4)エンドエンティティ証明書の検証	Step 4-5	Verify the EE certificates.

「表 9.13　現地タイ認証局との証明書更新テスト項目一覧」中の網掛けとしている Step4-1 に相当する部分は、本実験単位では対応する実験単位小項目はない。これは認証局の鍵を更新した場合を対象としており「認証局間相互接続テストガイドライン」では実施任意の項目である。

(3) 実験方法

「表 9.13　現地タイ認証局との証明書更新テスト項目一覧」に示した項目について、「認証局間相互接続テストガイドライン」の定める手順に従ってテストを実施する。

日本国認証局は、認証局のオペレーションである Step 4-2 から Step4-4 までを実施し、現地タイ認証局との間で相互認証関係を再構築する。Step 4-1 は認証局の鍵を変更しないので実施しない。

日本国検証局を用いて Step4-5 を実施する。Step4-5 では Step 1-3 で発行したエンドエンティティ証明書を検証する。実験にあたっては、利用者端末及び日本国検証局の設定が、「表 9.14　証明書更新テストにおける証明書検証モジュールへのインプット情報」を満たす設定となっていることを確認する。

表 9.14　証明書更新テストにおける証明書検証モジュールへのインプット情報

No.	インプット情報	設定
1	受け入れ可能なポリシ情報 (user-initial-policy-set)	設定しない
2	トラストアンカ情報 (trust anchor information)	日本国認証局の自己署名証明書を設定
3	ポリシマッピングを許可するか否かの情報 (initial-policy-mapping-inhibit)	ポリシマッピングを許可
4	証明書パス中にユーザが受け入れ可能なポリシが入っていることを要求するか否かの情報 (initial-explicit-policy)	要求しない
5	any-policy を許可するか否かの情報 (initial-any-policy-inhibit)	許可する

エンドエンティティ証明書検証の実験結果期待値を「表 9.15　Step4-5 の実験結果期待値」に示す。

表 9.15　Step4-5の実験結果期待値

No	テスト ID		期待値		備考
			結果	Return Code	
1	Step4-5	Step4-5-1	Valid	0	「Return Code=0」は証明書が有効であることを示す。
2		Step4-5-2	Invalid	203	「Return Code=203」は証明書パスを構築する証明書のいずれかが失効していることを示す。

(4) 実験結果

「認証局間相互接続テストガイドライン」の定めるテスト項目の実施結果を「表 9.16　現地タイ認証局との証明書更新テスト実施結果」に示す。

表 9.16　現地タイ認証局との証明書更新テスト実施結果

テスト ID		実施結果
Step4-1	Step4-1-1 ~ Step4-1-6	認証局の鍵を変更せず非該当
Step4-2	Step4-2-1~ Step4-2-3	
	Step4-2-4	問題なし
	Step4-2-5	問題なし
	Step4-2-6	問題なし
	Step4-2-7	問題なし
	Step4-2-8	問題なし
	Step4-2-9	問題なし
	Step4-2-10	問題なし
Step4-3	Step4-3-1	問題なし
	Step4-3-2	問題なし
	Step4-3-3	問題なし
	Step4-3-4	問題なし
Step4-4	Step4-4-1	問題なし
	Step4-4-2	問題なし
	Step4-4-3	問題なし
	Step4-4-4	問題なし
	Step4-4-5	問題なし
	Step4-4-6	問題なし
Step4-5	Step4-5-1	問題なし
	Step4-5-2	問題なし

日本国検証局のログ情報を解析し、エンドエンティティ証明書検証結果を評価した。結果を「表 9.17　現地タイ認証局との証明書更新テストにおけるエンドエンティティ証明書検証結果」に示す。

表 9.17　現地タイ認証局との証明書更新テストにおけるエンドエンティティ証明書検証結果

No.	テスト ID	期待値		検証結果	
		結果	Return Code	結果	Return Code
1	Step4-5-1	Valid	0	Valid	0
2	Step4-5-2	Invalid	203	Invalid	203

(5) 評価及び考察

本実験単位では、「表 9.13　現地タイ認証局との証明書更新テスト項目一覧」に示す通りすべてのテスト項目について期待した結果を得ることができた。

これらの結果から、利用者が認証局における証明書更新作業のテストを行う際に「認証局間相互接続テストガイドライン」を利用することで期待した結果を得られたと判断され、「認証局間相互接続テストガイドライン」は利用者にとって有効であると考える。

9.4　相互認証に係わる証明書再発行

(1) 目的

本実験単位は、相互接続に係わる証明書再発行を通しての「認証局間相互接続テストガイドライン」の有効性の検証を目的とする。

利用者に、認証局における証明書再発行作業のテストを「認証局間相互接続テストガイドライン」に従って実施してもらい、「認証局間相互接続テストガイドライン」が、利用者にとって、テスト実施に必要な事前チェックの要件やテスト手順、及び実施結果の正確な確認方法についての必要十分な内容であることを確認する。さらに「認証局間相互接続テストガイドライン」に従ったテスト実施で期待した結果が得られることでテスト手順や期待値の妥当性を確認する。これらにより「認証局間相互接続テストガイドライン」の有効性の確認とする。

(2) 実験内容

現地タイ認証局との間で、相互接続モデルとして CC モデルを選択し、「認証局間相互接続テストガイドライン」の「Step 5: 証明書再発行(Certificate renewal)」の認証局に係わるテスト項目を実施する。

実験内容の概要を「図 9.5　現地タイ認証局との証明書再発行テスト」に示す。

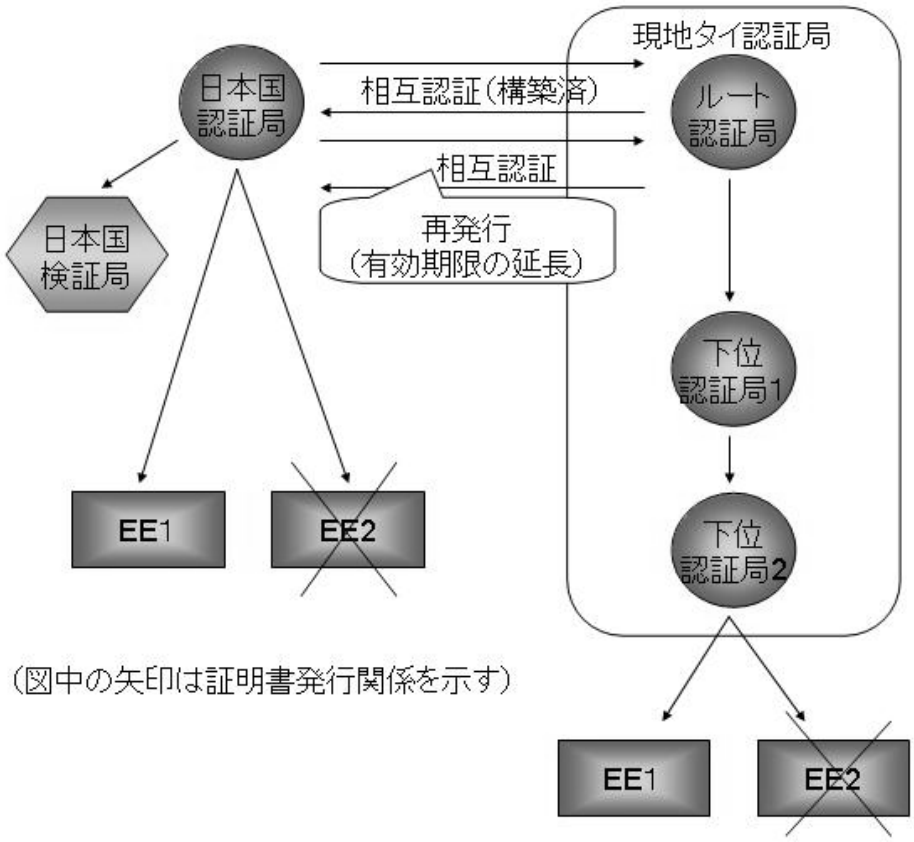


図 9.5　現地タイ認証局との証明書再発行テスト

証明書再発行とは、すでに発行された相互認証証明書の有効期限を延長して発行し直す処理として「認証局間相互接続テストガイドライン」で定義している。

実施するテスト項目を「表 9.18　現地タイ認証局との証明書再発行テスト項目一覧」に示す。

表 9.18　現地タイ認証局との証明書再発行テスト項目一覧

No.	実験単位小項目	ガイドライン内のテスト項目	
		テスト ID	内容
1	-	Step 5-1	Request Cross certification to the partner CA.
2	(1-4-1)相互認証証明書の再発行	Step 5-2	Issue a cross certificate (issuedByThisCA)
3	(1-4-2)相互認証証明書ペアの作成	Step 5-3	Make a cross certificate pair.
4	(1-4-3)相互認証証明書ペアのリポジトリへの格納	Step 5-4	Store the cross certificate pair into the repository
5	(1-4-4)エンドエンティティ証明書の検証	Step 5-5	Verify the EE certificates.

「表 9.18　現地タイ認証局との証明書再発行テスト項目一覧」中の網掛けとし

ている Step５-1 に相当する部分は、本実験単位では対応する実験単位小項目はない。これは認証局の鍵を更新した場合を対象としており「認証局間相互接続テストガイドライン」では実施任意の項目である。

(3) 実験方法

「表 9.18　現地タイ認証局との証明書再発行テスト項目一覧」に示した項目について、「認証局間相互接続テストガイドライン」の定める手順に従ってテストを実施する。

日本国認証局は、認証局のオペレーションである Step 5-2 から Step5-4 までを実施し、現地タイ認証局との間で相互認証関係の期間を延長する。Step 5-1 は認証局の鍵を変更しないので実施しない。

日本国検証局を用いて Step5-5 を実施する。Step5-5 では Step 1-3 で発行したエンドエンティティ証明書を検証する。実験にあたっては、利用者端末及び日本国検証局の設定が、「表 9.19　証明書際発行テストにおける証明書検証モジュールへのインプット情報」を満たす設定となっていることを確認する。

表 9.19　証明書際発行テストにおける証明書検証モジュールへのインプット情報

No.	インプット情報	設定
1	受け入れ可能なポリシ情報 (user-initial-policy-set)	設定しない
2	トラストアンカ情報 (trust anchor information)	日本国認証局の自己署名証明書を設定
3	ポリシマッピングを許可するか否かの情報 (initial-policy-mapping-inhibit)	ポリシマッピングを許可
4	証明書パス中にユーザが受け入れ可能なポリシが入っていることを要求するか否かの情報 (initial-explicit-policy)	要求しない
5	any-policy を許可するか否かの情報 (initial-any-policy-inhibit)	許可する

エンドエンティティ証明書検証の実験結果期待値を「表 9.20　Step5-5 の実験結果期待値」に示す。

表 9.20　Step5-5の実験結果期待値

No	テスト ID		期待値		備考
			結果	Return Code	
1	Step5-5	Step5-5-1	Valid	0	「Return Code=0」は証明書が有効であることを示す。

2		Step5-5-2	Invalid	203	「Return Code=203」は証明書パスを構築する証明書のいずれかが失効していることを示す。

(4) 実験結果

「認証局間相互接続テストガイドライン」の定めるテスト項目の実施結果を「表 9.21　現地タイ認証局との証明書再発行テスト実施結果」に示す。

表 9.21　現地タイ認証局との証明書再発行テスト実施結果

テスト ID		実施結果（1回目）	実施結果（2回目）
Step5-1	Step5-1-1	-	問題なし
	Step5-1-2	-	問題なし
	Step5-1-3	-	問題なし
	Step5-1-4	-	問題なし
	Step5-1-5	-	問題なし
	Step5-1-6	-	問題なし
Step5-2	Step5-2-1	-	問題なし
	Step5-2-2	-	問題なし
	Step5-2-3	-	問題なし
	Step5-2-4	問題なし	問題なし
	Step5-2-5	問題なし	問題なし
	Step5-2-6	問題なし	問題なし
	Step5-2-7	問題なし	問題なし
	Step5-2-8	問題なし	問題なし
	Step5-2-9	問題なし	問題なし
	Step5-2-10	問題なし	問題なし
Step5-3	Step5-3-1	問題なし	問題なし
	Step5-3-2	問題なし	問題なし
	Step5-3-3	問題なし	問題なし
	Step5-3-4	問題なし	問題なし
Step5-4	Step5-4-1	問題なし	問題なし
	Step5-4-2	問題なし	問題なし
	Step5-4-3	問題なし	問題なし
	Step5-4-4	問題なし	問題なし
	Step5-4-5	問題なし	問題なし
	Step5-4-6	問題なし	問題なし
Step5-5	Step5-5-1	データの有効期限不整合で実施不可	問題なし
	Step5-5-2	データの有効期限不整合で実施不可	問題なし

当初、当実験単位の項目は、認証局に関する項目は認証局の鍵を変更せずにテスト ID:Step5-1-1 から Step4-2-3 までの項目は非該当として「表 9.21　現地タイ認証局との証明書再発行テスト実施結果」の示す「実施結果（1回目)」のとおり期待値どおりの結果を得た。しかしその後エンドエンティティ証明書の検証において検証対象データの日付の不整合が発生したため、その対応策として検証対象データとなる証明書と証明書失効リストの日付を修正し、本実験単位を再度実施した。具体的には、現地タイ認証局による新しい下位認証局の自己署名証明書の発行と、新しいエンドエンティティ証明書の発行、日本国認証局による新しい自己署名証明書の発行とエンドエンティティ証明書の発行、及び相互認証証明書の再発行である。これらの手順を経て、すべての項目について期待した結果を得た。

日本国検証局のログ情報を解析し、エンドエンティティ証明書検証結果を評価した。結果を「表 9.22　現地タイ認証局との証明書再発行テストにおけるエンドエンティティ証明書検証結果」に示す。

表 9.22　現地タイ認証局との証明書再発行テストにおけるエンドエンティティ証明書検証結果

No.	テスト ID	期待値		検証結果	
		結果	Return Code	結果	Return Code
1	Step5-5-1	Valid	0	Valid	0
2	Step5-5-2	Invalid	203	Invalid	203

(5) 評価及び考察

本実験単位では、「表 9.21　現地タイ認証局との証明書再発行テスト実施結果」に示す通りすべてのテスト項目について期待した結果を得ることができた。しかし、1回目の実験実施の段階では、エンドエンティティ証明書の検証に用いるためにはデータの有効期限に不整合があって、実験再実施が必要となった。これは、現地タイ認証局の使用する認証局製品の実装上の制約に起因し、さらに以下に述べるいくつかの事象が重なったためである。

(a) 現地タイ認証局の使用する認証局製品が発行できる相互認証証明書の有効期限が最短で 2 ヶ月であることが、実験実施の段階で分かった。このため当初設計した有効期間どおりに相互認証証明書を発行することができなかった。

(b) (a)の制約の下発行された相互認証証明書と、現地タイ認証局の下位認証局 1 の証明書の有効期間が重なる期間が非常に短かった。

(c) (b)の期間に有効な証明書失効リスト（ARL）を現地タイ認証局の下位認証局

1 が発行することができなかった。

これらのことから、認証パスを構成する証明書と証明書失効リストが共通に有効な期間が設定できず、エンドエンティティ証明書の検証ができなかった。これを解決するために、「9.4(4)実験結果」に述べた対応を取った。以下、「認証局間相互接続テストガイドライン」の評価という観点から、この問題の発生はガイドラインの不備によるものか否かを評価する。「認証局間相互接続テストガイドライン」は、利用者の個別の要件は内容として含まないことで広く利用できるガイドラインとすべく設計したものであり、製品の実装の制限に依存する部分はテストを実施する利用者によって個別の対処を許すものとすべきである。この設計思想に照らせば、「認証局間相互接続テストガイドライン」の定めるテスト手順そのものは 2 回目の実験が問題なく実施できたことから現在の内容で妥当であるといえる。しかし、テスト実施にあたって適正なテスト計画を策定するための指針と、データの有効期限の設定に関する利用者への注意喚起の記載が必要と考え、「認証局間相互接続テストガイドライン」に反映した。

これらの結果から、利用者が認証局における証明書再発行作業のテストを行う際に「認証局間相互接続テストガイドライン」を利用することで技術的問題が正しく検出された上で期待した結果を得られたと判断され、「認証局間相互接続テストガイドライン」は利用者にとって有効であると考える。

10 仮想対象国認証局との認証局間相互接続テストガイドラインの実証実験

10.1　CC モデルにおける認証局間相互接続テストガイドラインの検証

(1) 目的

本実験単位は、日本国内に設置した実験環境で「認証局間相互接続テストガイドライン」の有効性の検証を行うものである。

利用者に、日本国認証局と日本国内の実験環境に設置した仮想対象国認証局との認証局間相互接続テストを「認証局間相互接続テストガイドライン」に従って実施してもらい、「認証局間相互接続テストガイドライン」の有効性を検証する。その際に、特に「認証局間相互接続テストガイドライン」の定めるテスト手順について下記の点についても確認する

- 「9現地タイ認証局との認証局間相互接続テストガイドラインの実証実験」において発生した、現地タイ認証局の使用する製品の実装上の制限によるテストデータの有効期限の不整合の問題を受けて、テスト手順が、利用者がテスト実施期間とテストデータの有効期限を矛盾なく設定することで無理なく実施できるものであることを確認する。
- テスト手順実施に必要な所要時間を計測して、利用者がテスト計画を立てる際の参考データとして「認証局間相互接続テストガイドライン」にフィードバックする。

(2) 実験内容

日本国内に設置した実験環境において、日本国認証局と仮想対象国認証局との間で、相互接続モデルとして CC モデルを選択し、「認証局間相互接続テストガイドライン」の定めるテスト項目を実施する。

実験内容の概要を「図 10.1　仮想対象国認証局との CC モデルテスト」に示す。

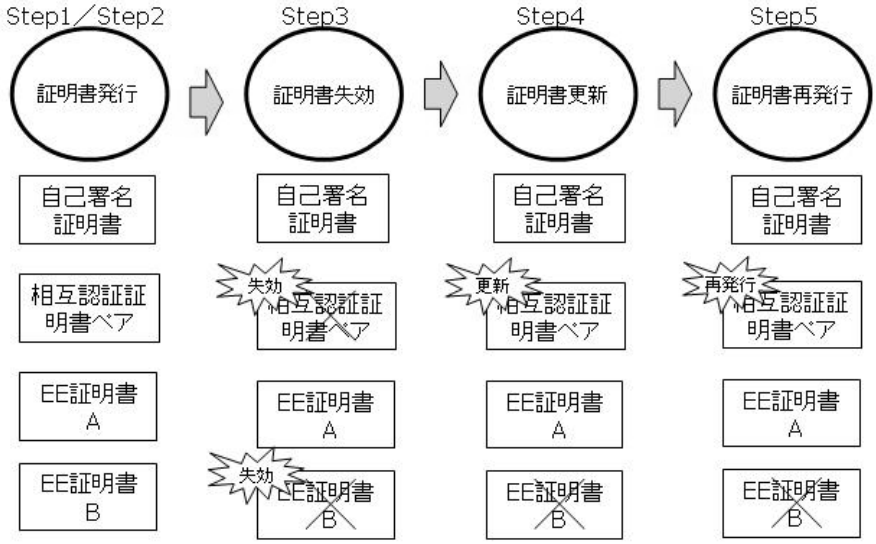


図 10.1　仮想対象国認証局との CC モデルテスト

テストを実施するにあたって、テスト実施時間を想定し、それに最適なデータの有効期限を設定して実施する。その前提条件となったのが下記の本実験単位の実施条件である。

(a) 日本国認証局及び仮想対象国認証局の操作員は相互認証作業の経験値は高く、トラブルシューティングの時間は織り込まなくて良い

(b) 使用コンポーネントはデータの有効期限設定に関して特に制約を持たない

この条件下で、短時間で効率良く進む場合のテスト計画を定め、それに従って無理なくテストが実施できることを確認する。実施結果として得られるテストの所要時間は「認証局間相互接続テストガイドライン」が利用者に提示するテスト実施計画策定の指針の基礎データとなる。

実施するテスト項目及び手順は「9現地タイ認証局との認証局間相互接続テストガイドラインの実証実験」で実施したものと同じである。しかし「10.1　目的」及び上記で述べたように、テスト実施の時間的要素に着目した評価を行うため、「9現地タイ認証局との認証局間相互接続テストガイドラインの実証実験」のようにテストステップ毎の個別の評価でなく、テストの全ステップをひとまとまりとして扱う。

(3) 実験方法

「認証局間相互接続テストガイドライン」のテストステップを下記タイムスケジュールに従って実施する。

表 10.1　仮想対象国認証局とのCCモデル検証タイムスケジュール

テスト ID	テスト実施		相互認証証明書	
	開始時間	終了時間	開始日時	終了日時
Step 1	準備作業として事前に実施		なし	なし
Step 2	11:00	12:00	040218000000Z	040331235959Z
Step 3	13:00	14:00	なし	なし
Step 4	14:00	15:00	040218000000Z	040218060000Z
Step 5	15:00	16:00	040218060000Z	040331235959Z

エンドエンティティ証明書の検証は、日本国検証局を用いて行う。

実験にあたっては、利用者端末及び日本国検証局の設定が、「表 9.14　証明書更新テストにおける証明書検証モジュールへのインプット情報」を満たす設定となっていることを確認する。

表 10.2　仮想対象国とのCCモデルテストにおける証明書検証モジュールへのインプット情報

No.	インプット情報	設定
1	受け入れ可能なポリシ情報 (user-initial-policy-set)	設定しない
2	トラストアンカ情報 (trust anchor information)	日本国認証局の自己署名証明書を設定
3	ポリシマッピングを許可するか否かの情報 (initial-policy-mapping-inhibit)	ポリシマッピングを許可
4	証明書パス中にユーザが受け入れ可能なポリシが入っていることを要求するか否かの情報 (initial-explicit-policy)	要求しない
5	any-policy を許可するか否かの情報 (initial-any-policy-inhibit)	許可する

エンドエンティティ証明書検証の実験結果期待値を「表 10.3　Step2-5 から Step5-5 の実験結果期待値」に示す。

表 10.3　Step2-5からStep5-5の実験結果期待値

No.	テスト ID		期待値		備考
			結果	Return Code	
1	Step2-5	Step2-5-1	Valid	0	「Return Code=0」は証明書が有効であることを示す。
2	Step3-4	Step3-4-1	Valid	0	
3		Step3-4-2	Invalid	203	「Return Code=203」は証明書パスを構築する証明書のいずれかが失効していることを示す。
4	Step3-8	Step3-8-1	Invalid	101	「Return Code=101」は証明書パスが存在しないことを示す。

5	Step4-5	Step4-5-1	Valid	0	
6		Step4-5-2	Invalid	203	「Return Code=203」は証明書パスを構築する証明書のいずれかが失効していることを示す。
7	Step5-5	Step5-5-1	Valid	0	
8		Step5-5-2	Invalid	203	「Return Code=203」は証明書パスを構築する証明書のいずれかが失効していることを示す。

(4) 実験結果

「認証局間相互接続テストガイドライン」の定めるテスト項目の実施結果を「表 10.4　仮想対象国認証局との CC モデルテスト実施結果」に示す。日本国認証局と仮想対象国認証局の両方でこの結果は同じである。

表 10.4　仮想対象国認証局とのCCモデルテスト実施結果

テスト ID		実施結果
Step1-1	Step1-1-1	問題なし
	Step1-1-2	問題なし
	Step1-1-3	問題なし
Step1-2	Step1-2-1	問題なし
	Step1-2-2	問題なし
	Step1-2-3	問題なし
	Step1-2-4	問題なし
	Step1-2-5	問題なし
Step1-3	Step1-3-1	問題なし
	Step1-3-2	問題なし
	Step1-3-3	問題なし
Step1-4	Step1-4-1	問題なし
	Step1-4-2	問題なし
	Step1-4-3	問題なし
	Step1-4-4	問題なし
	Step1-4-5	問題なし
Step1-5	Step1-5-1	問題なし
	Step1-5-2	問題なし
	Step1-5-3	問題なし
Step2-1	Step2-1-1	問題なし
	Step2-1-2	問題なし
	Step2-1-3	問題なし
	Step2-1-4	問題なし
	Step2-1-5	問題なし
	Step2-1-6	問題なし

Step2-2	Step2-2-1	問題なし
	Step2-2-2	問題なし
	Step2-2-3	問題なし
	Step2-2-4	問題なし
	Step2-2-5	問題なし
	Step2-2-6	問題なし
	Step2-2-7	問題なし
	Step2-2-8	問題なし
	Step2-2-9	問題なし
	Step2-2-10	問題なし
Step2-3	Step2-3-1	問題なし
	Step2-3-2	問題なし
	Step2-3-3	問題なし
	Step2-3-4	問題なし
Step2-4	Step2-4-1	問題なし
	Step2-4-2	問題なし
	Step2-4-3	問題なし
	Step2-4-4	問題なし
	Step2-4-5	問題なし
	Step2-4-6	問題なし
Step2-5	Step2-5-1	問題なし
Step3-1	Step3-1-1	問題なし
	Step3-1-2	問題なし
	Step3-1-3	問題なし
	Step3-1-4	問題なし
Step3-2	Step3-2-1	問題なし
	Step3-2-2	問題なし
	Step3-2-3	問題なし
	Step3-2-4	問題なし
	Step3-2-5	問題なし
Step3-3	Step3-3-1	問題なし
	Step3-3-2	問題なし
Step3-4	Step3-4-1	問題なし
	Step3-4-2	問題なし
Step3-5	Step3-5-1	問題なし
	Step3-5-2	問題なし
	Step3-5-3	問題なし
	Step3-5-4	問題なし
Step3-6	Step3-6-1	問題なし
	Step3-6-2	問題なし
	Step3-6-3	問題なし
	Step3-6-4	問題なし

	Step3-6-5	問題なし
Step3-7	Step3-7-1	問題なし
	Step3-7-2	問題なし
Step3-8	Step3-8-1	問題なし
Step4-1	Step4-1-1 ~ Step4-1-6	認証局の鍵を変更していない場合実施の必要がない項目であるため、実施していない
Step4-2	Step4-2-1 ~ Step4-2-3	
	Step4-2-4	問題なし
	Step4-2-5	問題なし
	Step4-2-6	問題なし
	Step4-2-7	問題なし
	Step4-2-8	問題なし
	Step4-2-9	問題なし
	Step4-2-10	問題なし
Step4-3	Step4-3-1	問題なし
	Step4-3-2	問題なし
	Step4-3-3	問題なし
	Step4-3-4	問題なし
Step4-4	Step4-4-1	問題なし
	Step4-4-2	問題なし
	Step4-4-3	問題なし
	Step4-4-4	問題なし
	Step4-4-5	問題なし
	Step4-4-6	問題なし
Step4-5	Step4-5-1	問題なし
	Step4-5-2	問題なし
Step5-1	Step5-1-1 ~ Step5-1-6	認証局の鍵を変更していない場合実施の必要がない項目であるため、実施していない
Step5-2	Step5-2-1~ Step5-2-3	
	Step5-2-4	問題なし
	Step5-2-5	問題なし
	Step5-2-6	問題なし
	Step5-2-7	問題なし
	Step5-2-8	問題なし
	Step5-2-9	問題なし
	Step5-2-10	問題なし
Step5-3	Step5-3-1	問題なし
	Step5-3-2	問題なし
	Step5-3-3	問題なし
	Step5-3-4	問題なし
Step5-4	Step5-4-1	問題なし

	Step5-4-2	問題なし
	Step5-4-3	問題なし
	Step5-4-4	問題なし
	Step5-4-5	問題なし
	Step5-4-6	問題なし
Step5-5	Step5-5-1	問題なし
	Step5-5-2	問題なし

日本国検証局のログ情報を解析し、エンドエンティティ証明書検証結果を評価した。結果を「表 10.5　仮想対象国認証局との CC モデルテストにおけるエンドエンティティ証明書検証結果」に示す。

表 10.5　仮想対象国認証局とのCCモデルテストにおけるエンドエンティティ証明書検証結果

No.	テスト ID	期待値		検証結果	
		結果	Return Code	結果	Return Code
1	Step2-5-1	Valid	0	Valid	0
2	Step 3-4-1	Valid	0	Valid	0
3	Step 3-4-2	Invalid	203	Invalid	203
4	Step 3-8-1	Invalid	101	Invalid	101
5	Step 4-5-1	Valid	0	Valid	0
6	Step 4-5-2	Invalid	203	Invalid	203
7	Step 5-5-1	Valid	0	Valid	0
8	Step 5-5-2	Invalid	203	Invalid	203

テスト実施に要した時間と発行した相互認証証明書の有効期限を「表 10.6　テスト実施時間」に示す。

表 10.6　テスト実施時間

テスト ID	テスト実施		相互認証証明書	
	開始時間	終了時間	開始日時	終了日時
Step 2	11:00	13:00	040218000000Z	040331235959Z
Step 3	14:00	15:00	なし	なし
Step 4	15:00	16:30	040218000000Z	040218073000Z
Step 5	16:30	15:15	040218073000Z	040331235959Z

(5) 評価及び考察

本実験単位では「表 10.4　仮想対象国認証局との CC モデルテスト実施結果」に示す通りすべてのテスト項目について期待した結果を得ることができた。

特に評価目的とした時間的な問題を分析する。「表 10.6　テスト実施時間」の示す通り、テスト所要時間は概ね計画通り各ステップ 1～2 時間以内で消化する

ことができた。それに合わせて設定した相互認証証明書の有効期限はエンドエンティティ証明書の検証時に矛盾なく有効であり、各ステップでのエンドエンティティ証明書の検証も問題なく実施できた。

今回のテスト実施にかかる所要時間は、短時間でのテスト実施の実績の一つであると考えられる。今回は実験環境で仮想対象国認証局を相手としての実験であるが、実際に運用している認証局がテストを行う場合には、認証局の運用ポリシに従った手順やオペレーションの冗長性（操作において権限のある責任者の承認を待つ、等）があると思われ、テスト項目の単純な消化以上の期間が必要となると考えられる。また、相互認証の経験の浅いテスト主体の場合にはトラブルシューティングの時間を見込んでおくことを、昨年度、一昨年度の実験の実績から見ても、強く推奨したい。本実験単位の実施結果から、テスト手順自体は短時間の実施も可能であることが実績として明らかになった。しかし、利用者がテスト計画を策定するにあたっては、下記の要素を加味してテスト期間と証明書や証明書失効リストの有効期間を決定しなければならない。

- テスト実施者となる認証局のポリシとして、オペレーションにどれだけの時間が必要か.（認証局の仕様による）
- テスト主体の相互接続についてのスキル
- 使用コンポーネントの機能的な制約

本実験単位の結果を受けた上記内容を、「認証局間相互接続テストガイドライン」に追記した。

以上から、「認証局間相互接続テストガイドライン」は利用者による適正な実施計画で有効に利用できることが確認されたと考える。

ヒアリングによる利用者の評価については他の実験単位と合わせて「13検証結果」で分析し考察するものとする。

10.2　CR モデルにおける認証局間相互接続テストガイドラインの検証

(1) 目的

本実験単位は、日本国内に設置した実験環境で「認証局間相互接続テストガイドライン」の有効性の検証を行うものである。

「認証局間相互接続テストガイドライン」は、認証局間の相互接続に適用するモデルによって選択できるよう、CC モデルと CR モデルの 2 つのテスト手順を定めている。「9現地タイ認証局との認証局間相互接続テストガイドラインの実証実験」で述べたタイとの実験では CC モデルを選択した。本実験単位ではもう一つの CR モデルのテスト手順を有効性検証の対象とする。

日本国内に設置した仮想対象国認証局との間で、利用者に CR モデルの相互接続テストを「認証局間相互接続テストガイドライン」に従って実施してもらい、「認証局間相互接続テストガイドライン」が、利用者にとって、テスト実施に必要な事前チェックの要件やテスト手順、及び実施結果の正確な確認方法について

の必要十分な内容であることを確認する。さらに「認証局間相互接続テストガイドライン」に従ったテスト実施で期待した結果が得られることでテスト手順や期待値の妥当性を確認する。これらにより「認証局間相互接続テストガイドライン」の有効性の確認とする。

(2) 実験内容

日本国内に設置した実験環境において、日本国認証局と仮想対象国認証局との間で、相互接続モデルとして CR モデルを選択し、「認証局間相互接続テストガイドライン」の定めるテスト項目を実施する。

実験内容の概要を「図 10.2　仮想対象国認証局との CR モデルテスト」に示す。

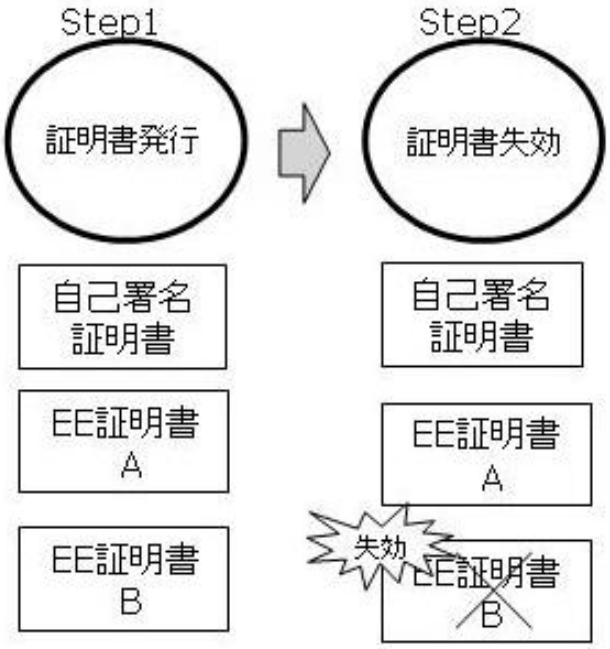


図 10.2　仮想対象国認証局との CR モデルテスト

実施するテスト項目を「表 10.7　仮想対象国認証局との CR モデルテスト項目一覧」に示す。

表 10.7　仮想対象国認証局とのCRモデルテスト項目一覧

No.	実験単位中項目	ガイドライン内のテスト項目	
		テスト ID	内容
1	(2-2)CR モデルにおける認証局間相互接続テストガイドラインの検証	Step 1-1	Issue a CA's self-signed certificate
2		Step 1-2	Store the self-signed certificate in repository.
3		Step 1-3	Issue EE certificates

4		Step 1-4	Store EE certificates in repository.
5		Step 1-5	Issue a VA certificate.
6		Step 2-1	Send the trust anchor certificate to the pertoner.
7		Step 2-2	Receive the trust anchor certificate from the pertoner.
8		Step 3-1	Revoke an EE certificate.
9		Step 3-2	Store the CRL in the repository.
10		Step 3-3	Step 3-3: Verify the valid EE certificate.

(3) 実験方法

「表 10.7　仮想対象国認証局との CR モデルテスト項目一覧」に示した項目について、「認証局間相互接続テストガイドライン」の定める手順に従ってテストを実施する。

日本国認証局は、認証局のオペレーションである Step 1-1 から Step3-2 までを実施する。

日本国検証局を用いて Step3-3 を実施する。Step3-3 では仮想対象国認証局が発行したエンドエンティティ証明書を検証する。実験にあたっては、利用者端末及び日本国検証局の設定が、「表 10.8　仮想対象国との CR モデルテストにおける証明書検証モジュールへのインプット情報」を満たす設定となっていることを確認する。

表 10.8　仮想対象国とのCRモデルテストにおける証明書検証モジュールへのインプット情報

No.	インプット情報	設定
1	受け入れ可能なポリシ情報 (user-initial-policy-set)	設定しない
2	トラストアンカ情報 (trust anchor information)	仮想対象国認証局の自己署名証明書を設定
3	ポリシマッピングを許可するか否かの情報 (initial-policy-mapping-inhibit)	ポリシマッピングを許可
4	証明書パス中にユーザが受け入れ可能なポリシが入っていることを要求するか否かの情報 (initial-explicit-policy)	要求しない
5	any-policy を許可するか否かの情報 (initial-any-policy-inhibit)	許可する

エンドエンティティ証明書検証の実験結果期待値を「表 10.9　Step3-3 の実験結果期待値」に示す。

表 10.9　Step3-3の実験結果期待値

No	テスト ID		期待値		備考
			結果	Return Code	
1	Step3-3	Step3-3-1	Valid	0	「Return Code=0」は証明書が有効であることを示す。
2		Step3-3-2	Invalid	203	「Return Code=203」は証明書パスを構築する証明書が失効していることを示す。

Step 3-2 が完了した時点で、アプリケーション利用者の立場から認証局間相互接続ガイドラインの有効性を検証するため、PKI を適用したアプリケーションを使用して電子署名の付与及びパス検証のテストを行った。

PKI を適用したアプリケーションとしては e-Procurement に関する業務システムを用い、図 9.3　業務アプリケーションの処理フロー」に示す一連の業務フローのうち電子署名の付与・検証を行う(2)から(5)の処理について、各国の証明書を格納した各国端末を使用して実行する。

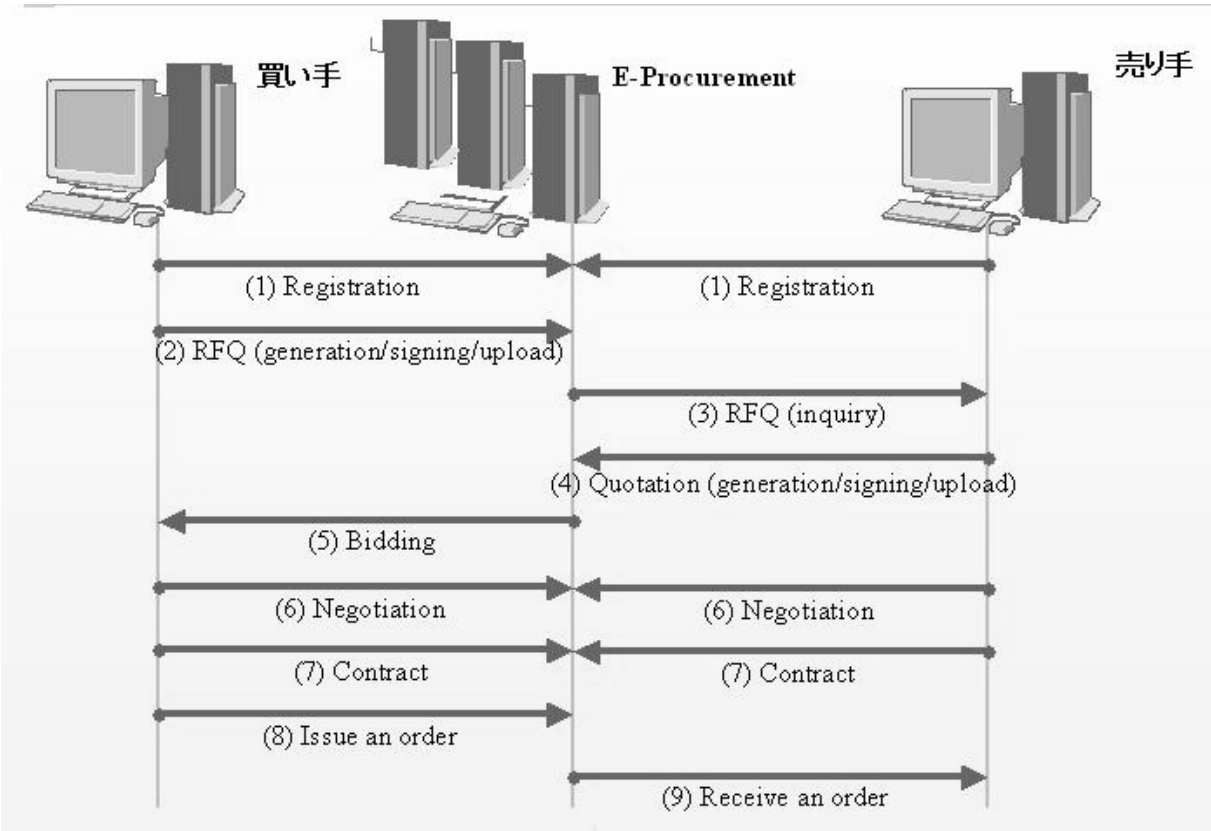


図 10.3　業務アプリケーションの処理フロー

この処理フローを用い、買い手と売り手を入れ替えた下記のテストケースについてテストを行う。

表 10.10　仮想対象国とのCRモデルテストにおける業務フローテストケース一覧

No.	内　　容	パターン
1	買い手側・正常系証明書、売り手側・正常系証明書を使用し、業務フローに沿って一連の処理を行う。検証モデルは EE モデルとする。	1)買い手：JP-EE 売り手：TH-EE 2)買い手：TH-EE 売り手：JP-EE
2	買い手側・正常系証明書、売り手側・失効系証明書を使用し、業務フローに沿って一連の処理を行う。検証モデルは EE モデルとする。	1)買い手：JP-EE 売り手：TH-EE 2)買い手：TH-EE 売り手：JP-EE
3	買い手側・失効系証明書、売り手側・正常系証明書を使用し、業務フローに沿って一連の処理を行う。検証モデルは EE モデルとする。	1)買い手：JP-EE 売り手：TH-EE 2)買い手：TH-EE 売り手：JP-EE

(4) 実験結果

「表 10.11　仮想対象国認証局との CR モデルテスト実施結果」に示す。

表 10.11　仮想対象国認証局とのCRモデルテスト実施結果

テスト ID		実施結果
Step1-1	Step1-1-1	問題なし
	Step1-1-2	問題なし
	Step1-1-3	問題なし
Step1-2	Step1-2-1	問題なし
	Step1-2-2	問題なし
	Step1-2-3	問題なし
	Step1-2-4	問題なし
	Step1-2-5	問題なし
Step1-3	Step1-3-1	問題なし
	Step1-3-2	問題なし
	Step1-3-3	問題なし
Step1-4	Step1-4-1	問題なし
	Step1-4-2	問題なし
	Step1-4-3	問題なし
	Step1-4-4	問題なし
	Step1-4-5	問題なし
Step1-5	Step1-5-1	問題なし
	Step1-5-2	問題なし
	Step1-5-3	問題なし
Step2-1	Step2-1-1	問題なし
	Step2-1-2	問題なし
	Step2-1-3	問題なし

Step2-2	Step2-2-1	問題なし
	Step2-2-2	問題なし
	Step2-2-3	問題なし
Step3-1	Step3-1-1	問題なし
	Step3-1-2	問題なし
	Step3-1-3	問題なし
	Step3-1-4	問題なし
Step3-2	Step3-2-1	問題なし
	Step3-2-2	問題なし
	Step3-2-3	問題なし
	Step3-2-4	問題なし
	Step3-2-5	問題なし
Step3-3	Step3-3-1	問題なし
	Step3-3-2	問題なし

国際間調達システムを使用したアプリケーションテストの結果を表 9.11　証明書失効テストにおけるアプリケーションテストの結果一覧に示す。

表 10.12　仮想対象国とのCRモデルテストにおけるアプリケーションテストの結果一覧

no	パターン	買い手	売り手	期待値	結果	判定
1	正常系(1)	日本・正常証明書	仮想対象国・正常証明書	○	○	○
2	正常系(2)	仮想対象国･正常証明書	日本・正常証明書	○	○	○
3	失効系 1(1)	日本・正常証明書	仮想対象国・失効証明書	×	×	○
4	失効系 1(2)	仮想対象国･正常証明書	日本・失効証明書	×	×	○
5	失効系 2(1)	日本・失効証明書	仮想対象国・正常証明書	×	×	○
6	失効系 2(2)	仮想対象国･失効証明書	日本・正常証明書	×	×	○

日本国検証局のログ情報を解析し、エンドエンティティ証明書検証結果を評価した。結果を「表 10.13　仮想対象国認証局との CR モデルにおけるエンドエンティティ証明書検証結果」に示す。

表 10.13　仮想対象国認証局とのCRモデルにおけるエンドエンティティ証明書検証結果

No.	テスト ID	期待値		検証結果	
		結果	Return Code	結果	Return Code
1	Step3-3-1	Valid	0	Valid	0
2	Step3-3-2	Invalid	203	Invalid	203

(5) 評価及び考察

本実験単位では、「表 10.11　仮想対象国認証局との CR モデルテスト実施結

果」に示す通りすべてのテスト項目について期待した結果を得ることができた。

これらの結果から、利用者が認証局における証明書失効作業のテストを行う際に「認証局間相互接続テストガイドライン」を利用することで期待した結果を得られたと判断され、「認証局間相互接続テストガイドライン」は利用者にとって有効であると考える。

11 パス検証テストガイドラインの実証実験

本章では、「パス検証テストガイドラインの実証実験」について述べる。

(1) 目的

本実験では、「パス検証テストガイドライン」を利用者に利用してもらい「パス検証テストガイドライン」の利用者にとっての有効性を明らかにする実験を行う。その際、「パス検証テストツールの開発」で開発されたツールを用いることにより「パス検証テストガイドラインの有効性の検証」の実証実験がより効率的に行えるようになったか、作業負担が軽減されたかを検証する。

また、実験実施にあたっては「表 11.1　パス検証テストガイドラインの実証実験における実験要件」に示す要件を満たす。

表 11.1　パス検証テストガイドラインの実証実験における実験要件

No.	実験要件
1	利用者は実験参加国/地域における認証局の設計、運用、又はソフトウェア開発に携わった経験があること。
2	ヒアリングがガイドラインの網羅性、テストパターンに関する妥当性、及びテスト期待値の的確さを含むこと。
3	パス検証テストツールに関するヒアリングは実証実験の作業負担の軽減、テストパターン設計の負担軽減などに効果があったか前年度との比較を含むこと。

(2) 実験内容

本実験では、本年度新たに「パス検証テストガイドライン」へ追加したテスト項目の実験を実施する。「パス検証テストガイドライン」に則って発行した証明書を、検証局を利用して検証する。検証を行うたびに得られた実験データを収集・分析する。

実験項目を、「表 11.2　パス検証テストの実験項目」に示す。なお、テスト項目の詳細については、「パス検証テストガイドライン」を参照のこと。

表 11.2　パス検証テストの実験項目一覧

No.	実験項目(小項目)	内容
1	Int.SH.DN.RP.01.01	DN マッチングテストのノーマルケーステスト
2	Int.SH.DN.RP.02.01	DN に Whitespace を含む場合の DN マッチングテスト
3	Int.SH.DN.RP.03.01	DN に大文字・小文字の違いがある文字列を含む場合の DN マッチングテスト
4	Int.SH.DN.RP.04.01	比較する DN の ASN.1 エンコーディング方式が異なる場合の DN マッチングテスト
5	Int.SH.DN.RP.05.01	RDN の並び順が異なる場合の DN マッチングテスト
6	Int.SH.DN.RP.06.01	DN が全く違う場合の DN マッチングテスト

7	Int.SH.DN.RP.07.01	DN に CJK キャラクタを含む場合の DN マッチングテスト
8	Int.SH.LDAPURI.RP.01.01	LDAPURI テストのノーマルテスト (cRLDP.distPoint.fullname に CertificateRevocationList 属性がある場合)
9	Int.SH.LDAPURI.RP.01.02	LDAPURI テストのノーマルテスト (cRLDP.distPoint.fullname に AuthorityRevocationList 属性がある場合)
10	Int.SH.LDAPURI.RP.02.01	LDAPURI のデリミターの両側に Whitespace がある場合のテスト
11	Int.SH.LDAPURI.RP.03.01	LDAPURI のデリミターとして"="を使用している場合のテスト
12	Int.SH.LDAPURI.RP.04.01	LDAPURI のデリミターとして semicolon";"を使用している場合のテスト
13	Int.SH.LDAPURI.RP.05.01	LDAPURI の中にエスケープ処理された文字列を含む場合のテスト(¥.(escape) -> %5c,)
14	Int.SH.LDAPURI.RP.05.02	LDAPURI の中にエスケープ処理された文字列を含む場合のテスト(¥.(escape) -> %5c2c)
15	Int.SH.LDAPURI.RP.05.03	LDAPURI の中にエスケープ処理された文字列を含む場合のテスト ("cn=AA,o=Sub" (escape) -> %22 cn=AA,o=Sub%22)
16	Int.SH.LDAPURI.RP.06.01	LDAPURI が示す LDAP サーバのポート番号が"389"以外の場合のテスト
17	Int.SH.CJK.RP.01.01	DN に Unicode "CJK Unified Ideographs(4E00-9FAF)"を含む場合のテスト (cDP 及び iDP が DN の場合)
18	Int.SH.CJK.RP.01.02	DN に Unicode "CJK Unified Ideographs(4E00-9FAF)"を含む場合のテスト (cDP 及び iDP が LDAPURI の場合)
19	Int.SH.CJK.RP.01.03	DN に Unicode "CJK Unified Ideographs(4E00-9FAF)"を含む場合のテスト (cDP 及び iDP が back slash でエスケープされた文字列を含む LDAPURI 場合)
20	Int.SH.CJK.RP.02.01	DN に Unicode "CJK Compatibility Ideographs(F900-FAFF)"を含む場合のテスト (cDP 及び iDP が DN の場合)
21	Int.SH.CJK.RP.02.02	DN に Unicode "CJK Compatibility Ideographs(F900-FAFF)"を含む場合のテスト (cDP 及び iDP が LDAPURI の場合)
22	Int.SH.CJK.RP.02.03	DN に Unicode "CJK Compatibility Ideographs(F900-FAFF)"を含む場合のテスト (cDP 及び iDP が back slash でエスケープされた文字列を含む LDAPURI 場合)

23	Int.SH.CJK.RP.03.01	DN に Unicode "Hiragana(3040-309F)"を含む場合のテスト (cDP 及び iDP が DN の場合)
24	Int.SH.CJK.RP.03.02	DN に Unicode "Hiragana(3040-309F)"を含む場合のテスト (cDP 及び iDP が LDAPURI の場合)
25	Int.SH.CJK.RP.03.03	DN に Unicode "Hiragana(3040-309F)"を含む場合のテスト (cDP 及び iDP が back slash でエスケープされた文字列を含む LDAPURI 場合)
26	Int.SH.CJK.RP.04.01	DN に Unicode "Katakana(30A0-30FF)"を含む場合のテスト (cDP 及び iDP が DN の場合)
27	Int.SH.CJK.RP.04.02	DN に Unicode "Katakana(30A0-30FF)"を含む場合のテスト (cDP 及び iDP が LDAPURI の場合)
28	Int.SH.CJK.RP.04.03	DN に Unicode "Katakana(30A0-30FF)"を含む場合のテスト (cDP 及び iDP が back slash でエスケープされた文字列を含む LDAPURI 場合)
29	Int.SH.CJK.RP.05.01	DN に Unicode "Halfwidth and Fullwidth Forms(FF00-FFEF)"を含む場合のテスト (cDP 及び iDP が DN の場合)
30	Int.SH.CJK.RP.05.02	DN に Unicode "Halfwidth and Fullwidth Forms(FF00-FFEF)"を含む場合のテスト (cDP 及び iDP が LDAPURI の場合)
31	Int.SH.CJK.RP.05.03	DN に Unicode "Halfwidth and Fullwidth Forms(FF00-FFEF)"を含む場合のテスト (cDP 及び iDP が back slash でエスケープされた文字列を含む LDAPURI 場合)
32	Int.SH.CJK.RP.06.01	DN に Unicode "Hangul Syllables(AC00-D7AF)"を含む場合のテスト (cDP 及び iDP が DN の場合)
33	Int.SH.CJK.RP.06.02	DN に Unicode "Hangul Syllables(AC00-D7AF)"を含む場合のテスト (cDP 及び iDP が LDAPURI の場合)
34	Int.SH.CJK.RP.06.03	DN に Unicode "Hangul Syllables(AC00-D7AF)"を含む場合のテスト (cDP 及び iDP が back slash でエスケープされた文字列を含む LDAPURI 場合)
35	Int.SH.CJK.RP.07.01	DN に Unicode "CJK Symbols and Punctuations"を含む場合のテスト (cDP 及び iDP が DN の場合)

36	Int.SH.CJK.RP.07.02	DN に Unicode "CJK Symbols and Punctuations"を含む場合のテスト (cDP 及び iDP が LDAPURI の場合)
37	Int.SH.CJK.RP.07.03	DN に Unicode "CJK Symbols and Punctuations"を含む場合のテスト (cDP 及び iDP が back slash でエスケープされた文字列を含む LDAPURI 場合)
38	Int.SH.CJK.RP.08.01	DN に Unicode CJK 及び ASCII characters を含む場合のテスト (cDP 及び iDP が DN の場合)
39	Int.SH.CJK.RP.08.02	DN に Unicode CJK 及び ASCII characters を含む場合のテスト (cDP 及び iDP が LDAPURI の場合)
40	Int.SH.CJK.RP.08.03	DN に Unicode CJK 及び ASCII characters を含む場合のテスト (cDP 及び iDP が back slash でエスケープされた文字列を含む LDAPURI 場合)

(3) 実験方法

「表 11.2　パス検証テストの実験項目」に示す各実験項目について、検証局を用いて実験を実施する。また、実験にはパス検証テストツールを用いる。パス検証テストツールは、パス検証テストに必要な証明書、リポジトリ環境等を提供するツールである。実験環境を「図 11.1　パス検証テストの実験環境」に示す。

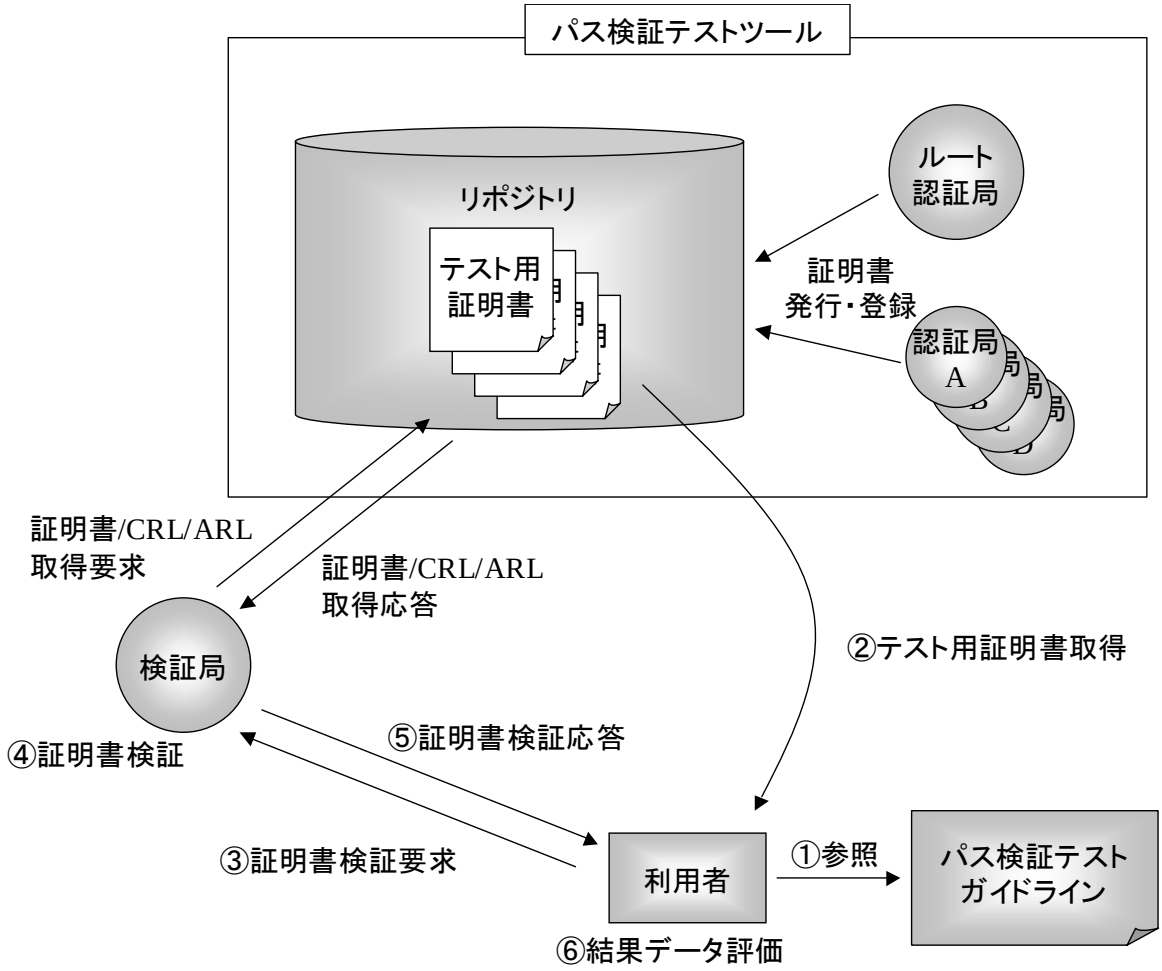


図 11.1　パス検証テストの実験環境

テスト手順を「表 11.3　パス検証テスト手順一覧」に示す。なお、「図 11.1　パス検証テストの実験環境」に記載している番号①～⑥は、「表 11.3　パス検証テスト手順一覧」の作業内容欄に記載している番号①～⑥と対応している。

表 11.3　パス検証テスト手順一覧

No.	作業内容
1	①利用者はパス検証テストガイドラインを参照して実験項目を確認する。
2	②パス検証テストツールのリポジトリへアクセスして実験項目に必要な証明書を取得する。
3	③利用者は検証局に対して証明書検証要求を送信する。
4	④検証局は証明書を検証する。検証する際には、パス検証テストツールのリポジトリから必要な証明書/CRL/ARL を取得する。
5	⑤検証局は証明書検証結果を利用者へ返信する。
6	⑥利用者は検証結果を評価する。

実験にあたっては、利用者端末及び日本国検証局の設定が、「表 9.2　証明書発行テストにおける証明書検証モジュールへのインプット情報」を満たす設定とな

っていることを確認する。

表 11.4　パス検証テストにおける証明書検証モジュールへのインプット情報

No.	インプット情報	設定
1	受け入れ可能なポリシ情報 (user-initial-policy-set)	設定しない
2	トラストアンカ情報 (trust anchor information)	テスト用ルート認証局の自己署名証明書
3	ポリシマッピングを許可するか否かの情報 (initial-policy-mapping-inhibit)	ポリシマッピングを許可
4	証明書パス中にユーザが受け入れ可能なポリシが入っていることを要求するか否かの情報 (initial-explicit-policy)	要求しない
5	any-policy を許可するか否かの情報 (initial-any-policy-inhibit)	許可する

パス検証テストの結果期待値を「表 11.5　パス検証テストの結果期待値」に示す。

表 11.5　パス検証テストの結果期待値

No.	実験項目		期待値		備考
	中項目	小項目	結果	Return Code	
1	Interconnection Model - Strict Hierarchy	Int.SH.DN.RP.01.01	Valid	0	「Return Code=０」は証明書が有効であることを示す。
2		Int.SH.DN.RP.02.01	Valid	0	－
3		Int.SH.DN.RP.03.01	Valid	0	－
4		Int.SH.DN.RP.04.01	Valid	0	－
5		Int.SH.DN.RP.05.01	Invalid	101	「Return Code=101」は証明書パスが存在しないことを示す。
6		Int.SH.DN.RP.06.01	Invalid	101	－
7		Int.SH.DN.RP.07.01	Valid	0	－

8		Int.SH.LDAPURI.RP.01.01	Invalid	203	「Return Code=203」は証明書パスを構築する証明書が失効していることを示す。
9		Int.SH.LDAPURI.RP.01.02	Invalid	203	－
10		Int.SH.LDAPURI.RP.02.01	Invalid	203	－
11		Int.SH.LDAPURI.RP.03.01	Invalid	203	－
12		Int.SH.LDAPURI.RP.04.01	Invalid	203	－
13		Int.SH.LDAPURI.RP.05.01	Invalid	203	－
14		Int.SH.LDAPURI.RP.05.02	Invalid	203	－
15		Int.SH.LDAPURI.RP.05.03	Invalid	203	－
16		Int.SH.LDAPURI.RP.06.01	Invalid	203	－
17		Int.SH.CJK.RP.01.01	Valid	0	－
18		Int.SH.CJK.RP.01.02	Valid	0	－
19		Int.SH.CJK.RP.01.03	Valid	0	－
20		Int.SH.CJK.RP.02.01	Valid	0	－
21		Int.SH.CJK.RP.02.02	Valid	0	－
22		Int.SH.CJK.RP.02.03	Valid	0	－
23		Int.SH.CJK.RP.03.01	Valid	0	－
24		Int.SH.CJK.RP.03.02	Valid	0	－
25		Int.SH.CJK.RP.03.03	Valid	0	－
26		Int.SH.CJK.RP.04.01	Valid	0	－
27		Int.SH.CJK.RP.04.02	Valid	0	－
28		Int.SH.CJK.RP.04.03	Valid	0	－
29		Int.SH.CJK.RP.05.01	Valid	0	－
30		Int.SH.CJK.RP.05.02	Valid	0	－
31		Int.SH.CJK.RP.05.03	Valid	0	－
32		Int.SH.CJK.RP.06.01	Valid	0	－
33		Int.SH.CJK.RP.06.02	Valid	0	－
34		Int.SH.CJK.RP.06.03	Valid	0	－
35		Int.SH.CJK.RP.07.01	Valid	0	－
36		Int.SH.CJK.RP.07.02	Valid	0	－
37		Int.SH.CJK.RP.07.03	Valid	0	－
38		Int.SH.CJK.RP.08.01	Valid	0	－
39		Int.SH.CJK.RP.08.02	Valid	0	－
40		Int.SH.CJK.RP.08.03	Valid	0	－

(4) 実験結果

実験結果を、「表 11.6　パス検証テストにおけるパス検証テストの実験結果」に示す。

表 11.6　パス検証テストにおけるパス検証テストの実験結果

No.	実験項目(小項目)	結果期待値		検証結果		期待値比較
		結果	Return Code	結果	Return Code	
1	Int.SH.DN.RP.01.01	Valid	0	Valid	0	一致
2	Int.SH.DN.RP.02.01	Valid	0	Valid	0	一致
3	Int.SH.DN.RP.03.01	Valid	0	Valid	0	一致
4	Int.SH.DN.RP.04.01	Valid	0	Valid	0	一致
5	Int.SH.DN.RP.05.01	Invalid	101	Invalid	101	一致
6	Int.SH.DN.RP.06.01	Invalid	101	Invalid	101	一致
7	Int.SH.DN.RP.07.01	Valid	0	Valid	0	一致
8	Int.SH.LDAPURI.RP.01.01	Invalid	203	Invalid	203	一致
9	Int.SH.LDAPURI.RP.01.02	Invalid	203	Invalid	203	一致
10	Int.SH.LDAPURI.RP.02.01	Invalid	203	Invalid	203	一致
11	Int.SH.LDAPURI.RP.03.01	Invalid	203	Invalid	203	一致
12	Int.SH.LDAPURI.RP.04.01	Invalid	203	Invalid	203	一致
13	Int.SH.LDAPURI.RP.05.01	Invalid	203	Invalid	203	一致
14	Int.SH.LDAPURI.RP.05.02	Invalid	203	Invalid	203	一致
15	Int.SH.LDAPURI.RP.05.03	Invalid	203	Invalid	203	一致
16	Int.SH.LDAPURI.RP.06.01	Invalid	203	Invalid	203	一致
17	Int.SH.CJK.RP.01.01	Valid	0	Valid	0	一致
18	Int.SH.CJK.RP.01.02	Valid	0	Valid	0	一致
19	Int.SH.CJK.RP.01.03	Valid	0	Valid	0	一致
20	Int.SH.CJK.RP.02.01	Valid	0	Valid	0	一致
21	Int.SH.CJK.RP.02.02	Valid	0	Valid	0	一致
22	Int.SH.CJK.RP.02.03	Valid	0	Valid	0	一致
23	Int.SH.CJK.RP.03.01	Valid	0	Valid	0	一致
24	Int.SH.CJK.RP.03.02	Valid	0	Valid	0	一致
25	Int.SH.CJK.RP.03.03	Valid	0	Valid	0	一致
26	Int.SH.CJK.RP.04.01	Valid	0	Valid	0	一致
27	Int.SH.CJK.RP.04.02	Valid	0	Valid	0	一致
28	Int.SH.CJK.RP.04.03	Valid	0	Valid	0	一致
29	Int.SH.CJK.RP.05.01	Valid	0	Valid	0	一致
30	Int.SH.CJK.RP.05.02	Valid	0	Valid	0	一致
31	Int.SH.CJK.RP.05.03	Valid	0	Valid	0	一致
32	Int.SH.CJK.RP.06.01	Valid	0	Valid	0	一致
33	Int.SH.CJK.RP.06.02	Valid	0	Valid	0	一致
34	Int.SH.CJK.RP.06.03	Valid	0	Valid	0	一致
35	Int.SH.CJK.RP.07.01	Valid	0	Valid	0	一致

36	Int.SH.CJK.RP.07.02	Valid	0	Valid	0	一致
37	Int.SH.CJK.RP.07.03	Valid	0	Valid	0	一致
38	Int.SH.CJK.RP.08.01	Valid	0	Valid	0	一致
39	Int.SH.CJK.RP.08.02	Valid	0	Valid	0	一致
40	Int.SH.CJK.RP.08.03	Valid	0	Valid	0	一致

⑸ 評価及び考察

評価を「(a)実験データに対する評価」、「(b)実験方法に対する評価」及び「(c)実験結果に対する評価」、考察を「(d)考察」に示す。

(a) 実験データに対する評価

本実験では、パス検証テストツールを用いて実験データを作成した。日本国検証局のログ情報による検証内容データ及び利用者端末のログ情報を評価した結果、実験データがパス検証テストガイドラインのテスト要求を満たしていることを確認した。

(b) 実験方法に対する評価

本実験では、日本国検証局のログ情報及び利用者端末のログ情報を収集した。日本国検証局のログ情報及び利用者端末のログ情報を確認することで、実験結果の有効性が検証できることを確認した。

(c) 実験結果に対する評価

(a)及び(b)で示す実験データ及び実験方法で得られた実験結果を用いて、パス検証テストガイドラインの有効性を評価できることを確認した。

(d) 考察

全てのテスト項目において、パス検証テストガイドラインのテスト期待値と検証結果が一致した。このことから、パス検証テストガイドラインが定義するテスト項目とテスト期待値の有効性が確認できた。

なお、ヒアリング結果を含めたパス検証テストガイドラインの評価・考察については、「13検証結果」及び「14全体考察（まとめ)」で述べる。

12 アプリケーションインターフェースの実証実験

12.1　実験環境

サーバ及びクライアントの環境を「図 12.1　アプリケーション実験環境」に示す。

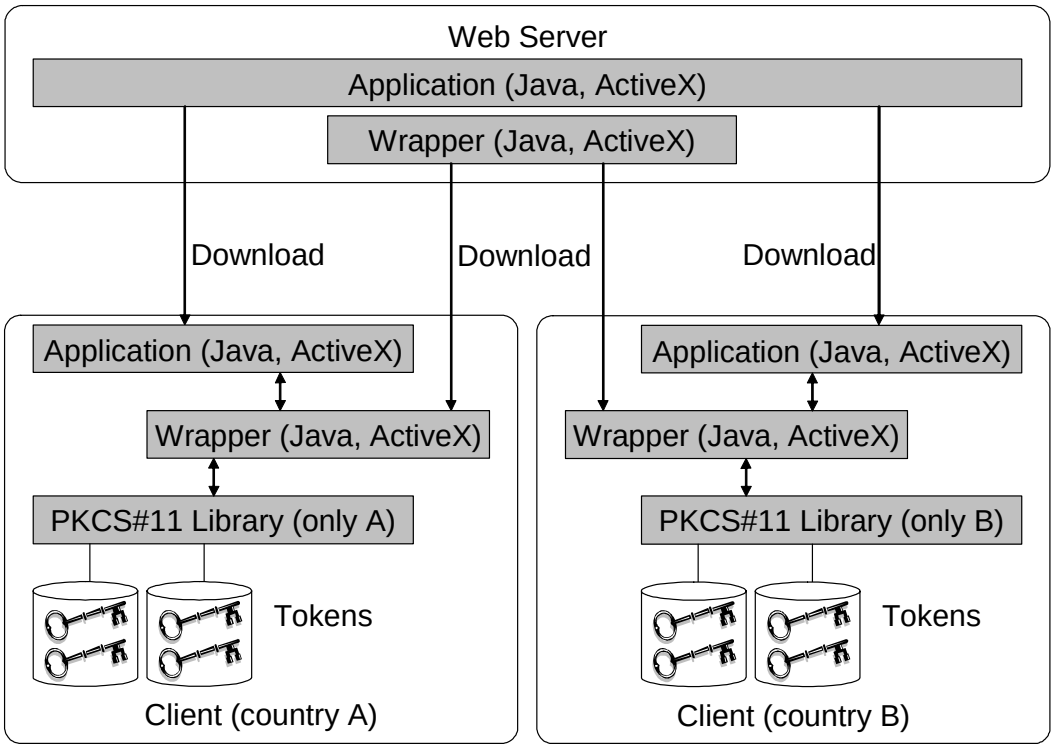


図 12.1　アプリケーション実験環境

本環境では、Web Server 上のアプリケーションとセットで Wrapper と呼ばれるモジュール（Java Applet or ActiveX）が置かれ、クライアントからの要求に応じてクライアントへダウンロードされる。

クライアントにおいては、PKCS#11 ライブラリが予めインストールされており、その PKCS#11 ライブラリが管理するトークンに秘密鍵と証明書が格納されている。ここで、トークン及びトークンに格納される秘密鍵と証明書は複数存在することが可能である。クライアントは Web ブラウザを使用して Web Server 上のアプリケーションへアクセスし、必要に応じて Wrapper をダウンロードし、Wrapper を介してアプリケーションと PKCS#11 ライブラリが連携する。

Web Server がアプリケーションサービスを提供する。クライアントにおいては、Web Server へアクセスしている状態で、PKCS#11 ライブラリの機能を呼び出す際に Web Server から Wrapper をダウンロードし、PKCS#11 ライブラリの処理結果をアプリケーションへ渡す。Wrapper とは、各国／地域の機能的な差異及び PKCS#11 ライブラリとアプリケーションの開発言語の差異を吸収するための機能として導入している。

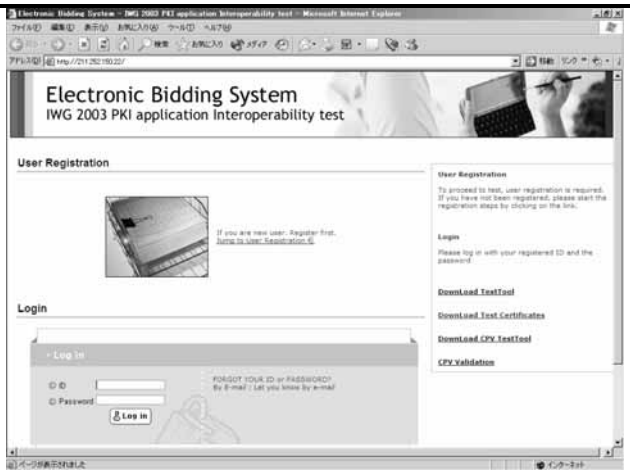
PKCS#11 ライブラリとトークンは各国／地域が利用可能なものを使用するが、

Wrapper はアプリケーションサービス提供者が共通なモジュールとして提供する。

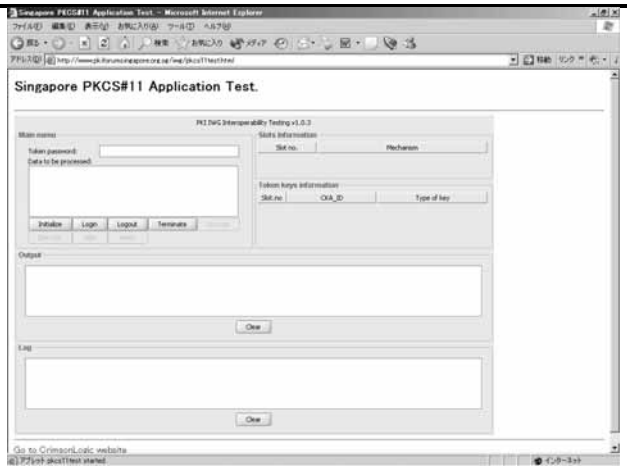
12.2　実験参加国／地域のアプリケーション画面イメージ

各国／地域のアプリケーションについて概要を「表 12.1　各国／地域アプリケーション概要」に挙げる。

表 12.1　各国／地域アプリケーション概要

日本		アプリタイプ　動作確認用テストツール 開発言語　Java1.3.1 備考　全ての機能を 1 画面に納めている。
韓国		アプリタイプ　e-bidding システム 開発言語　ActiveX 備考　最初に利用者登録が必要であり、利用者情報をサーバへアップロードする際に暗号化を行う実践的なものとなっている。最初に利用者登録が必要であり、利用者情報をサーバへアップロードする際に暗号化を行う実践的なものとなっている。

シンガポール

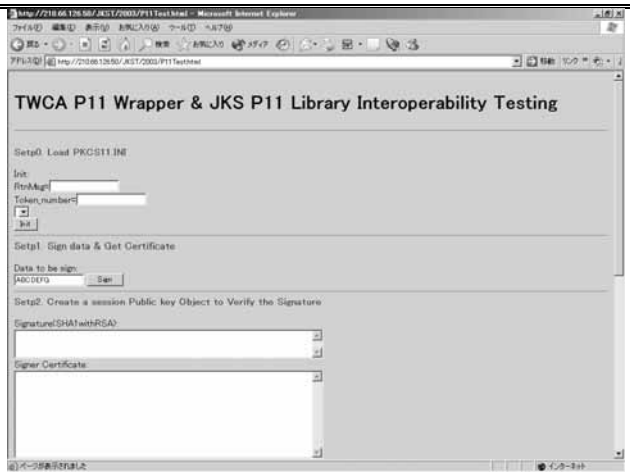


アプリタイプ　動作確認用テストツール

開発言語　Java1.4.1

備考　日本と似たイメージのアプリケーションで検証する機能を 1 画面に納めたものとなっている。

チャイニーズ台北



アプリタイプ　動作確認用テストツール

開発言語　ActiveX

備考　実験で使用するテストシナリオに沿って進めることが可能な画面レイアウト。Step2 の下に Step3：暗号化の機能が配置されている。

12.3　複数鍵／スロット処理検証

(1) 目的

本実験単位は、クライアント環境に複数存在するスロット、又は 1 つのスロットに複数存在する利用者鍵をアプリケーションが正しく認識し、利用者が意図した鍵を正しく取得することを確認することで、アプリケーションインターフェース仕様の有効性を検証することを目的とする。

(2) 実験内容

各国／地域の複数の正常なエンドエンティティ証明書を格納した利用者端末を使用し、日本、韓国、シンガポール、チャイニーズ台北が構築するアプリケーションへ相互にアクセスし、アプリケーションが複数鍵／スロットの環境を正しく認識し、全ての鍵を利用可能であることを確認する。

本実験で実施する項目を「表 12.2　正常系署名付与･検証の実験項目」に示す。

表 12.2　正常系署名付与･検証の実験項目

ID	検証者	検証対象	確認内容
JP-(1)	日本国	日本アプリケーション	・複数鍵／スロットの確認

JP-(2)	韓国	日本アプリケーション	・複数鍵／スロットの確認
JP-(3)	シンガポール	日本アプリケーション	・複数鍵／スロットの確認
JP-(4)	チャイニーズ台北	日本アプリケーション	・複数鍵／スロットの確認
KR-(1)	日本国	韓国アプリケーション	・複数鍵／スロットの確認
SG-(1)	日本国	シンガポールアプリケーション	・複数鍵／スロットの確認
CT-(1)	日本国	チャイニーズ台北アプリケーション	・複数鍵／スロットの確認

(3) 実験結果

本実験における検証結果を「表 12.3　複数鍵／スロットにおける検証結果」に示す。

表 12.3　複数鍵／スロットにおける検証結果

ID	検証者	検証対象	結果		判定
			期待値	検証結果	
JP-(1)	日本国	日本アプリケーション	○	○	○
JP-(2)	韓国	日本アプリケーション	○	○	○
JP-(3)	シンガポール	日本アプリケーション	○	○	○
JP-(4)	チャイニーズ台北	日本アプリケーション	○	○	○
KR-(1)	日本国	韓国アプリケーション	○	×	×
SG-(1)	日本国	シンガポールアプリケーション	○	○	○
CT-(1)	日本国	チャイニーズ台北アプリケーション	○	○	○

KR-(1)において検証結果が「×」となっている。本件は、韓国アプリケーションが、トークンに格納するオブジェクトを作成する際に設定する属性値テンプレートについて今回策定した仕様以外の属性値（CKA_MODULES_BITS）を設定していたことに起因する。今回策定した仕様では、相互運用を行うために必要となる属性値をすべて盛り込んでおり、仕様に無い属性値を設定した場合は、相互運用性は保証されない。従って、韓国側へアプリケーションの変更を依頼した。

(4) 評価・考察

本実験において、複数鍵／スロットの検証について、KR-(1)の実験項目以外で

期待どおりの結果を得ることができた。KR-(1)については、韓国アプリケーションが今回策定した仕様に無い機能を使用したことが原因であり、韓国アプリケーションを今回策定した仕様に則ったものへ修正した上で、改めて確認作業を実施する必要がある。

以上から、共通のアプリケーションを各国／地域の異なる PKCS#11 ライブラリを用いた複数鍵／スロットの動作について相互運用性を確保するための条件を確認することができた。また、アプリケーションインターフェース仕様の複数鍵／スロット処理に関する有効性を検証することができた。

エラーが発生しているテスト ID については、「14.2.3アプリケーションインターフェース仕様」で考察する。

12.4　暗号化・鍵保護処理検証

(1) 目的

本実験単位は、データの暗号化及び復号、その際に使用した公開鍵を共通鍵によりラップ・アンラップする。これにより、暗号化を安全に処理できることを確認し、アプリケーションインターフェース仕様の有効性を検証することを目的とする。

(2) 実験内容

各国／地域の複数の正常なエンドエンティティ証明書を格納した利用者端末を使用し、日本、韓国、シンガポール、チャイニーズ台北が構築するアプリケーションへ相互にアクセスし、暗号化・復号及び鍵の保護としてラッピング・アンラッピングについて期待どおりの結果を得られることを確認する。

本実験で実施する項目を「表 12.4　暗号化・鍵保護の実験項目」に示す。

表 12.4　暗号化・鍵保護の実験項目

ID	検証者	検証対象	確認内容
JP-(1)	日本国	日本アプリケーション	・暗号化・鍵保護の確認
JP-(2)	韓国	日本アプリケーション	・暗号化・鍵保護の確認
JP-(3)	シンガポール	日本アプリケーション	・暗号化・鍵保護の確認
JP-(4)	チャイニーズ台北	日本アプリケーション	・暗号化・鍵保護の確認
KR-(1)	日本国	韓国アプリケーション	・暗号化・鍵保護の確認
SG-(1)	日本国	シンガポールアプリケーション	・暗号化・鍵保護の確認
CT-(1)	日本国	チャイニーズ台北アプリケーション	・暗号化・鍵保護の確認

(3) 実験結果

本実験における検証結果を「表 12.5　暗号化／鍵保護における検証結果」に示

す。

表 12.5　暗号化／鍵保護における検証結果

ID	検証者	検証対象	結果		判定
			期待値	検証結果	
JP-(1)	日本国	日本アプリケーション	○	○	○
JP-(2)	韓国	日本アプリケーション	○	○	○
JP-(3)	シンガポール	日本アプリケーション	○	○	○
JP-(4)	チャイニーズ台北	日本アプリケーション	○	○	○
KR-(1)	日本国	韓国アプリケーション	○	×	×
SG-(1)	日本国	シンガポールアプリケーション	○	○	○
CT-(1)	日本国	チャイニーズ台北アプリケーション	○	○	○

KR-(1)において検証結果が「×」となっている。本件は、韓国アプリケーションが、トークンに格納するオブジェクトを作成する際に設定する属性値テンプレートについて今回策定した仕様以外の属性値（CKA_MODULES_BITS）を設定していたことに起因する。今回策定した仕様では、相互運用を行うために必要となる属性値をすべて盛り込んでおり、仕様に無い属性値を設定した場合は、相互運用性は保証されない。従って、韓国側へアプリケーションの変更を依頼した。

⑷ 評価・考察

本実験において、暗号化・鍵保護処理の検証について、KR-(1)の実験項目以外で期待どおりの結果を得ることができた。KR-(1)については、韓国アプリケーションが今回策定した仕様に無い機能を使用したことが原因であり、韓国アプリケーションを今回策定した仕様に則ったものへ修正した上で、改めて確認作業を実施する必要がある。

以上から、共通のアプリケーションを各国／地域の異なる PKCS#11 ライブラリを用いて暗号化・鍵保護処理について相互運用性を確保するための条件を確認することができた。また、アプリケーションインターフェース仕様の暗号化・鍵保護に関する有効性を検証することができた。

エラーが発生したテスト ID については、「14.2.3アプリケーションインターフェース仕様」にて考察する。

12.5　署名付与・検証処理検証（正常系）

(1) 目的

本実験単位は、平成 14 年度に策定した署名用トークンインターフェース仕様における署名付与・検証機能にハッシュ値作成を盛り込むことで本来あるべき電子署名の機能を実現し、電子署名の正常な動作を確認することで、アプリケーションインターフェース仕様の有効性を検証することを目的とする。

(2) 実験内容

各国／地域の複数の正常なエンドエンティティ証明書を格納した利用者端末を使用し、日本、韓国、シンガポール、チャイニーズ台北が構築するアプリケーションへ相互にアクセスし、署名付与・検証について期待どおりの結果を得られることを確認する。

本実験で実施する項目を「表 12.6　署名付与・検証の実験項目」に示す。

表 12.6　署名付与・検証の実験項目

ID	検証者	検証対象	確認内容
JP-(1)	日本国	日本アプリケーション	・署名付与・検証の確認
JP-(2)	韓国	日本アプリケーション	・署名付与・検証の確認
JP-(3)	シンガポール	日本アプリケーション	・署名付与・検証の確認
JP-(4)	チャイニーズ台北	日本アプリケーション	・署名付与・検証の確認
KR-(1)	日本国	韓国アプリケーション	・署名付与・検証の確認
SG-(1)	日本国	シンガポール アプリケーション	・署名付与・検証の確認
CT-(1)	日本国	チャイニーズ台北 アプリケーション	・署名付与・検証の確認

(3) 実験結果

本実験における検証結果を「表 12.7　署名付与・検証における検証結果」に示す。

表 12.7　署名付与・検証における検証結果

ID	検証者	検証対象	結果		判定
			期待値	検証結果	
JP-(1)	日本国	日本アプリケーション	○	○	○
JP-(2)	韓国	日本アプリケーション	○	○	○
JP-(3)	シンガポール	日本アプリケーション	○	○	○
JP-(4)	チャイニーズ台北	日本アプリケーション	○	○	○

KR-(1)	日本国	韓国アプリケーション	○	×	×
SG-(1)	日本国	シンガポールアプリケーション	○	○	○
CT-(1)	日本国	チャイニーズ台北アプリケーション	○	○	○

KR-(1)において検証結果が「×」となっている。本件は、韓国アプリケーションが、トークンに格納するオブジェクトを作成する際に設定する属性値テンプレートについて今回策定した仕様以外の属性値（CKA_MODULES_BITS）を設定していたことに起因する。今回策定した仕様では、相互運用を行うために必要となる属性値をすべて盛り込んでおり、仕様に無い属性値を設定した場合は、相互運用性は保証されない。従って、韓国側へアプリケーションの変更を依頼した。

⑷ 評価・考察

本実験において、署名付与・検証の検証について、KR-(1)の実験項目以外で期待どおりの結果を得ることができた。KR-(1)については、韓国アプリケーションが今回策定した仕様に無い機能を使用したことが原因であり、韓国アプリケーションを今回策定した仕様に則ったものへ修正した上で、改めて確認作業を実施する必要がある。

以上から、共通のアプリケーションを各国／地域の異なる PKCS#11 ライブラリを用いて署名の付与・検証について相互運用性を確保するための条件を確認することができた。また、アプリケーションインターフェース仕様の署名付与・検証に関する有効性を検証することができた。

発生したエラーについては、「14.2.3アプリケーションインターフェース仕様」にて考察する。

12.6　署名付与・検証処理検証（異常系）

⑴ 目的

本実験単位は、平成 14 年度に策定した署名用トークンインターフェース仕様における署名付与・検証機能にハッシュ値作成を盛り込むことで本来あるべき電子署名の機能を実現し、電子署名の改ざん検出に関する動作を確認することで、アプリケーションインターフェース仕様の有効性を検証することを目的とする。

⑵ 実験内容

各国／地域の複数の正常なエンドエンティティ証明書を格納した利用者端末を使用し、日本、韓国、シンガポール、チャイニーズ台北が構築するアプリケーシ

ョンへ相互にアクセスし、署名付与・検証について改ざんを検出し期待どおりの結果を得られることを確認する。

本実験で実施する項目を「表 12.8　署名付与・検証の実験項目（改ざん検出）」に示す。

表 12.8　署名付与・検証の実験項目（改ざん検出）

ID	検証者	検証対象	確認内容
JP-(1)	日本国	日本アプリケーション	・署名付与・検証の確認
JP-(2)	韓国	日本アプリケーション	・署名付与・検証の確認
JP-(3)	シンガポール	日本アプリケーション	・署名付与・検証の確認
JP-(4)	チャイニーズ台北	日本アプリケーション	・署名付与・検証の確認
KR-(1)	日本国	韓国アプリケーション	・署名付与・検証の確認
SG-(1)	日本国	シンガポールアプリケーション	・署名付与・検証の確認
CT-(1)	日本国	チャイニーズ台北アプリケーション	・署名付与・検証の確認

(3) 実験結果

本実験における検証結果を「表 12.9　署名付与・検証における検証結果」に示す。

表 12.9　署名付与・検証における検証結果（改ざん検出）

ID	検証者	検証対象	結果		判定
			期待値	検証結果	
JP-(1)	日本国	日本アプリケーション	×	×	○
JP-(2)	韓国	日本アプリケーション	×	○	○
JP-(3)	シンガポール	日本アプリケーション	×	×	○
JP-(4)	チャイニーズ台北	日本アプリケーション	×	×	○
KR-(1)	日本国	韓国アプリケーション	×	×	×
SG-(1)	日本国	シンガポールアプリケーション	×	×	○
CT-(1)	日本国	チャイニーズ台北アプリケーション	×	×	○

KR-(1)において検証結果が「×」となっている。本件は、韓国アプリケーションが、トークンに格納するオブジェクトを作成する際に設定する属性値テンプレ

ートについて今回策定した仕様以外の属性値（CKA_MODULES_BITS）を設定していたことに起因する。今回策定した仕様では、相互運用を行うために必要となる属性値をすべて盛り込んでおり、仕様に無い属性値を設定した場合は、相互運用性は保証されない。従って、韓国側へアプリケーションの変更を依頼した。

(4) 評価・考察

本実験において、署名値の改竄の検証について、KR-(1)の実験項目以外で期待どおりの結果を得ることができた。KR-(1)については、韓国アプリケーションが今回策定した仕様に無い機能を使用したことが原因であり、韓国アプリケーションを今回策定した仕様に則ったものへ修正した上で、改めて確認作業を実施する必要がある。

以上から、共通のアプリケーションを各国／地域の異なる PKCS#11 ライブラリを用いて署名値の改竄の検出について相互運用性を確保する条件を確認することができた。また、アプリケーションインターフェース仕様の改竄検出に関する有効性を検証することができた。

13 検証結果

13.1　認証局間相互接続及びそのテストに関する標準の有効性の検証

13.1.1 認証局間相互接続テストガイドラインの有効性の検証

本項目では、「9現地タイ認証局との認証局間相互接続テストガイドラインの実証実験」及び「10仮想対象国認証局との認証局間相互接続テストガイドラインの実証実験」として検証した検証結果とヒアリングの結果を分析する。

検証作業を通じて、「認証局間相互接続テストガイドライン」に従ってテストを実施することで、利用者は目的とする認証局間の相互接続についての確認を正しく行うことができることが明らかになった。「認証局間相互接続テストガイドライン」の定める仕様とテスト手順に基づいて構築した認証局間の相互接続関係の中では、双方の認証ドメインでエンドエンティティの検証が正しく機能することが確認され、PKI の利用環境として正しく相互接続関係を構築することができたと判断される。また利用者であるテスト主体の認証局が不正な証明書を発行した場合や、データ環境に不備がある状態に陥った場合には、テストによって正しく検出された。これらのことから「認証局間相互接続テストガイドライン」を使用して相互接続関係の構築を検証することは､利用者にとって有効であることが確認されたものと判断する。

他の検証データとして、「認証局間相互接続テストガイドライン」について本実証実験の実施者である現地タイ認証局及び日本国認証局の利用者と、昨年度までに認証局間の相互接続実証実験に参加した JKSTIWG メンバからアンケートまたはヒアリングによる評価データを採取した。以下に構成の妥当性、内容の網羅性及び記述の明確性の点からそれら評価データを分析する。

まず、構成の妥当性については、問題の指摘はなかった。タイから要望として、「認証局間相互接続テストガイドライン」に含まれると便利な内容として、掲示板など利用者が対話的に情報を得られる仕組みが Web サイトで提供されると有益である、トラブルシューティング、もしくは FAQ か Q&A があると良い、サンプル証明書や DIT が Web サイトからダウンロードできると便利、といった提言がなされた。どれも本実証実験でタイが自らの体験として必要を感じた点と思われ、利用者からの有益な提言である。提言を「認証局間相互接続テストガイドライン」に取り込むことは今後の拡張として行いたい。

次に内容の網羅性については、不足している内容として韓国からテスト対象として認証局の鍵更新が含まれるべきであるとの指摘を受けた。確かに鍵更新は認証局の鍵・証明書管理という業務の上で重要なフェーズであり「認証局間相互接続テストガイドライン」に含まれるのが適切な内容である。しかし鍵更新についてはこれまでの実証実験で対象としておらず、現時点では実証実験の成果として利用者に提供できるノウハウを持たない。よって指摘に従って「認証局間相互接続テストガイドライン」に鍵更新についての枠組みを追加したが、内容については将来課題としている。この対応によって認証局の鍵・証明書管理について主要なフェーズが網羅され、利用者に提示されることになった。

記述の明確性については､チャイニーズ台北より､「認証局間相互接続テストガイドライン」が定めるのはテストの手順であり実運用とは切り分ける必要がある箇所はその旨明確にすべきとの指摘を受けた。具体的には、エンドエンティティ証明書と鍵を認証局で生成して配布する手段として、PKCS#12 フォーマットを安全性が保証されない経路で配布するのはテスト時のみに取りうる手段であり、実運用はより厳密でなければならないことを追記すべきとの内容であった｡妥当であり､かつ、本ガイドラインをより有益なものとする指摘と判断されるため、指摘に従って「認証局間相互接続テストガイドライン」への追記を行った。

以上のような指摘はどれも「認証局間相互接続テストガイドライン」をより良くするための積極的な提言であり、各国／地域が「認証局間相互接続テストガイドライン」の普及に向けて真摯に検討した結果である。これらの指摘を受け、それを反映することで､「認証局間相互接続テストガイドライン」はより有効なものとなったと考える。

総合的な評価として、本年度の実証実験の実施主体であるタイからは「認証局間相互接続テストガイドライン」は作業を進める上で非常に役に立ったと評価されている。

以上､検証結果とヒアリング結果の両面から､「認証局間相互接続テストガイドライン」は十分な有効性を持つものと判断する。

13.2　証明書検証に関する標準の有効性の検証

13.2.1 パス検証テストガイドラインの有効性の検証

本項目では､「11パス検証テストガイドラインの実証実験」として検証した検証結果とアンケート結果を分析する。実証実験では、全てのテスト項目において、テスト期待値と検証結果が一致し、テストデータおよびパス検証テストガイドラインのテスト期待値の的確性に高い評価が得られた。

以下に、実験参加国／地域におけるソフトウェア開発に携わった経験のある者、又は PKI 技術者へヒアリング及びアンケートを実施し､収集した評価データを分析することにより、パス検証テストガイドラインの有効性検証の評価を行う。

平成 15 年度成果により追加された DN マッチング、LDAP URI および UTF8 CJK 文字セットのテストケースの有用性に関する問いについては､これら３つのテストケースは全て有益であるとの総意を得た。

DN マッチングに関しては特に PrintableString の空白文字や大文字小文字の正規化処理の検証を幅広く行うものでありパス検証の開発者並びに利用者にとって有用なテストケースとなっている｡LDAP URI テストケースに関しては､LDAP URI のエンコーディング方式について網羅的に検証可能なテスト項目であり、米国 NIST に代表されるテストケースにも含まれない特徴的なテストケースとなっている。

UTF8 CJK テストケースは、RFC3280 にて述べられている IETF PKIX-WG の 2003 年 12 月 31 日以降の PrintableString から UTF8String への移行勧告と大いに関係のあるテスト項目である。韓国、チャイニーズ台北、日本でディレクトリ名として使用される可能性のある 7 つの文字区画に関するテストケースをディレクトリ名および 2 つのエスケープ方式の LDAP URI に対して設計し、アジア圏の特色を示す無比のテストケースとなっている。

UTF8 CJK に関してはパス検証の要件だけでなく認証局の発行要件として、パス検証ガイドライン外で規定されるガイドラインが必要ではないかとの意見も得られた。

平成 15 年度追加の３つのテストケースに対するテスト項目の網羅性に関する問いについては、IWG プロファイルに基づいており、そのプロファイルを網羅する必要十分なテスト項目となっているとの評価を得た。

DN matching、LDAP URI および UTF8 CJK のテストケースでは全て cRLDistributionPoints と issuingDistributionPoint のマッチングが含まれている。この両者がそれぞれ複数あった場合に関するセキュリティ考察については、さらなる検討が必要であるとの意見があった。(参考 : 10.2.1 (4) CRL 配布に関するセキュリティ考察)

また、DN matching、LDAP URI では空白文字の正規化を扱うテストケースが含まれているが、空白文字は SPACE(ASCII 0x20)だけでなくタブ文字などの非印字文字も含まれるのではないか。RFC3280 では空白文字の定義がなされておらず、IETF PKIX WG 等にて確認を取る必要があるのではないかという意見があった。

平成 15 年度までに設計された全テストケースを以って、IWG プロファイルの全てを網羅するテストケースが完成したと見なしてよいとの総意を得た。IWG プロファイル外ではあるが現在興味がある分野として HTTP URI、OCSP のテストケースが挙げられた。特にチャイニーズ台北からは政府認証基盤で HTTP URI が使われていることもあり、是非実験を行いたいとの要望があった。

実施したアンケートにより平成 14 年度作成されたパス検証ガイドライン全体を概観した意見として３つの意見が得られた。

- トラストモデルの分析としてモデルが網羅的に紹介されている事は好ましいがテストケースは IWG プロファイルに基づくべきであり、テストケースとして提供するトラストモデルの範囲を厳密に規定すべきである。
- 失効モデルとサービスモデルの分類についてトラストモデルと重複するテスト項目が多くトラストモデルに対するオプショナルテストとした方が理解しやすいのではないか。

・ テストレベルに関して現行の３つのレベルにおいて、どのレベルまで満足すべきであるかという指標が無いため理解し辛いものとなっている。テストレベルをプロファイルの必須／オプションの分類により分けているが、これは認証局側の発行要件であり、パス検証ガイドラインは署名検証者の立場で利用するものであり署名検証者側は必須であってもオプションであっても正しく処理できる必要がある。

以上のアンケート結果に基づき、テストケースで扱うモデルを整理し、テストレベルを必ず実施すべき標準テストと、利用する環境に基づきテスト実施を判断可能なオプションテストとの２つのレベルにするようパス検証ガイドラインの改訂を行った。

このように「11パス検証テストガイドラインの実証実験」とアンケートの結果からパス検証テストガイドラインの利用者にとっての有効性が明らかになった。また、実証実験結果及びアンケートの結果を踏まえ、パス検証テストガイドラインの改訂を行ったことで懸案を全て解消し、平成 15 年度を以ってパス検証ガイドラインは完成したと評価できる。

13.2.2 パス検証テストの効率化・作業負担軽減に関する検証

本項目では、「11パス検証テストガイドラインの実証実験」が「6開発作業」で開発されたツールを用いることにより効率的に行えるようになったか、作業負担が軽減されたか、この検証結果とアンケート結果を分析する。

パス検証テストにおいては以下のステップでテストツールを利用した。

1. 日本国検証局から発行された証明書発行要求の受理
2. 日本国検証局用の証明書の発行
3. DN matching, LDAP URI テストケースのテストデータの登録
4. UTF8 CJK テストケースの設計と登録
5. テストデータの生成
6. リポジトリへの登録
7. リポジトリの公開

上述ステップ 3～7 についてはオペレーションミスによるテストデータの入力の誤り、並びにテストデータの設計の誤りをテスト実施により確認することができたため 6 回の繰り返しがあったが、データ生成並びにリポジトリ登録はバッチ処理であり、またデータの一括変更もテストツールのデータベース内容を置換することで可能であるため大きな作業負担ではなかった。

以下に、実験参加国／地域におけるソフトウェア開発に携わった経験のある者、又は PKI 技術者へヒアリング及びアンケートを実施し、収集した評価データを分析することにより、「11パス検証テストガイドラインの実証実験」の実証実験が効率的に行えるようになったか検証を行う。

平成 14 年度実験では、オープンソースソフトウェアを使用してテストデータは生成していたが、テストデータによってはソースコード自体にその都度改変を加えなければならないものも多数あり、データ生成には大変負担があった。平成 15 年度、パス検証テストの実証実験のため開発されたテストツールでは、JKSTIWG で利用するあらゆる拡張領域を含む様々な属性を持つデータをウェブブラウザを用い容易かつ柔軟に発行できる。現在のところ、このような機能を持ったツールは見あたらないため他のツールとの比較は難しいが、前年度と比較して作業負担は十分の一になり作業時間の短縮に役立ったとの総意を得た。また、テストツールの証明書データコピー機能により、パス検証テストで頻繁に現れる一部の属性のみが異なるデータを非常に簡単に生成できる点が有益であるとの評価を得た。

テストケース設計時の効率化については、ツールを用いて設計を行ったチャイニース台北の設計者から、鍵ペア管理、証明書管理、証明書失効リスト管理およびリポジトリ管理より構成される統合された GUI を利用することにより、テストケースの設計時に関係するテストデータの一覧を見られることはテストケースを概観する上で非常に役立ったとの評価を得た。

実験実施の効率化については、テストデータである証明書や証明書失効リストの ASN.1 ダンプが簡単に見られるため、実験で問題が生じた際に、サーバ上のデータに問題があるのか、クライアント側のパス検証 API に問題があるのか切り分けが容易であったことも実験の効率化に大きな効果があったとの意見を得た。テストツール上では個々のテストデータに対し ID 番号が付与されているが、実験者が問題をトレースしバグの原因を調査する際に、さらには実験者間で問題点を共有する際に非常に有用であったとの評価を得た。

リポジトリへの登録に関しては平成 14 年度は、テストデータ生成と、リポジトリ定義ファイル(LDIF)の生成と、リポジトリについて別々のオペレータが行う必要があるため申し渡しに要するタイムラグがあり非効率であった。平成 15 年度はデータベースに登録されたテストデータに基づきバッチ処理を用いワンアクションでリポジトリに登録することができ、効率的にテスト用リポジトリを構築することができたとの評価を得た。

テストツールに対する改善提案としては、テストツールのデータ生成が自由度が高いために、テスト設計者のオペレーションミスにより間違ったデータを作ってし

まうことがあるため、そのためのデータ生成は可能としながらも登録時等に警告を出す機能が必要であろうとの事であった。その他、アンケートにより得られた改善提案を以下に示す。

- テストツールの基本設計の改変を必要としない機能
  1. LDAP URI のエンコードは入力に迷うためこれをサポートする機能
  2. LDAP エントリは LDIF ファイルのエントリ部分のテンプレートであり LDIF の文法を満たす満たさないに関わらず任意のテキストが入力可能であるが、何を入力すべきか判断に迷うためこれをサポートする GUI 機能
  3. シリアル番号等では大きな整数値が自由に 16 進数形式で入力可能であるが、RFC3280 準拠でない負の値への対策
- テストツールの基本設計の改変を必要とする機能
  4. authorityKeyIdentifyer の keyIdentifier の値、subjectKeyIdentifier の値を鍵ペアデータから自動的に設定される機能
  5. keyUsage、basicConstraints など自動的に設定する機能
  6. 証明書の ASN.1 構造として不正な証明書を発行する場合には警告する機能
  7. 検証に失敗するであろう証明書をチェックする機能

テストツールの基本設計に触れる改善提案については即座に対応することができないが、対処可能な項番 1、2 および 3 についてはツールの改善を行った。項番 4 については基本設計の変更が必要であるため対応できない。項番 5 および 6 についてはデータコピー機能を利用したり証明書ファイル生成機能を使うなど使用法を工夫することにより解決可能である。項番 7 については「6.2.1(1)パス検証テストデータローダー」で提供されるチェック機能を用いることで対応できる。

このようにアンケートの結果から「6開発作業」で開発されたツールを用いることにより「11パス検証テストガイドラインの実証実験」がより効率的に行え、作業負担が軽減されることが明らかとなった。また、アンケート結果を踏まえテストツールの改善を行ったことで、より作業効率を高めることができるようになったと評価できる。

13.3　アプリケーションインターフェースに関する標準の有効性の検証

13.3.1 アプリケーションインターフェース仕様の有効性の検証

本項目では、「12アプリケーションインターフェースの実証実験」における実験結果とヒアリングの結果を分析し、検証を行う。

今回の実験において、一部エラーが発生しているが、アプリケーションの運用要件に関するアプリケーション開発者の解釈の違いから発生しているものである。

この解釈の違いは、アプリケーションの運用設計において決められるべきもので

あり、今回の仕様の有効性には影響しないと判断することができる。

ヒアリングの結果からも、本標準はアプリケーション利用者及びアプリケーション開発者にとっての有効性が高い仕様であると評価することができる。以下に、韓国、シンガポール、チャイニーズ台北の PKI 技術者及びアプリケーション開発経験者へ行ったヒアリングの結果を示しながら検証を行う。

まず、必要機能の網羅性については、全員が網羅していると回答している。アプリケーション開発者にとっての使いやすさについても全員が使いやすいものであると回答している。機能の安全性・確実性についても全員が安全かつ確実なものであるとしている。このことから、PKI を活用するアプリケーションで必要となる機能のうち、PKCS#11 で実現可能な機能については網羅できており、有効であることが検証できたと言える。

この標準の有効性に関する問いにも、全員が有効であると回答している。ただし、更に見易さを向上させるため、必須項目とオプション項目をグループ化し、わかり易くする工夫が必要であるとの意見があった。この点は更に見易さに配慮した修正を行い、解釈に差が生じないよう工夫する必要がある。

実用性に関する問いには、利用者に対して、アプリケーションサービス提供者に対して、アプリケーション開発者に対しての 3 点でヒアリングを行った。全員が全てについて実用的であると回答しているが、以下の意見が出た。

- 実験参加国／地域以外に今回の仕様をいかに知ってもらい、いかに使ってもらうかが課題である。
- 今回の仕様は Windows に偏りすぎている。OS に関してオープンな仕様とすべきである。
- 署名検証で必須となるパス検証について、CryptoAPI 等が必要になるため、それらの要素も盛り込むべきである。

この 3 点に関して、1 点目は、アジア PKI ガイドラインへ掲載することで達成することが可能であり、今後アジア PKI ガイドラインへ掲載するための活動が必要である。2 点目は、PKCS#11 が OS に関してオープンな仕様であるが、今回の実験で Windows 用の PKCS#11 ライブラリを使用したことに起因していると思われる。Windows 以外の OS に対応した PKCS#11 ライブラリを用意することで実験の実施が可能であり、今回策定した仕様とは関係が無い。3 点目について、今回の仕様で実現した機能以外で必要となる機能に関する問いには、CryptoAPI や Java を使用したパス検証という意見があった。パス検証は署名の有効性を検証する際に必須の機能であり、今後の検討が必要である。

他に必要となる機能では、クライアント環境の正当性保証が挙げられた。この件は考察でも述べるが、アプリケーションに関するコード署名やサーバ証明書という考え方はこれまでに例があるものの、クライアント環境についてアプリケーション

が保証するという仕組みはこれまでに実例の無い考え方である。しかし、アプリケーションの環境設定情報、或いはアプリケーションが使用するトークン環境についてアプリケーションが保証していくという考え方は必要であると思われ、課題の抽出から時間をかけて議論することが必要になると思われる。

以上のことから、本標準は若干の改善点、PKCS#11 以外で必要となる機能の追加等の指摘があったものの、PKCS#11 に関する有効性、網羅性、実用性については十分に高い評価を得られたと分析することができ、本標準の有効性を高いレベルで検証できたと判断することができる。

14 全体考察（まとめ）

14.1　成果

14.1.1 認証局間相互接続及びそのテストに関する成果

(1) テストガイドライン作成による簡易で安全な認証局間相互接続テスト手法の確立

平成 14 年度までの実証実験では IWG プロファイル及び PKI コンポーネントのインターフェースは決められていたが、実証実験を進めるためのノウハウや実験参加国／地域で議論された内容は必ずしも整理・明文化されているわけではなかった。

そこで、今年度相互接続のノウハウを持たない国／地域を含んだ相互接続においても、安全でスムーズな関係構築を可能とするために、今年度これまでの認証局間相互接続実証実験を標準テストパターンとしてブラッシュアップし、「認証局間相互接続テストガイドライン」を作成した。

本ガイドラインは、相互接続時の事前確認項目、テストの具体的な作業手順及び結果確認の方法を明記したものであり、認証局間固有の運用及び使用しているアプリケーションに依存するテスト項目・手順等を除き、ドキュメントの可読性及び作業の効率的な流れに注意し設計したものである。

テストガイドライン構築の成果は主に以下の 3 点である。

(a) ノウハウのない利用者でも安全な相互接続を実現

これまでの実証実験で、異なるドメイン間の相互接続については実験参加国の国内で実績が無く、本実証実験で初めて実施するという場合が少なくなかった。そのような場合、実験参加国である CA 運用者及び証明書検証者は相互接続のノウハウを持たず、テスト手順自体が相互接続作業の具体的な作業手順そのものであった。ノウハウを持たない作業者にとっては、必要十分な作業手順やチェック項目を漏れなく設定すること自体が困難である。

ガイドラインの作成により、テスト手順に従えば相互接続関係がスムーズに構築でき、また結果の確認を恣意的でなく行うことが出来るようになり、安全な相互接続を実現できるようになったと言える。

(b) テスト作業の効率化によるテスト期間の短縮

これまでの実証実験のノウハウの蓄積から、(i)相互接続を行う場合に陥りやすいポイントやパターンを整理し、事前確認項目として明確化し、(ii)テスト環境の変更を極力少なくするテスト手順を構築することにより、テスト作業の効率化を図ることができテスト期間を短縮することが出来た。

テスト実施期間としては、トラブルシューティング、各認証局におけるセキュリティ確保のための運用ポリシの定める手順及びオペレーションの冗長性等

の時間を除けば、シミュレーションセンターの環境で行ったように最短1日で消化できることが検証できた。

(c) ガイドライン利用者の可読性の向上

コンテンツ提供形態を平成 14 年度までのプレーンなドキュメントスタイルからインタラクティブな Web インターフェースに移行し、関連する PKI ドキュメントを整理してハイパーリンクを行った。これによりガイドライン利用者の可読性を大きく向上することができた。

⑵ 実運用に向けた認証局間相互接続仕様の確立

これまでの韓国・シンガポール・チャイニーズ台北・香港チャイナ、及びタイの合計 5 カ国／地域と相互接続実証実験を行い、仕様としても平成 14 年度のものをそのまま適用することができた。

これはすなわち、本仕様が一定の完成度に達したことを示しており、他国のヒアリング結果においても高い評価を得ていることから、国際的に通用する相互接続仕様として確立できたといえる。

また、今年度は実証実験と実運用との相違点を注記することにより、実運用により適用しやすくなったと言える。

14.1.2 パス検証に関する成果

⑴ パス検証ガイドラインの確立

パス検証ガイドラインとは、認証局の設計担当者及びソフトウェア開発者がアプリケーションにおける証明書検証方法の設計作業を行う際に必要となる検証要件、評価作業におけるテスト項目を検討する際に使用する標準である。

基本的には X.509 や RFC3280 の標準において記述されているパス検証ロジックを対象としたテスト項目となっているが、JKSTIWG 証明書および証明書失効リストプロファイルに準拠していることを前提としたテスト内容となっている。

パス検証ガイドラインは平成 14 年度、第一版が公開されたが、その後、国内外の実証実験参加メンバより 3 つ指摘事項があった。相互接続モデルによってテストパターンを分類しているが、テストパターンに関しては現行の IWG プロファイルに必要な接続モデルのみにすべきであるという意見。1 つは失効モデル、サービスモデルという分類は有益であるが、現行のテストパターンは、重複しているものが多くテストが冗長であり整理が必要であるという意見。もう 1 つは、最後は、テストレベルにおいてどこまでを満足する必要があるのか、その規範が提示されていないために誤解を招きやすいという意見であった。

以上の課題を解決するためにパス検証ガイドラインで定義されるテストパター

ンの見直しを行い、失効モデル、サービスモデルについて、トラストモデルを中心とするようなテスト体系に変更した。また、テストレベルは標準テストとオプションテストの 2 種類のみとした。これにより、本ガイドラインの利用者が何をすべきか判断が容易になった。

パス検証ガイドラインはドキュメントのみならず、テストデータを証明書、証明書失効リストのファイルのみでなく、データベースのデータファイルとしても配布することが可能であり、既存の JKSTIWG メンバ国／地域や新規加入国がパス検証テストを行う場合、テスト環境の再現が容易で再利用性の高いデータとなっている。

このように本ガイドラインは、平成 15 年度の改訂を以って課題として残された問題をクリアしアプリケーションが相互接続環境において必要なパス検証機能を実装しているかどうかを評価するためのフレームワークとして利用することができる有効なガイドラインとすることができた。

(2) テストツールを用いたパス検証テストの効率化

平成 15 年度実験では、パス検証テストに用いる証明書や証明書失効リストの発行およびリポジトリの提供をテストツールを用いて行った。

パス検証実験に用いられる認証局やリポジトリ等のサーバ環境については Apache や OpenLDAP 等のオープンソースソフトウェアで構成される 1 台のテストツールサーバにより、複数のリポジトリを提供しており、パス検証実験に必要なリポジトリ環境等の構築にフリーウェアのみで構成されている。

テストパターン設計およびテストデータの生成においては、テストツールサーバ上のウェブサーバを介して、日本とチャイニーズ台北が協調作業によりテストパターンの設計やデータのデータベース投入を効率的に行えた。

テストツールを用いたテストデータの設計、テストデータとなる証明書、証明書失効リストおよびリポジトリ設定ファイル(LDIF)をバッチ処理により一括自動生成でき、効率的にテストデータおよびリポジトリを構築することができた。

また、平成 15 年度の成果であるテストデータが JKSTIWG におけるパス検証実験の再実験や新規参加国の実験に対して、実験環境の再構築が格段に容易になり、構成の変更、テストケースの変更にも柔軟に対応可能であり再利用性も 1 つの大きな成果となっている。

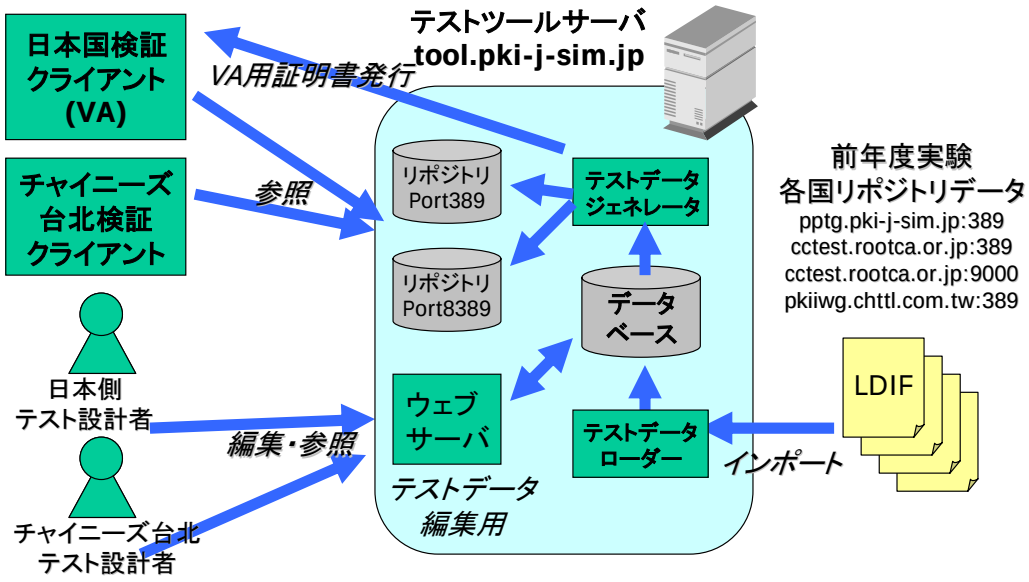


図 14.1　テストツールを用いたパス検証テスト実験環境

14.1.3 アプリケーションインターフェースに関する成果

(1) アプリケーションサービスの国際的利用における共通仕様の確立

昨今の PKI 推進活動は国の内外を問わず盛んであり、具体的な活動としてはアメリカにおける OASIS（旧 PKI フォーラム）、EU 圏における EESSI、国内における GPKI、LGPKI、CALS/EC、公的個人認証などが挙げられる。これらの状況において、PKI を活用する PKI アプリケーションの観点で見た場合、特定の仕様或いは特定の製品に依存した PKI アプリケーションとなることも多々見受けられる。このような場合、ある PKI アプリケーションサービスを利用するには利用者が PKI アプリケーションの要求を満たす署名・暗号機能をインストールする等の作業が必要となり、PKI アプリケーションごとにクライアント環境を構築する必要があるなどの煩雑さを増す要因となる。

1 国内で使用する場合にはこのような問題がある。また国際間で 1 つの PKI アプリケーションサービスを利用しようとした場合、クライアントで準備する必要がある署名・暗号機能は輸出規制の対象となり、特定の製品に依存する PKI アプリケーションを構築することは実用的ではない。また、PKI アプリケーションが各国で使用可能な署名・暗号機能を使用するには、署名・暗号機能の仕様が統一されていないことから、PKI アプリケーションの構築が不可能となるという問題がある。

これらの問題は、国際間における PKI アプリケーションの普及を妨げる要因であり、国際間における PKI アプリケーションを普及させるにはこれらの問題解決を行うことが最大の課題であると言っても過言ではない。国際的な標準に基づいた上で、最低限の共通ルールを設定する必要がある。

平成 14 年度の本事業において、PKCS#11 に関する署名機能について共通仕様を策定した。本年度はこれを拡張し、署名機能の改善を行うと共に暗号化機能、鍵管理機能を追加することでアプリケーションからの用途を拡大した。

現在、PKI を活用するアプリケーションと PKI コンポーネントとの間のインターフェースに関して世界的な標準として使用されているものには、Windows の CryptoAPI、Java の JCA/JCE、鍵管理に関する PKCS#11 等がある。

CryptoAPI や Java は世界中で共通に使用することが可能であり、CryptoAPI では InternetExplorer のバージョン、Java では JRE（Java Runtime Environment）のバージョンを合わせることで共通な動作をさせることが可能である。

しかし、PKCS#11 については、仕様がスタンダードとして広まっているものの、ライブラリとしての実装は各国各社における製品化、或いは GNU 等におけるフリーのライブラリ化のようにそれぞれバラバラに実装を進めているのが現状であり、国際的に共通のアプリケーションを利用しようとした場合、アプリケーションと PKCS#11 ライブラリの間のインターフェースに関して以下の問題が発生する可能性がある。

- PKCS#11 ライブラリのファイル名がそれぞれ異なるため、アプリケーションから PKCS#11 ライブラリを認識させることが困難である。
- スタンダードな仕様に沿っていても、実装の際の様々な事情により詳細機能で個別差が発生し、期待どおりの動作結果を得られない可能性がある。

今回のアプリケーションインターフェース仕様の策定は、これらの問題を解消するものであり、実証実験により仕様の有効性を検証することができた。これにより、アジア圏において PKCS#11 を CryptoAPI や Java と同様に共通アプリケーションサービスで利用することが可能となり、CryptoAPI や Java と並んでアプリケーションサービスの共通利用におけるセキュリティプラットフォームの選択肢の 1 つとすることが可能となった。

一方、今回のアプリケーションインターフェース仕様の中では、署名付与・検証機能、暗号化・復号機能、鍵の安全な管理のためのラップ・アンラップ機能、1 台のクライアント PC を複数人で利用する場合を想定したマルチスロット／鍵管理機能を盛り込んだ。

実ビジネスにおけるアプリケーションの運用では、PKI に関する機能として以下の機能が必要とされている。

1. 安全な鍵管理機能
2. 送信する電子文書への電子署名付与機能
3. 送信された電子署名の検証機能
4. 電子署名と合わせて送信された署名者の証明書の有効性検証
5. 電子文書の暗号化・復号機能

6. 暗号化通信機能
7. 利用者認証機能
8. アプリケーション環境の正当性保証機能

これらの機能のうち、PKCS#11 がサポートする機能は 1.～3.と 5.であり、PKCS#11 に関する今回のアプリケーションインターフェース仕様で全て網羅されている。

更に、マルチスロット／鍵管理機能を盛り込むことでアプリケーション利用時の利便性を向上させ、より実用的な仕様とすることができた。

以上により、PKCS#11 に関しアプリケーションサービスの国際的利用における共通仕様を確立することができた。

⑵ 平成 14 年度の署名処理の課題解決

平成 14 年度に策定した署名用トークンインターフェースでは、署名アルゴリズムとして「CKM_RSA_PKCS」を使用するというものであり、これは署名対象データを PKCS#1 形式で暗号化するものであった。これは、署名対象データからハッシュ値を作成し、署名者の秘密鍵で暗号化するという本来あるべき署名付与動作とは違うものであり、以下の点で課題が残った。

・ 署名者の成りすましが可能である。
・ 署名データの改ざんを検出することができない。

署名アルゴリズムとして「CKM_RSA_PKCS」を採用した背景としては、SmartCard を使用する場合、署名値を作成する演算が SmartCard 内で行われるため、SmartCard のメモリ容量の制限からハッシュ値を暗号化する複数の処理を一括して行うことに無理が生じる危険性があったためである。

本年度はこれを改善し、署名に関するインターフェースを共通化するのみにとどまらず、署名本来の機能を実現させることができた。

改善点は次のとおりである。

・ 署名処理をハッシュ値作成とハッシュ値の暗号化の 2 つの処理に分割する。
・ ハッシュ値作成については「CKM_SHA1_PKCS」のアルゴリズムパラメータを使用し、現在、多く用いられている SHA1 アルゴリズムによりハッシュ値を作成する。
・ ハッシュ値の暗号化については、平成 14 年度と同様に「CKM_RSA_PKCS」のアルゴリズムパラメータを使用し、署名者の秘密鍵を使用してハッシュ値を暗号化する。

以上により、SmartCard のメモリ容量が厳しい環境でも、SHA1withRSA で署名値を作成する場合と同等の結果を得ることができる仕様を確立することがで

きた。

14.2　考察及び今後の展開

14.2.1 認証局間相互接続テストガイドライン

(1) keyID の計算方法について

証明書内の keyIdentifier[15](以下 keyID と略す。) の生成方法は、製品の実装や PKI ドメインのポリシによって異なる場合がある。実際に過去 3 年間にわたる実証実験に参加した認証局間においても異なる方法を採用している例があった。この keyID 生成方法の不一致が PKI ドメイン間で相互接続を行う場合にもたらす影響と、その解決策について考察する。

(a) 論理的検証

keyID の一般的な生成方法として、RFC3280 4.2.1.2 では認証局用としては公開鍵から生成する 2 つの方法と、単純に増加する整数が例示されている。しかしこれらはあくまで例示であり、keyID がすでに作成されていなければこれらの方法のうちのひとつを使用することが推奨されているが(RECOMMEND)、すでに作成されていれば作成された keyID を使うべき(SHOULD)としており、強制ではない。

IWG プロファイルでは、RFC3280 4.2.1.2(1)で示される 160bit SHA-1 hash を推奨方式とし、相互接続を行う認証局はこの方法を共通で採用すべきとしている。これは以下に述べる理由による。

パス構築の範囲が 1 つの PKI ドメイン内に閉じている限り、keyID の計算方法は、たとえローカルな方法を採用していたとしても PKI ドメイン内で統一されていれば問題はない。しかし、ローカルな keyID 生成方法を用いている PKI ドメインと相互接続を行う場合には問題が発生する可能性がある。keyID の生成方法が異なる PKI ドメイン間で相互認証証明書を発行すると、相互認証証明書を含む認証パスでは相互認証証明書の subjectKeyIdentifier と相互認証証明書の subjejct である認証局の発行する証明書の authorityKeyIdentifier 拡張の keyID が一致せず、keyID のチェーンは不整合を含むことがある。keyID はパス構築時の効率化のために記載されるものであり、上記のような不整合を含んでしまっては機能しなくなってしまう。この問題に対する認証局側の要件及び検証者側の要件を以下に述べる。

(i) 認証局側の要件

認証局としては、検証者がどのようなアプリケーションを用いるかを制御できない以上、認証局が発行する証明書の keyID の不整合をできるだけ防がなければならない。そのためには、以下の方法が考えられる。

[15] 証明書、証明書失効リスト、証明書発行要求に含まれる。

(ｲ) keyID を同じ方法で生成する(IWG では RFC3280 4.2.1.2(1)で示される keyID 生成方法を推奨している)。

(ﾛ) 相互認証証明書発行要求には keyID を必ず含め、相互認証証明書発行時には発行要求に含まれる keyID を subjectKeyIdentifier に設定する。

(ii) 検証者側の要件

keyID はパス構築時の効率化のために記載されるものであり、keyID にのみ頼ったパス構築を行うアプリケーションは、keyID に不整合がないことを保証できない場合には利用することができない。証明書検証アプリケーションは、keyID に不整合があった場合でも、正しくパス構築ができることがことが望ましい。しかし鍵更新時の証明書(Self-issued certificates)がパス構築に含まれる場合には、keyID を参照できないためにパス構築処理が複雑化するのは避けられないであろう。

(b) 現実的対応

運用をすでに開始している認証局で keyID の生成方法を変更できないケース等、前述(a)(i)(ｲ)だけの対応は現実的とは言いがたい段階になっている。セキュリティ要件を確保しつつ効率的なパス構築を行うための現実的な方法を探らなければならない。その前提であらためて検討を進める。

例として､いくつかの認証局製品の実装を｢表 14.1　認証局製品の実装状況｣に示す。

表 14.1　認証局製品の実装状況

製品	keyID 生成方法	証明書発行要求の keyID の扱い
A(ver.X)	・RFC3280 に準拠しない方式(公開鍵情報のタグ/長さ/未使用ビットを含めてハッシュ)(*1) ・変更不可	・証明書発行要求に subjectKeyIdentifier を含めない ・発行時は設定に従って subjectKeyIdentifier を生成
A(ver.Y)	・既定値は(*1)に同じ ・RFC3280 4.2.1.2(1)に変更可能	・証明書発行要求に subjectKeyIdentifier を含めない ・発行時は設定に従って subjectKeyIdentifier を生成
A(ver.Z)	・既定値 RFC3280 4.2.1.2(1) ・(*1)に変更可能	・証明書発行要求に subjectKeyIdentifier を含めて発行
B	・(*1)に同じ	・証明書発行要求に subjectKeyIdentifier を含めて発行 ・発行時は証明書発行要求中の subjectKeyIdentifier を使う
C	・既定値 RFC3280 4.2.1.2(1) ・RFC3280 4.2.1.2 (2)に変更可能	・証明書発行要求に subjectKeyIdentifier を含めない ・発行時は設定に従って subjectKeyIdentifier を生成

この例から、異なる製品間での keyID 生成方法は、一致するもの、設定変更により一致させることができるもの、一致させることができないもの、があることがわかる。設定変更により一致させることができる場合にも、すでに一致しない設定で運用している認証局では、それまでに発行した証明書と不整合が起こるため運用中の設定変更はできない。認証局製品の組み合わせ、運用状況により前述(a)(i)(ｲ)または(ﾛ)がどのような方法で可能か、あるいは不可能かが決定される。

(c) 結論

keyID の整合性が保たれていることは、パス構築の効率化に有効であり、認証局には keyID の整合性確保が要求される。また、相互認証時の証明書発行要求には keyID を記載し、相互認証証明書の発行時には、証明書発行要求に記載されている keyID をそのまま利用することで、異なる PKI ドメイン間での相互認証時における keyID の整合性を保つことができる。この方法が不可能な場合には、keyID の計算方法について事前確認が必要となる。

証明書検証アプリケーションは、keyID の整合性が確保されていない状況でも、正しくパス構築できることが望ましい。

⑵ リポジトリとしての LDAP と HTTP の比較

IWG 仕様[16]では、リポジトリは LDAP を想定し、証明書失効リストの配布点の指定方法についても DN 又は LDAP URI を用いることとしているが、実際に実証実験に参加した PKI ドメインでは、LDAP を利用している場合と HTTP を利用している場合が存在する。また、最近の証明書検証ライブラリの一部は、CRL の取得方法として、HTTP URI による指定のみをサポートしているものがある。

以下に、それぞれの特徴を考察する。

(a) データ管理方式

LDAP ではスキーマが決まっていて、標準的な階層構造を持つのに対して、HTTP では HTTP サーバのファイルシステムに依存する。また、LDAP はスキーマが標準化されていることから、標準的なデータの検索（姓、名、メールアドレスなどをキーとした検索）が容易に実現できるのに対し、HTTP では標準としては何も決まっていないが、CGI などを利用すればさらに柔軟なデータ検索は可能となる。

LDAP が階層構造の一部の管理責任の分割・委譲が標準機能として実現できるのに対して、HTTP にはそのような標準機能は存在しない。従って、HTTP

[16]日本、韓国、シンガポール、チャイニーズ台北、香港チャイナ及びタイで合意した認証局間相互接続のための技術仕様であり IWG プロファイルを含む。http://www.japanpkiforum.jp/shiryou/IWG_2002/Appendix.pdfの Appendix 2. CA-CA Interoperability Interface Specification for experiment に相当する。

サーバを用いた場合には、階層構造を持つデータを分散し管理するのは不向きと見なされることも多い。

(b) 証明書・証明書失効リスト配布手段

LDAP をリポジトリとして利用する場合には、その配布点は標準によって定められている。HTTP をリポジトリとして利用する場合には、配布点をローカルに定めなければならない。実際、認証局証明書の配布点を AIA 拡張に、証明書失効リストの配布点を cRLDP 及び iDP 拡張に記載しなければならない。

クライアント要件として、分散配布されている証明書のパス構築は、標準として配布方法や管理方法が定まっている方が好ましい。従ってこの場合は、LDAP の方が好ましいと言える。

(c) データの取得処理における相互運用性

LDAP サーバからのデータの取得時における処理では、大文字小文字、空白文字、特殊文字の正規化処理に加えて、attribute や binary オプションの処理、referral 処理など、クライアントからのリクエストに対するサーバの処理が複雑である。このことにより、標準仕様に完全に準拠していないサーバソフトウェアが存在する一方で、柔軟なデータ検索を可能としている。

HTTP サーバからのデータの取得時における処理は、配布点の変更が難しいなどの運用の柔軟性は低いものの、LDAP と比較して単純であり、相互運用性の確保がしやすい。

(d) クライアントアプリケーションの実装負荷

LDAP クライアントは、さまざまな機能（検索フィルタなど）を実装しなければならないために、HTTP クライアントの実装と比較して負荷が高くなるかもしれない。また、最近注目されている Web アプリケーションの開発においては、HTTP をリポジトリとしたほうが新たなプロトコルを開発する必要がなく効率的である。

(e) 実際の証明書検証ライブラリの実装状況

LDAP リポジトリからの CRL 取得に関しては WindowsXP 以降の Microsoft Windows であれば可能であるという報告がある。それに対して、HTTP リポジトリのからの CRL 取得は、CRLDP 拡張に HTTPURI を記載してあれば、Microsoft Windows や、JDK などのライブラリにおいて可能であると報告がある。

AIA 拡張を利用した HTTP URI を用いた証明書取得は、最近の Microsoft Windows においてサポートされているのに対して、LDAP を用いた認証局証明書取得が可能な検証ライブラリは、広く利用できるものとしては存在しない。

「表 14.2　リポジトリとしての LDAP と HTTP のそれぞれの特徴」に上記 4 項目をまとめた表を示す。

表 14.2　リポジトリとしての LDAP と HTTP のそれぞれの特徴

	LDAP	HTTP
データ管理方式	スキーマが標準として決まっている。	HTTP サーバのファイルシステムに依存する。
	標準的な階層構造に基づいて分散管理ができる（管理責任を分割・委譲できる）	データの配置方法は何も決まっておらず、管理者の自由裁量に任されている。複数の管理者による管理には不向き。
	アクセスプロトコル、スキーマ、検索ルール、API が標準として決まっているため、相互運用性を保ちやすい。	プロトコルしか決まっておらず、その他に関しては、実装者依存となっている。
証明書配布手段	証明書及び証明書失効リストの配布方法が標準で定められている。	証明書の AIA 拡張及び CRLDP 拡張を用いれば配布することは可能だが、配布点に関しては独自ルールを策定しなければならない。
データの取得処理における相互運用性	処理が複雑であり、相互運用性確保の妨げになっている一方で、柔軟な検索方法を標準で実現している。	処理が単純であり、相互運用性確保が容易である。
クライアントアプリケーションの実装負荷	多機能なため、実装負荷が高い。	特に、最近の流行である Web ベースのアプリケーション開発時に効率的である。
証明書検証ライブラリの実装状況	WindowsXP 以降の CAPI において取得が可能との報告がある。	WindowsOS 付属の CAPI 及び最近の JDK などの証明書検証ライブラリにおいて利用可能

(f) 結論

HTTP はサーバ運営が容易で、インターネット及び Web アプリケーションとの親和性が高い等メリットがあるが、異なる PKI ドメインにおける相互運用の観点で言えば、現時点ではデータ管理方式等に関する各種標準が定められ、運用実績が豊富な LDAP を採用することが望ましいといえる。

⑶ 証明書の ASN.1 構造及びエンコーディング方法に関する考察

現地タイ認証局との認証局間相互接続テストガイドライン実験において、タイが発行した証明書を日本国検証局が証明書として認識できない事象が発生した。これは、現地タイ認証局が発行した証明書に含まれる CRLDistributionPoints の DistributionPoint.fullname の ASN.1 構造及びエンコーディング方法が間違っていることに起因していた。本項では、証明書の ASN.1 構造及びエンコーディング

方法について、CRLDistributionPoints の DistributionPoint.fullname に注目して考察する。

(a) 正しい ASN.1 構造及びエンコーディング方法

RFC3280「Appendix A. Psuedo-ASN.1 Structures and OIDs」では、CRLDistributionPoints の DistributionPoint.fullname の ASN.1 構造を以下のように定義している。

```
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
(中略)

DistributionPoint ::= SEQUENCE {
     distributionPoint          [0]     DistributionPointName OPTIONAL,
     reasons                    [1]     ReasonFlags OPTIONAL,
     cRLIssuer                  [2]     GeneralNames OPTIONAL }

DistributionPointName ::= CHOICE {
     fullName                   [0]     GeneralNames,
     nameRelativeToCRLIssuer    [1]     RelativeDistinguishedName }

(中略)

GeneralNames ::= SEQUENCE SIZE (1..MAX) OF GeneralName

GeneralName ::= CHOICE {
     otherName                          [0]     OtherName,
     rfc822Name                         [1]     IA5String,
     dNSName                            [2]     IA5String,
     x400Address                        [3]     ORAddress,
     directoryName                      [4]     Name,
     ediPartyName                       [5]     EDIPartyName,
     uniformResourceIdentifier          [6]     IA5String,
     iPAddress                          [7]     OCTET STRING,
     registeredID                       [8]     OBJECT IDENTIFIER }

(中略)
END
```

上記引用部で規定している ASN.1 構造のエンコーディング方法について、X.690(12/97)「8.14 Encoding of a tagged value」では以下のような例を示している。

With ASN.1 type definitions (in an explicit tagging environment) of:

Type1 ::= VisibleString

Type2 ::= [APPLICATION 3] IMPLICIT Type1

Type3 ::= [2] Type2

Type4 ::= [APPLICATION 7] IMPLICIT Type3

Type5 ::= [2] IMPLICIT Type2

a value of:

"Jones"

is encoded as follows:

For Type1:

VisibleString	Length	Contents
$1A_{16}$	05_{16}	$4A6F6E6573_{16}$

For Type2:

[Application 3]	Length	Contents
43_{16}	05_{16}	$4A6F6E6573_{16}$

For Type3:

[2]	Length	Contents		
$A2_{16}$	07_{16}			
		[APPLICATION 3]	Length	Contents
		43_{16}	05_{16}	$4A6F6E6573_{16}$

For Type4:

[Application 7]	Length	Contents		
67_{16}	07_{16}			
		[APPLICATION 3]	Length	Contents
		43_{16}	05_{16}	$4A6F6E6573_{16}$

For Type5:

[2]	Length	Contents
82_{16}	05_{16}	$4A6F6E6573_{16}$

前述した RFC3280 の定義と上記に示す X.690 の記述を比較すると、CRLDistributionPoints の DistributionPoint.fullname に uniformResourceIdentifier を記載する場合のエンコーディング方法は、上記引用部の“Type5”と同様の方法であることが分かる。このときの TLV(Type、Length、Value)形式を、「表 14.3　TLV 表記 1」に示す。

表 14.3　TLV表記1

Type								Length	Value
クラス		構造化フラグ	タグ番号						
1	0	0	0	0	1	1	0	XX	ldap://･･･

「表 14.3　TLV 表記 1」の Type フィールドの意味を、「表 14.4　TLV 表記 1 の Type フィールド」に示す。

表 14.4　TLV表記1のTypeフィールド

項番	フィールド名	値 (2 進数)	内容	備考
1	クラス	0010	コンテキスト特定	ー
2	構造化フラグ	0000	単一型	オブジェクトが 1 つだけの値をもつ。
3	タグ番号	0 0110	タグ番号が「6」	ここでは、「uniformResourceIdentifier」を示す。

(b) 間違った ASN.1 構造及びエンコーディング方法

一方、日本国検証局が証明書として認識できなかった証明書の CRLDistributionPoints の DistributionPoint.fullname の ASN.1 構造は以下のようなものであった。

```
609 A6   99:                    [6] {
611 04   97:                      OCTET STRING
           :                    'ldap://xxx.xxxx.xxx.xx/cn=CRL1,ou=xxxx%20xx%20xx'
           :                    '%20x-xx,o=xxx,c=xx?authorityRevocationList;binar'
           :                    'y'
           :                    }
           :                  }
           :                }
```

(ホスト名、DN は実際のテストで使用した値ではない。)

上記太字部分の TLV 表記を「表 14.5　TLV 表記 2」に示す。

表 14.5　TLV表記2

Type								Length	Value
クラス		構造化フラグ	タグ番号						
1	0	1	0	0	1	1	0	99	

「表 14.5　TLV 表記 2」の Type フィールドの意味を、「表 14.6　TLV 表記 2 の Type フィールド」に示す。

表 14.6　TLV 表記 2 の Type フィールド

項番	フィールド名	値 (2 進数)	内容	備考
1	クラス	0010	コンテキスト特定	ー
2	構造化フラグ	0001	構造型	オブジェクトが複数の値を持っている。
3	タグ番号	0 0110	タグ番号が「6」	ここでは、「uniformResourceIdentifier」を示す。

「表 14.6　TLV 表記 2 の Type フィールド」から分かるようにコンテキスト特定クラスでタグ番号が 0x06(uniformResourceIdentifier)となっているにも係わらず、Type フィールドの構造化フラグが“1(構造型)”となっている。しかし、前述したように RFC3280 及び X.690 では uniformResourceIdentifier を入れる場合は、構造化フラグを“0(単一型)”にすることを規定しており、「表 14.5 TLV 表記 2」のような TLV は標準に準拠していない。この誤った ASN.1 構造及びエンコーディング方法が原因で、日本国検証局は現地タイ認証局の証明書を証明書として認識できなかった。

なお、該当部分のエンコーディング方法を修正した証明書の検証は問題なく行えた。タイが ASN.1 構造及びエンコーディング方法を修正して発行した証明書を以下に記載する。

```
652 86    97:                          [6]
              :                        'ldap://xxx.xxxx.xxx.xx/cn=CRL1,ou=xxxx%20xx%20xx'
              :                        '%20xxxx,o=xxx,c=xx?authorityRevocationList;binar'
              :                        'y'
```

(ホスト名、DN は実際のテストで使用した値ではない。)

(c) 問題点の検討

本問題が発生した原因は、現地タイ認証局の認証局操作員が CRLDistributionPoints の DistributionPoint.fullname の ASN.1 構造及びエンコーディング方法を間違えたからであった。現地タイ認証局では、使用している認証局ソフトの仕様のために､認証局操作員が自ら ASN.1 エンコーディングした DistributionPoint.fullname の uniformResourceIdentifier のデータを CRLDistributionPoints 領域へ格納して証明書を発行する必要があった。このような場合、認証局操作員の知識不足やオペレーションミスなどが要因となり標準(RFC3280 等)に準拠していない ASN.1 構造の証明書を発行してしまう可能性がある。そして、ASN.1 構造が標準に準拠していない場合、検証局などが証明書として認識できない問題が起こる可能性がある。この問題の発生を回避するためには、認証局で証明書を発行する際には、証明書の ASN.1 構造、エンコーディング方法が RFC3280 等の標準に準拠しているかを厳密に確認する必要がある。

一方、MS-Windows[17]上では、本実験で問題となった ASN.1 構造が標準に準拠していない証明書でも、証明書として認識することができた。また、同 MS-Windows 上では、このような証明書の CRLDistributionPoints を参照して CRL を取得できる、との報告もある。このように、ソフトウェアの中には ASN.1 構造が多少間違っていても証明書として受け入れる設計のものがある。(このことは、ASN.1 構造の間違いを発見することを難しくする要因の一つかもしれない。) 一方で、日本国検証局のように証明書の ASN.1 構造が標準に準拠しているかどうかを厳密に確認するソフトウェアも存在する。認証局は、どのようなソフトウェアでも受け入れることが可能な ASN.1 構造を有した証明書を発行するべきであると言える。

(d) 結論

ソフトウェアによっては、ASN.1 構造及びエンコーディング方法が RFC3280 等の標準に準拠していない証明書を許容するものがあるが、認証局は標準に厳密に準拠した証明書を発行するべきである。

[17] 確認した WindowsOS のバージョンは、「Microsoft Windows 2000 SP4」である。

(4) CRL 配布に関するセキュリティ考察

証明書の cRLDP と失効リストの iDP については、本年度も今までと同様に問題が発生した。タイ側の認証局ソフトウェアの制限により cRLDP と iDP に同じ値を入れることができなかった。この両者の比較について、本年度も引き続き考察する。

(a) 失効リスト分割のリスク

失効リストを分割するのは様々な場合が存在する。ひとつの認証局で複数のサービスを行っている場合はサービス毎に失効リストを分割することがありうるし、証明書の発行枚数が肥大化した結果、失効リストが非常に大きくなりそうな場合には、シリアルナンバーによって分割する場合があり得る。

失効リストを分割する場合、失効リスト置換攻撃の対策が必要となる。失効リスト置換攻撃は、失効リストを分割したときに、分割された失効リストにその失効リストが誰のための失効リストであるかを記載できないことに起因する。つまり以下のような場合において証明書検証は正常に終了してしまうのである。

失効された証明書の証明書検証を行う場合を考える。

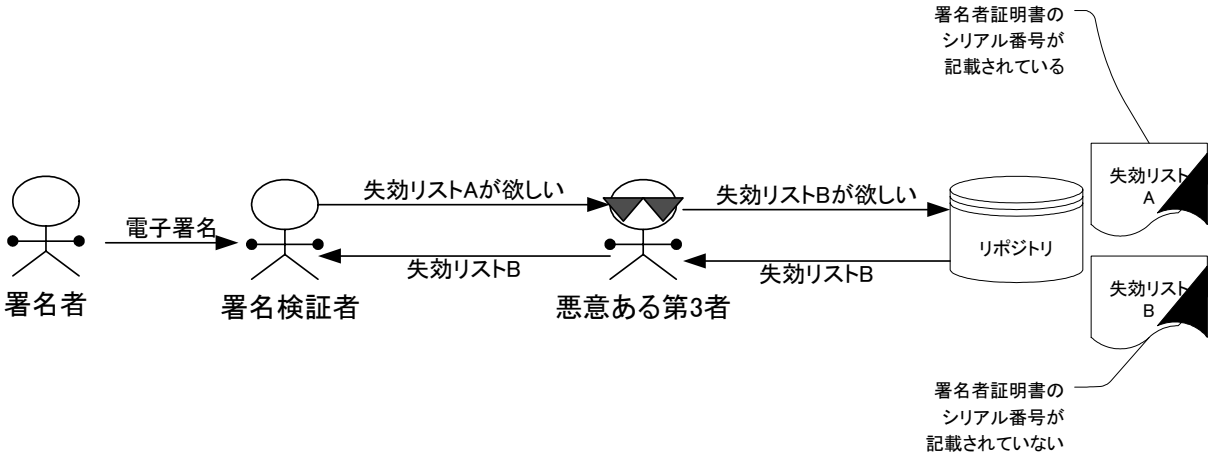


図 14.2　失効リスト置換攻撃例　(man in the middle attack)

検証者が署名者の証明書を検証する時点において、署名者の証明書は失効しており、失効リスト A にそのシリアル番号が記載されていると仮定する。「図 14.2　失効リスト置換攻撃例　(man in the middle attack)」の様にリポジトリと検証者の間に悪意ある攻撃者が存在し、失効リスト A と失効リスト B をすりかえた場合、検証者が取得に成功した失効リストには署名者のシリアル番号は記載されておらず、かつ失効リストの署名や有効期限などは有効であるため、検証者は失効されたはずの証明書を失効されていないと判断してしまう。検証者は途中ですりかえられたと判断する材料が無いからである。

過去 3 年間の議論の結果、現在の認証局間相互接続仕様においては、CRL の

配布形態とそのリスク対策について、下記の 4 通りの選択肢を示している。これらの場合分けとその対策は CRL 置換攻撃への対策を含んでいる。

・ FullCRL を発行する場合
・ CRL と ARL を発行する場合
・ CRL を分割して発行する場合
・ 複数の CRL と ARL を発行する場合

それぞれの場合について以下に詳細に記述する。

(b) 失効リストの分割と認証局側に対する推奨

失効リストを分割すると、失効リスト置換攻撃が発生するのは上述したとおりである。失効リスト置換攻撃に対する認証局側に対する推奨を以下に考察する。

(i) Full CRL を発行する場合

Full CRL を発行する場合には、失効リスト置換攻撃は発生しない。したがって、証明書及び CRL プロファイルに制約は発生しない。しかし、証明書検証者には、署名者の認証局が FullCRL を発行しているかどうかを判断する材料がない場合が存在する。したがって、CA エントリ以外で失効リストを配布する場合には、cRLDP,iDP の両方に fullName を記載するのが望ましい。

配布点は基本的には CA エントリの certificateRevocationList 属性を利用する。CA エントリ以外で配布する場合には、cRLDP オブジェクトクラスの certificateRevocationList 属性を用い、その配布点を、証明書・CRL のいずれにも記載する。

(ii) Full CRL と Full ARL を発行する場合

CRL 及び ARL をそれぞれ分割しないで発行する場合は、それぞれが CRL または ARL であることを識別することができれば、失効リスト置換攻撃を防ぐことができる。したがって、それぞれに onlyContaintsUserCert または、onlyContaintsCACert を設定すれば失効リスト置換攻撃を防ぐことができる。cRLDP.distPoint.fullName と iDP.distPoint.fullName に同じ値を入れることで対応することもできるが、前者による対応を推奨する。

配布点は基本的には各々を CA エントリの certificateRevocationList 属性又は、authorityRevocationList 属性を利用する。CA エントリ以外で配布する場合には、cRLDP オブジェクトクラスの certificateRevocationList 属性又は、authorityRevocationList 属性のそれぞれ対応する属性を用い、その配布点を、証明書・CRL のいずれにも記載する。

(iii) CRL を分割して発行する場合

CRL を CRL/ARL 以外の方法で分割する場合には、CRL 自身にその CRL の目的を記載することができない。したがって、失効リスト置換攻撃の可能性がある。失効リスト置換攻撃を防ぐためには、検証者が必要としている CRL を取得したことを確かめるための情報を CRL に含める必要がある。そのひとつの方法として、iDP.distPoint.fullName を用いる方法がある。

配布点は cRLDP オブジェクトクラスの certificateRevocationList 属性を用い、その配布点を、証明書・CRL のいずれにも記載する。混乱を避けるためにも、CA エントリは利用しないほうが良い。

(iv) 複数の CRL と ARL を発行する場合

この場合も、(iii)の場合と同様に失効リスト置換攻撃の可能性がある。onlyContaintsUserCert/onlyContaintsCACerts 及び、distPoint.fullName を組み合わせて利用することで、失効リスト置換攻撃を防ぐことができる。

配布点は cRLDP オブジェクトクラスの certificateRevocationList 属性又は、authorityRevocationList 属性のそれぞれ対応する属性を用い、その配布点を、証明書・CRL のいずれにも記載する。混乱を避けるためにも、CA エントリは利用しないほうが良い。

また、失効リストを CA エントリ以外で配布する場合には、CA エントリ以下のエントリを用いて配布することが望ましい。例えば、c=JP, o=CA という認証局が発行する失効リストは、c=JP, o=CA, cn=CRL1 といった配布点を利用することが望ましい。

(c) 失効リストの分割と検証者側に対する推奨

上述したとおり、失効リストを分割した場合には、cRLDP.distPoint.fullName と iDP.distPoint.fullName を比較しなければならない場合がある。前述した通り、証明書検証者は披検証者の認証局が失効リストをどのケースで配布しているかを知ることが困難な場合がある。以下の場合わけに応じて考察する。

それぞれの場合について以下に詳細に記述する。

(i) CRLDP, iDP がそれぞれ存在しない場合

この場合は、検証者側は FullCRL が配布されていると仮定するしかない。従って、失効リスト置換攻撃対策は不要である。

(ii) CRLDP が存在せず iDP に onlyContaintsXXCert のみが記載されている場合

この場合は、検証者側は、FullCRL と FullARL が配布されていると仮定するしかない。このときには、対象となる証明書の種別 と失効リストの種別

が正しいことを確認する必要がある。

(iii) CRLDP.distPoint.fullName 及び iDP.distPoint.fullName がある場合

この場合には、検証者側は失効リストが何らかの形で分割配布されているか、もしくは、CA エントリ以外で配布されていることを仮定できる。この場合の失効リスト置換攻撃対策としては、前者であることを前提にしなければならない。従って、CRLDP.distPoint.fullName と、iDP.distPoint.fullName の値(複数記載されている場合は記載されている配布点のどれかひとつ)が一致しなければならない。昨年度までの認証局間相互接続仕様では、この両者は同じ値でなければならないとしている。しかし、認証局ソフトウェアの実装によっては、両者が異なってしまう場合がある。この両者を比較することは、意図した失効リストを取得しているかどうかの確認であるので、上記の確認方法で問題ない。

(d) 結論

IWG プロファイルでは、cRLDP と iDP の値は同じであることとしているが、必ずしもその必要は無い。cRLDP 及び iDP に複数の値を入れるときにはいずれか 1 つが同じであれば問題なく平成 14 年度 IWG プロファイルはオーバースペックであり、本年度版において改訂した。

14.2.2 パス検証テストガイドライン

(1) DN マッチングとテストケースデザインポリシ

DN マッチングの課題を解決するためのテスト項目設計及び、実験が行われた。しかしながら、設計時に、DN マッチングルールがあまりにも整理されていない状況にあることが判明した。DN マッチングは、パス構築、LDAP リクエスト処理、パス検証等に使われ、それぞれマッチング処理の意図が違ってくる。今回の実験では、DN マッチングを、「認証局が責任を持って構築する名前連鎖に対する検証者による確認処理」として位置づけるようにした。

本考察では、DN マッチングの不明確部分を挙げ、なぜ上記のような位置づけを行いテスト項目として纏めていったのかを論じることとする。従ってここでは、まず DN マッチングの概要を述べ、次に不明確部分を記述し、DN マッチングデザイン、そして最後に結論を述べる。

(a) DN とは

Distinguished Name (DN)とは、RDN 列(RDNSequence)であり、RFC 3280 では、以下のように定義している。

```
Name ::= CHOICE {RDNSequence }
RDNSequence ::= SEQUENCE OF RelativeDistinguishedName
RelativeDistinguishedName ::=
  SET OF AttributeTypeAndValue
AttributeTypeAndValue ::= SEQUENCE {
  type   AttributeType,
  value   AttributeValue }
AttributeType ::= OBJECT IDENTIFIER
AttributeValue ::= ANY DEFINED BY AttributeType
```

例えば、C=JP や O=PKI-J や CN=Foo Bar は、RDN と呼び、この列を DN と呼ぶ。

(b) DN マッチングとは

DN マッチングとは、2 つの DN 値(C=JP,O=PKI-J 等)を比較することであるが、どこの処理に対するものなのかは不明確なところが多い。DN マッチングが関連する処理は、

1) LDAP Request を受け取り、当該証明書を検索する LDAP サーバ
2) パス構築を処理する PKI クライアント
3) パス検証を処理する PKI クライアント

等が考えられる。

1) の処理は、情報検索サービスの一環として、不特定多数のクライアントからの問い合わせに対して一定のルールをもうけ、そのルールで検索結果を探し出すというサーバ側処理である。

2)、3) は、検証者が証明書の名前連鎖（結びつき）を見つける及び確認するという処理である。証明書とは鍵と名前を結びつけることで成立する。この結びつきの妥当性は、正しい上位認証局から発行されていることが前提である。正しい上位認証局から発行されたという検証は、「図 14.3　鍵と名前の検証」のように上位認証局の鍵と名前を検証することで行われる。

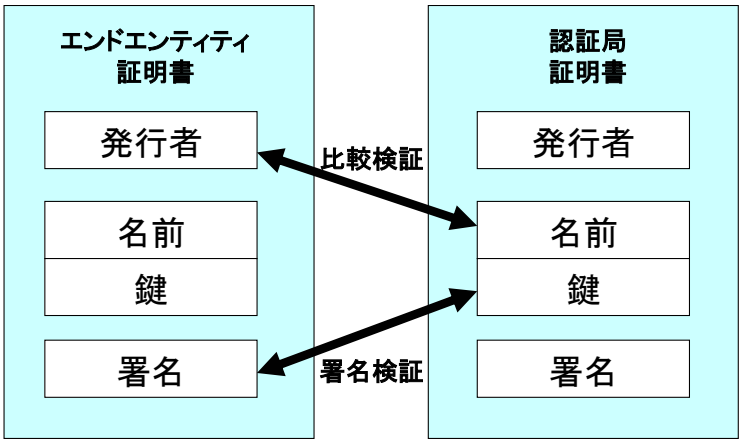


図 14.3　鍵と名前の検証

ここでの重要なポイントは、認証局は名前連鎖の正しさの責任を持ち、検証者は、その正しさを確認するという関係である。つまり認証局側がエンドエンティティの名前及び自分自身の名前を証明書に正しく設定し、認証局証明書からの名前連鎖を保証するべきであって、検証者側は多くの解釈やルールを持つべきではなく、単に確認を行うべきであるということにある。

従って今回のパス検証テストで行う DN マッチングとは、パス検証時に行われる「認証局が責任を持って構築する名前連鎖に対する検証者による確認処理」と定義した。

実際の処理としては、検証対象証明書の発行者名(Issuer Name)と上位認証局証明書の主体者名(Subject Name)の比較を行う。また、subjectAltName, issuerAltName, cRLDistributionPoints の拡張等にも挿入され比較されることがある。

(c) DN マッチングルールとは

RFC3280 では、どこの部分がどのように比較されるべきであるか検討している。基本的には、X.500 シリーズで規定されているマッチングルールのサブセットであると明言している。以下にマッチングルールに関する記述を RFC3280 より引用する。

(a)　attribute values encoded in different types (e.g., PrintableString and BMPString) MAY be assumed to represent different strings;

(b) attribute values in types other than PrintableString are
case sensitive (this permits matching of attribute values as binary objects);

(c) attribute values in PrintableString are not case sensitive (e.g., "Marianne Swanson" is the same as "MARIANNE SWANSON"); and

(d) attribute values in PrintableString are compared after removing leading and trailing white space and converting internal substrings of one or more consecutive white space characters to a single space.

引用 a) の要件から順に解釈すると、異なったエンコーディングタイプの属性値(attribute value)は、違うものとみなしてもよい。そして、PrintableString エンコードタイプ以外は、大文字小文字の区別される（つまりバイナリマッチング処理してもよく）、PrintableString は、大文字小文字の区別されない（つまり、文字列の正規化が行われる）。

(d) 不明確な部分

(i) マッチング部分

上記の記述では、マッチングさせる部分は、属性値(attribute value)である記述されているが、どこの値を実際マッチングさせるのであろうか。

属性値の ASN.1 データ構造はデータ型を表すタグ(T)、値データ長(L)および値(V)により構成される TLV 構造を持っているため、まず T 部分を検証し、PrintableString かそれ以外かを判別する処理が入ると考えられる。 そして PrintableString の場合、文字列の正規化が必要となるために、正規化を行う値、つまり属性値の V 値が比較対象となる。バイナリマッチング時においても V 値のみを取り出し、比較とすることで処理が効率化すると考えれらる。

(ii) バイナリマッチングかストリングマッチングか両方

次の問題として、もしバイナリマッチングする場合でも、バイナリマッチングとストリングマッチングを両方実装すべきであろうか。我々はこの問いかけ対し、両方実装すべきであると考える。これは単なる仕様の範囲でしかないが、いくら UTF8String で文字をエンコードした DN を作ろうとしても

"Country"属性は、PrintableString でエンコードされなければならない。従って、Country 属性に対しては、ストリングマッチングを採用することが望ましい。

(iii) PrintableString と UTF8String が両方混在する場合のマッチングルール

PrintableString と UTF8String には、両方同じコードポイントを持つアスキーキャラクタが入る。従って、属性値の V 値のみを比較した場合は、同じと解釈される。しかしながら、前述 RFC3280 のマッチングルールでは、エンコーディングタイプが違えば、違う DN とみなしてもよいという記述が存在する。一般的に、UTF8String 及び PrintableString 両方許す証明書プロファイルが多いためマッチングルールの解釈に、ぶれが生じる可能性がある。

(e) DN マッチングデザインポリシ

IWG プロファイルを想定したパス検証テストガイドラインにおけるテスト項目の設計指針として、バイナリマッチング及びストリングマッチングどちらを前提とすべきであろうか。パス検証ガイドラインで採用したのは以下である。

1) アジア圏の証明書プロファイルの DN として当然 CJK キャラクタが含まれると予想される。
2) CJK キャラクタのみならずアスキーキャラクタも含まれるかもしれない。また英語/漢字が日常に使用される文化圏も存在するために、キャラクタが混合する場合も考えられる。
3) また実際に相互認証する場合は、UTF8String のみとは限らず、PrintableString もあり得る。
4) 最後に、証明書の DN の一致は、認証局で確保されなければならない。つまり検証者は、その一致を確認するだけである。(検証者は DN マッチングにおいてキャラクタ表記の多様性を解釈するべきではない) 各認証局は、名前の一意性と完全性は保証すべきである。

これを受け、テスト項目の設計方針としては以下を挙げる。

1) CJK キャラクタの存在から UTF8String を基本エンコードタイプとする。しかしながら PrintableString も可とする。
2) CJK キャラクタの存在からマッチングルールは、バイナリマッチングを基本とする。UTF8String と PrintableString が混在する場合、バイナリマッチングとし、両方が PrintableString の場合のみ、ストリングマッチングとする。

3) 運用で名前の挿入は、依頼された値と"まったく同様"に挿入することを前提とする。
4) 名前の挿入と同じで、DN の属性(RDN)の数や並びはまったく同じに設定されなければならない。
5) エンコードタイプを変えるだけでは、違う名前とはみなさない。

つまり、DN マッチングルールの基本思想は、認証局によって"すべて同じに挿入されるはず"であり、違えばエラーである。検証者は多様性を解釈せず、認証局が設定する DN 連鎖を"確認"するのみというポリシを採用している。CA-CA 相互接続環境を考慮すれば、事前に認証局間における DN 連鎖が保証されていることが前提条件であり、検証者はその認証局間で保証された DN 連鎖を"確認"するのみである。簡単なアルゴリズムを以下に示す。

```
If( T(type)s of both attribute values == PrintableString){
   stringNormalization( V(value)s of both attribute values )
}
result = binaryCompare( V(value)s of both attribute values )
```

(f) 結論

アジア圏で予想される DN は言語の多様性から複雑化する傾向にあるため、DN の連鎖は、認証局(間)で保証されるべきであり、検証者はそれを確認するのみというポリシを採用した。認証局は"まったく同様"な DN を挿入しなければならない。 そして、DN マッチングルールは、基本的にバイナリマッチングとし、間違えればエラーとする。

しかしながら、レガシーシステムとの共存のために、PrintableString を用いてもよい。しかしそこでも認証局は、まったく同様な DN を挿入するべきであり、エンコーディングタイプの違いが DN の違いと見なせないことを追記しておく。つまり、PrintableString と UTF8String の混合の場合、バイナリマッチングでマッチすれば検証は成功である。

(2) PrintableString から UTF8String エンコーディングへの移行

DN マッチングのテスト項目設計時に挙がった課題として、PrintableString から UTF8String へ移行がある。現在 IPA 勧告や PKIX でも指摘されている。この考察では、これらの議論を参考にし、この移行時の DN マッチングルールに与えるインパクトを考察する。

(a) PrintableString から UTF8String への移行とは

RFC3280 において、DN に使用できるエンコードタイプは、DirectoryString であると規定されており、その DirectoryString は、PrintableString,TeletexString, BMPString, UTF8String, and UniversalString から一つ選択することになっている。

ここでなぜ移行の話が出て来るかというと、IETF 全体としてストリングエンコード方式は、UTF8String へ移行していることから、RFC3280 では、UTF8String を優先しており、2003 年 12 月 31 日からは DN は、PrintableString を使っている場合は、UTF8String に移行するように推奨している。以下が記載となる。

> The UTF8String encoding [RFC 2279] is the preferred encoding, and all certificates issued after December 31, 2003 MUST use the UTF8String encoding of DirectoryString (except as noted below)
>
> (a)
> CAs MAY issue "name rollover" certificates to support an orderly migration to UTF8String encoding. Such certificates would include the CA's UTF8String encoded name as issuer and the old name encoding as subject, or vice-versa.
>
> (b)
> As stated in section 4.1.2.6, the subject field MUST be populated with a non-empty distinguished name matching the contents of the issuer field in all certificates issued by the subject CA regardless of encoding.

(b) 問題点 1

移行の例外とされる name rollover certificate とは一体何なのかよく理解できない。この証明書が現在のドメインの何処の証明書に当るのか不明確である。一般的には、self-issued certificate と理解されるが、この場合証明書検証エンジンに影響を与える。basicConstraints のパス長や policyConstrains のポリシ長である。従って、この name rollover certificate たる証明書はおそらく明確な記述が新たにされない限り誰も発行しないであろう。この議論は、PKIX で行われている。

(c) 問題点 2

PrintableString から UTF8String に移行すると DN マッチングルールが変わるために、予期しない検証結果を生む可能性がある。これは最近の IPA 勧告の指摘である。例えば、以下の例を取って考えてみよう。P は PrintableString, U は UTF8String とする。

(a) ある証明書の issuer 名：C=JP, O=PKI-J, CN=Certificate Authority (P)
(b) CA 証明書の subject 名　：C=JP, O=pki-j, CN=Certificate Authority (P)

この 2 つの証明書が PrintableString でエンコードされていれば、検証はマッチする。しかし、PrintableString から UTF8String へ移行した場合、以下のようなパスを作る可能性がある。特に相互認証を利用する環境で起きやすい。

一方の相手のみが UTF8String へ移行する場合、DN マッチングルールは、実際どちらを使うかは明確にはなっておらず、もしバイナリマッチングルールを採用しているのであれば、検証は失敗し、ストリングマッチングルールを採用しているのであれば、検証は成功する。

(a) ある証明書の issuer 名：C=JP, O=PKI-J, CN=Certificate Authority (P)
(b) CA 証明書の subject 名　：C=JP, O=pki-j, CN=Certificate Authority (U)

両方が UTF8String へ移行する場合、両方が UTF8String となっているので、バイナリマッチングを採用することが必要となっており、検証に失敗する。なぜなら PrintableString 場合は、”pki-j”はストリングの正規化をするために、”pki-j”であっても、”PKI-J”であっても同じストリングとして見なされるのに対して、UTF8String では”pki-j”と”PKI-J”は違うオブジェクトとして扱われるために、異なってしまう。

(a) ある証明書の issuer 名：C=JP, O=PKI-J, CN=Certificate Authority (U)
(b) CA 証明書の subject 名　：C=JP, O=pki-j, CN=Certificate Authority (U)

従って、移行する場合、移行前のマッチングルール、移行時のマッチングルール、移行後のマッチングルールを意識しながら DN を表す値を変更する必要がある。

(d) 結論

PrintableString では、大文字と小文字は区別しないために、この値が運用

上の基礎になってしまうと問題が発生する可能性が高いため、まず第一にその可能性をなくす必要がある。意図的に発行証明書の発行者名と認証局証明書の主体者名を異なるようには設定しないと考えられる。このような場合が発生するのは認証局によるオペレーションミス(subjectAltName に対して手入力等)によるものであろう。従って認証局は、正しく値が挿入されるような工夫を施すべきである。

また、DN マッチングポリシを明確にし、全く同じ検証結果を得られるようにすべきである。前述の考察を参照にしていただければ幸いである。

更には、IPA の勧告どおり、問題 1 と問題 2 がクリアになるまでは、慌てて証明書を再発行する等は行わない方がよいと思われる。また、IETF 等ではこれらの問題について明確な方針と方法の作成を早急に議論される必要があるであろう。

(3) CJK キャラクタと DN マッチング

アジア PKI として最も関心のあるトピックの一つである CJK キャラクタと DN マッチングに関する考察を行う。CJK キャラクタは、そのキャラクタ数や中国、韓国および日本の異文化圏で使用されている漢字が統合されているためどのようなテストをおこなえばよいのか、また何が一体懸案となるのかが予想できない。CJK キャラクタに対して DN マッチングルールとして、バイナリマッチングを採用しテストをしたが、幾つかの懸案が挙がった。

本考察では、今回採用した CJK キャラクタの範囲と実験内で発生した懸案事項を述べ、最後に推奨を述べることとする。

(a) CJK キャラクタとは

CJK キャラクタとは Unicode で登録されている漢字部分の文字コード領域であり、ここでは中国語、日本語、韓国語を含んでいる。ベトナム語も含めて CJKV と呼ぶ場合もある。さて、本年度のパス検証テストガイドラインの改訂で対象となる CJK キャラクタとして検討された範囲を以下に定めた。

CJK Unified Ideographs
CJK Compatibility Ideographs
Hiragana
Katakana
Halfwidth and Fullwidth Forms

Hangul Syllables
CJK Symbols and Punctuation
CJK characters mixed with ASCII characters

Microsoft Windows XP で実際に表示できるもののみを範囲[18]とした。

(b) CJK キャラクタの DN マッチング

実験の結果、バイナリマッチングで十分に動作することは確認できた。

実際相互認証をする場合は、

a) CJK キャラクタを十分に確認しながら証明書を発行する作業、
b) および検証作業

の大切さを確認した。つまり、

本来、PKCS#10 のデータに記述されている主体者名は、自動的に証明書の中へ書き込まれるために、CJK であろうがなかろうが、バイナリマッチングに不適切なオペレーション環境は生まれるとは考えにくい。しかしながら、URI 等に書き込むときに、手作業のオペレーションがうまれる。その時全く似た CJK キャラクタが存在しすれば、目視では区別できない場合があり、誤ったキャラクタが入力される可能性がある。

ここで強調したいことは、必ず主体者名や発行者名のコードポイントを参照し、目視のみの確認作業での手入力は、避けることを心がけることである。更に、その後に必ず検証作業を行い、バイナリマッチングできる環境かどうかを確認すべきである。

(c) 懸案 1：Stringprep と CJK キャラクタ

CJK キャラクタの DN マッチングに Stringprep を使用すればどうかという議論がある。この問題は、CJK キャラクタの中には、前述したが同じに見える文字が存在することから起因する。この場合のマッチングを行うために、RFC3454 において Stringprep という処理を標準化している。基本的にこの処理は、CJK キャラクタをバイナリで比較するのではなく、ストリングマッチングで比較しようとするものである。しかしながら、この Stringprep の導入は、

[18] Microsoft Office XP で提供されてる Arial Unicode MS フォントは 5 万以上のグリフを持ち、現在最も広い範囲をカバーする Unicode フォントである。平成 15 年度実験では、これを実験範囲とした。

時期尚早又は必要がないと判断してもよいのではと考える。理由は以下に述べる。

1) Stringprep は、DN マッチングの意図から外れる可能性がある。DN マッチングとは、CA が意図した DN 連鎖が正しいかの確認である。CA は、責任を持って依頼された"名前"を正確に証明書に記入し発行しなければならない。特に相互認証関係とは、そのようなものであろう。わざわざ Stringprep 処理を行って検証者にとって一層の利益になるとは思えない。むしろ認証局のオペレーションミスを誘発する方向につながると考えられる。

2) Stringprep を採用することは、新たな相互運用性の確認を必要とする。つまり、すべてが一社のライブラリを使うわけではなく、他社のライブラリや自社で開発したりすると想定され、これらのテストを十分に行う必要がある。実際必要とする局面と相互運用性の確保のコストを判断してみても、まだ Stringprep を必要としている段階にはなく、これからもないのではないかと想像する。

(d) 懸案 2：変換表の相違

一部の文字において、意図した文字へ正確に UTF8String に変換できない事態が発生する可能性がある。例えば、YEN 記号であるが、SJIS では、0x5C である。ベンダは独自の UTF8 変換表を作成しているために、SJIS で確認できる文字が意図とは違った文字へとマッピングされる可能性を残す。変換表としては、標準団体が作成したもの、ベンダが作ったものが存在する。以下がその例である。

SJIS では、	0x5C
x-sjis-jdk1.1.7 では、	U+005C
x-sjis-unicode-0.9 では、	U+00A5
x-sjis-jisx0221-1995 では、	U+00A5
x-sjis-cp932 では、	U+005C

x-sjis-jdk1.1.7 では、SJIS の 0x5C を Unicode の U+005C に変換している。また、EUC でも違う変換表を使っている。（「図 14.4　変換表」参照）

ある環境では、現在は、直接 Unicode のコードポイントを入力し、表記できる場合があるために、変換表の相違は関係ないかもしれない。しかしながら、その場合でも、どこで Unicode に変換されるか留意する必要がある。例えば、

ブラウザを使った入力端末は、Windows であり、直接 Unicode 入力は可能であるが、一旦 EUC 等に変換され、証明書発行ソフトウェアが稼動している UNIX 環境へ転送されるかもしれない。

詳細は、http://www.w3.org/TR/2000/NOTE-japanese-xml-20000414/XML Japanese Profile を参照。

(e) 結論

1) ソフトウェアによっては前述したベンダ特定および標準団体特定の変換表を使って、EUC->UTF8 あるいは UTF8->EUC(とある内部コード)->UTF8 のように変換を行っている認証局ソフトウェア、検証ソフトウェアがある。実装では、最後に文字を表示する部分(即ちプレゼンテーションレイヤー)でのみ、このようなコード変換を行うようにする。入力レイヤーでは意図しない変換が起こりそうな文字については拒否するような機能が必要である。
2) 自分の認証局ソフトウェアや、検証クライアントが UTF8 スルーであるかどうかを確認し、内部で変換を行っていることが判明した場合、二国間で利用可能な文字セットを合意しておく必要がある。
3) subjectAltName 等 DN 値をマニュアルで入力する可能性が少なからずあるため、その場合はコードポイントに基づいた文字入力を行うべきである。
4) 変換表で指摘されている疑わしいキャラクタは「図 14.5　注意すべき文字」になる。これらの文字は、DN には入力しない/DN のデザインをしないことを推奨する。

Octets in Shift-JIS	x-sjis-jdk1.1.7	x-sjis-unicode-0.9	x-sjis-jisx0221-1995	x-sjis-cp932
0x5C(YEN SIGN)	U+005C(REVERSE SOLIDUS)	U+00A5(YEN SIGN)	U+00A5(YEN SIGN)	U+005C(REVERSE SOLIDUS)
0x7E(OVERLINE)	U+007E(TILDE)	U+203E(OVERLIN	U+203E(OVERLINE)	U+007E(TILDE)
0x815C(FULLWIDTH EM DASH)	U+2015(HORIZONTAL BAR)	U+2015(HORIZONTAL BAR)	U+2014(EM DASH)	U+2015(HORIZONTAL BAR)
0x815F(REVERSE SOLIDUS)	U+005C(REVERSE SOLIDUS)	U+005C(REVERSE SOLIDUS)	U+005C(REVERSE SOLIDUS)	U+FF3C(FULLWIDTH REVERSE SOLIDUS)
0x8160(WAVE DASH)	U+301C(WAVE DASH)	U+301C(WAVE DASH)	U+301C(WAVE DASH)	U+FF5E(FULLWIDTH TILDE)
0x8161(DOUBLEVERTICAL LINE)	U+2016(DOUBLEVERTICAL LINE)	U+2016(DOUBLEVERTICAL LINE)	U+2016(DOUBLEVERTICAL LINE)	U+2225(PARALLEL TO)
0x817C(MINUS SIGN)	U+2212(MINUS SIGN)	U+2212(MINUS SIGN)	U+2212(MINUS SIGN)	U+FF0D(FULLWIDTH HYPHEN-
0x8191(CENT SIGN)	U+00A2(CENT SIGN)	U+00A2(CENT SIGN)	U+00A2(CENT SIGN)	U+FFE0(FULLWIDTH CENT SIGN)
0x8192(POUND SIGN)	U+00A3(POUND SIGN)	U+00A3(POUND SIGN)	U+00A3(POUND SIGN)	U+FFE1(FULLWIDTH POUND SIGN)
0x81CA(NOT SIGN)	U+00AC(NOT SIGN)	U+00AC(NOT SIGN)	U+00AC(NOT SIGN)	U+FFE2(FULLWIDTH NOT SIGN)
Extended characters in 13th, 89th-92th, and 115th-119th rows	None	None	None	Included

Octets in Japanese EUC	x-eucjp-unicode-0.9	x-eucjp-jisx0221-1995	x-eucjp-open-19970715-ms	x-eucjp-open-19970715-0201	x-eucjp-open-19970715-ascii
0x5C(REVERSE SOLIDUS)	U+005C(REVERSE SOLIDUS)	U+005C(REVERSE SOLIDUS)	U+005C(REVERSE SOLIDUS)	U+00A5(YEN SIGN)	U+005C(REVERSE SOLIDUS)
0x7E(TILDE)	U+007E(TILDE)	U+007E(TILDE)	U+007E(TILDE)	U+203E(OVERLIN	U+007E(TILDE)
0xA1B1(OVERLINE/NEGATION	U+FFE3(FULLWIDTH MACRON)	U+FFE3(FULLWIDTH MACRON)	U+FFE3(FULLWIDTH MACRON)	U+FFE3(FULLWIDTH MACRON)	U+203E(OVERLINE)
0xA1BD(FULLWIDTH EM DASH)	U+2015(HORIZONTAL BAR)	U+2014(EM DASH)	U+2015(HORIZONTAL BAR)	U+2014(EM DASH)	U+2014(EM DASH)
0xA1C0(REVERSE SOLIDUS)	U+005C(REVERSE SOLIDUS)	U+005C(REVERSE SOLIDUS)	U+FF3C(FULLWIDTH REVERSE SOLIDUS)	U+005C(REVERSE SOLIDUS)	U+FF3C(FULLWIDTH REVERSE SOLIDUS)
0xA1C1(WAVE DASH)	U+301C(WAVE DASH)	U+301C(WAVE DASH)	U+FF5E(FULLWIDTH TILDE)	U+301C(WAVE DASH)	U+301C(WAVE DASH)
0xA1C2(DOUBLE VERTICAL LINE)	U+2016(DOUBLE VERTICAL LINE)	U+2016(DOUBLE VERTICAL LINE)	U+2225(PARALLEL TO)	U+2016(DOUBLE VERTICAL LINE)	U+2016(DOUBLE VERTICAL LINE)
0xA1DD(MINUS SIGN)	U+2212(MINUS SIGN)	U+2212(MINUS SIGN)	U+FF0D(FULLWIDTH HYPHEN-MINUS)	U+2212(MINUS SIGN)	U+2212(MINUS SIGN)
0xA1EF(YEN SIGN)	U+FFE5(FULLWIDTH YEN SIGN)	U+FFE5(FULLWIDTH YEN SIGN)	U+FFE5(FULLWIDTH YEN SIGN)	U+FFE5(FULLWIDTH YEN SIGN)	U+00A5(YEN SIGN)
0xA1F1(CENT SIGN)	U+00A2(CENT SIGN)	U+00A2(CENT SIGN)	U+FFE0(FULLWIDTH CENT SIGN)	U+00A2(CENT SIGN)	U+00A2(CENT SIGN)
0xA1F2(POUND SIGN)	U+00A3(POUND SIGN)	U+00A3(POUND SIGN)	U+FFE1(FULLWIDTH POUND SIGN)	U+00A3(POUND SIGN)	U+00A3(POUND SIGN)
0xA2CC(NOT SIGN)	U+00AC(NOT SIGN)	U+00AC(NOT SIGN)	U+FFE2(FULLWIDTH NOT SIGN)	U+00AC(NOT SIGN)	U+00AC(NOT SIGN)
0x8FA2B7(TILDE)	U+007E(TILDE)	U+007E(TILDE)	U+FF5E(FULLWIDTH TILDE)	U+007E(TILDE)	U+FF5E(FULLWIDTH TILDE)
0x8FA2C3(BROKEN BAR)	U+00A6(BROKEN BAR)	U+00A6(BROKEN BAR)	U+FFE4(FULLWIDTH BROKEN BAR)	U+00A6(BROKEN BAR)	U+00A6(BROKEN BAR)

Referred from W3C (Worlde Wide Web Consortium). XML Japanese Profile, http://www.w3.org/TR/2000/NOTE-japanese-xml-20000414, 2000.

図 14.4　変換表

(出典：W3C (Worlde Wide Web Consortium). XML Japanese Profile, http://www.w3.org/TR/2000/NOTE-japanese-xml-20000414, 2000)

¥	YEN SIGN
¥	FULLWIDTH YEN SIGN
~	OVERLINE/TILDE
～	WAVE DASH
～	FULLWIDTH TILDE
−	MINUS SIGN
－	FULLWIDTH HYPHEN-MINUS
–	FULL WIDTH EM DASH
—	EM DASH
―	HORIZONTAL BAR
￣	FULLWIDTH MACRON
\	REVERSE SOLIDUS
＼	FULL WIDTH REVERSE SOLIDUS
‖	DOUBLE VERTICAL LINE
∥	PARALLEL TO
¢	CENT SIGN
￠	FULLWIDTH CENT SIGN
£	POUND SIGN
￡	FULLWIDTH POUND SIGN
¦	BROKEN BAR
￤	FULLWIDTH BROKEN BAR
¬	NOT SIGN
￢	FULLWIDTH NOT SIGN

図 14.5　注意すべき文字

(4) 認証局ソフトウェア製品とツール利用の比較

平成 15 年度パス検証テストガイドラインの検証実験において、証明書および証明書失効リストの発行はテストツールを用いて行った。認証局ソフトウェア製品によってもこれらの発行は可能であるが、パス検証テストにおける両者利用の差異について考察する。

(a) イレギュラーなケースを含む幅広い種類のテストデータの生成

パス検証テストで用いる証明書や証明書失効リストでは、RFC3280 や X.509 等の標準仕様や IWG プロファイルに準拠したデータはもちろん、準拠しないデータもまた必要となる。認証局ソフトウェア製品では、プロファイルに基づく厳密な運用に基づく証明書等の発行を行うため、例外的な証明書等の発行に向かない。また、製品によってはディレクトリ名などに UTF8String を指定した場合、日本語や韓国語などの文字を含む証明書を発行できない製品もある。テストツールを用いた場合、日本語文字、中国語繁体字文字、中国語簡体字文字、韓国語文字、タイ語文字、またその他の特殊文字(チルダ、バックスラッシュ、バー、様々なハイフン記号など)を含む証明書を発行できテストすることができる。

(b) UTF8 スルーな証明書、証明書失効リスト発行

アジア諸国との PKI 相互運用実証実験を行うには、証明書中のディレクトリ名において各国の文字を含み、これを正しく検証できる必要がある。また特殊

な文字についても正しく扱える必要がある。

例えば「図 14.6　UTF8 スルーでない認証局ソフトウェアによる証明書発行(例)」で示すように一部の認証局ソフトウェアでは入力のフロントエンドである GUI 部分において内部コードとして日本語 EUC の文字コードを使っていたとする。入力された EUC の日本語文字列は変換表を用いて UTF8 に変換され UTF8String としてディレクトリ名等で格納されることになるが、変換表には OS や Windows、Java などの環境やプログラミング言語の違い、JIS・シフト SJIS・EUC などの文字コードの違いにより幾つかの変換表が存在しており、どの変換表を使うかによって、変換された結果の文字が異なる場合がある。これは「変換表問題」として知られているおり本報告書中「14.2.2(3)(d)懸案 2：変換表の相違」において詳説されている。

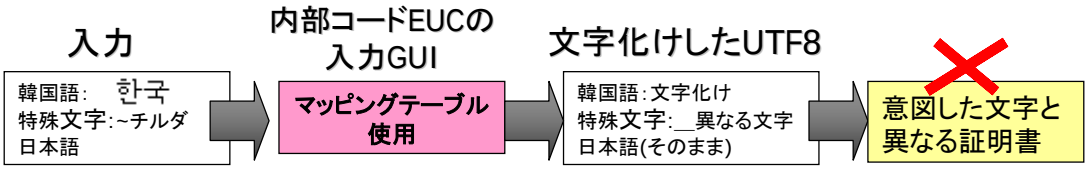


図 14.6　UTF8 スルーでない認証局ソフトウェアによる証明書発行(例)

テストツールにおいては、GUI 部分はウェブブラウザで表示される UTF8 によるデータ編集フォームであり、管理データベースには Unicode を扱えるデータベースを用いている。「図 14.7　UTF8 スルーなテストツールによる証明書発行」に示す通りデータベース中のデータに基づき発行される証明書や証明書失効リストは何ら文字を変更せずに UTF8String として格納される。即ち入力した Unicode の文字が意図した通りに証明書発行されるのである。

本機能は UTF8 CJK の実験データ作成の際に使用されているが、認証局ソフトウェア製品で Unicode の CJK 文字や特殊文字を正しく発行できる製品は少ない。

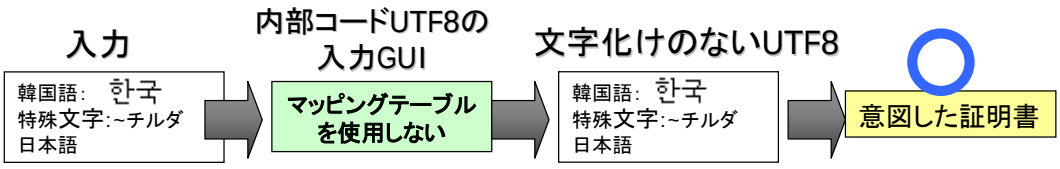


図 14.7　UTF8 スルーなテストツールによる証明書発行

(c) 一部の属性のみが異なる証明書、証明書失効リストの発行

パス検証テストパターンでは、一つの属性値、一つの拡張領域の値のみが異

なり、他は同じであるような証明書や証明書失効リストを対象にテストを行う場合が多い。認証局ソフトウェア製品を用いてこのようなデータを生成するには、個別のプロファイルが必要になる。作成したプロファイルを用いて発行する証明書が 1 つだけであったとしても、プロファイルは各々必要となり、作業負荷が高い。平成 15 年度設計されたテストデータでは cRLDistributionPoints が一部だけ異なるデータを多数生成する必要があったが、テストツールではデータのコピーおよび編集機能により、一旦コピーした後、違いのあるデータのみを変更し保存すればよい。

(d) 鍵管理

認証局ソフトウェア製品とテストツールにおいて最も大きな差異は鍵管理である。認証局ソフトウェア製品では鍵は厳重に管理されており、秘密鍵を外部に取り出すことはできない。一方、テストツールは公開鍵、秘密鍵を含めウェブブラウザから PEM 形式で参照することができる。これは実験用として秘密鍵が公開されているため、問題点の早期発見に役立つというメリットをもたらす。しかしながら、テストツールで生成された鍵や証明書や決して実運用で使用してはならないので注意が必要である。

(e) 結論

テストツールの持つ鍵ペアや証明書はあくまで実験用であり、実運用で利用しないよう注意する必要がある。パス検証テストに必要なイレギュラーなケースを含む多様な証明書、証明書失効リストを柔軟に発行できるという点において、パス検証テストの実証実験においてはテストツールを利用することが好ましい。

⑸ パス検証テストデータの再利用性

前年度実験で作成されたパス検証テスト用のデータは証明書、証明書失効リストおよびリポジトリファイル(LDIF)として保存されている。平成 15 年度は、平成 14 年度成果であるパス検証テストデータをテストツールにインポートし、また、平成 15 年度実験を行った DN matching、LDAPURI および UTF8 CJK に関するテストケースもまたテストツールに登録した。テストツールをテストデータ再利用性の観点から考察する。

(a) テスト環境構築の容易さとコスト

パス検証テストのためのサーバ環境には、認証局、ディレクトリサーバを最低限用意しなければならないが、テストツールサーバの場合、一台の Linux マシンで複数の認証局、複数のディレクトリサーバの役割を担うことができる。テストツールを構成するソフトウェアは全てオープンソースソフトウェアであ

り、RPM パッケージや make コマンドの実行により簡単にインストールすることができる。また、全て無償ソフトウェアで構成されておりサーバ環境構成するソフトウェアの準備にかかるコストはゼロである。

(b) パス検証テストに係わるサーバ環境の再構築

パス検証テストの平成 14 年度の実験データを他のホストで利用し、実験項目を追加しようとした場合、認証局およびディレクトリサーバの再構築、データの再投入といった甚大な作業負担を必要とする。特に cRLDistributionPoints、 issuingDistributionPoint、autorityInfoAccess に含まれるホスト名や GeneralName の IP アドレスなど、同じネットワーク環境を再現できない場合には一つ一つ証明書を修正し再発行することになり、その負荷は大きい。

平成 15 年度整備したテストツールのデータベースに登録されているテストデータを利用する場合､以下の手順によりサーバ環境を再現することができる。

- Linux 環境の構築およびツールのインストール
- データベースのダンプファイルの生成
- データベースダンプファイルにおいてホスト名、ディレクトリ名の置換
- 修正データをテストツールにインポート
- 証明書、CRL、LDIF データを一括生成
- リポジトリへ LDIF 登録

これら手順はバッチ処理も可能であり、テストツールサーバの構築に半日～1 日、データ修正と登録に 1～2 時間で環境を再現することができる。

(c) パス検証テストの対象国の拡大と成果展開

来年度以降 JKSTIWG への新規参加国や、他国、他団体などの組織が JKSTIWG の設計によるパス検証テストを実施したい場合、また既存の JKSTIWG メンバが将来同じテスト環境で再実験を行いたい場合、平成 14 年度のファイルデータのままでは環境の再現が難しい。証明書や証明書失効リストがホスト名を含む場合、これらを変更する必要があるからである。テストデータとしてテストツールからエクスポートされたデータベースのダンプによるデータを公開するならば平成 15 年度の成果を容易に利用することが可能である。

「図 14.8　バーチャルホスト機能を用いた 1 台による CC テスト環境」のように、Linux のバーチャルホスト機能を用いれば自ら構築した Linux マシン 1 台のみを用いてテストツールをインストールし､テストデータをインポートし､データ生成およびリポジトリ登録を行えば簡単に同じ環境が再現できる。

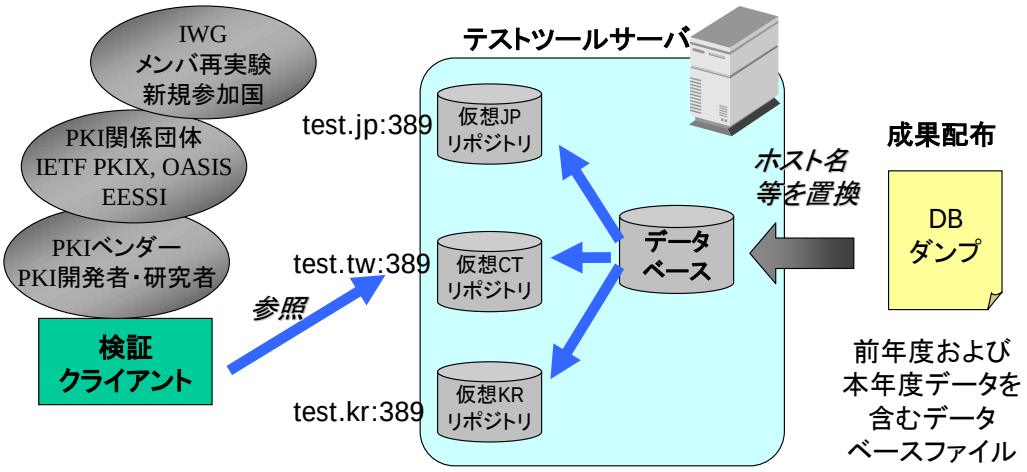


図 14.8　バーチャルホスト機能を用いた 1 台による CC テスト環境

新規参加国が参加国の認証局製品およびリポジトリ製品を用いて JKSTIWG 各国と CC モデルのテストを希望している場合、各国がそのような実験環境を提供することは負荷が高く現実的には難しい。このような場合、テストツールは相互認証もサポートしているため、「図 14.9　新規参加国と仮想 A、B 国との CC モデル実験環境」のように、各国認証局とリポジトリをシミュレートするテストツールサーバ環境を構築し、新規参加国と相互認証すれば、シミュレーション環境での CC モデルでのパス検証テストを実施することができる。

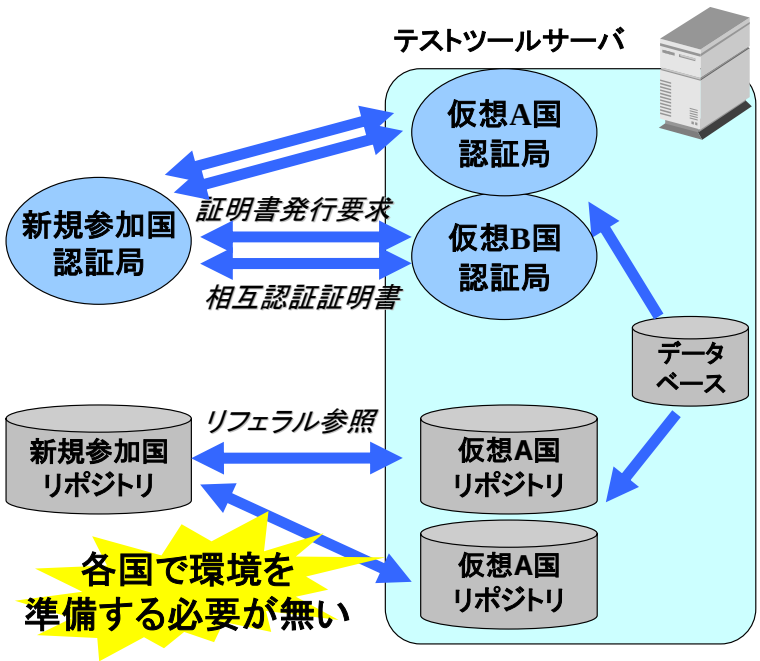


図 14.9　新規参加国と仮想 A、B 国との CC モデル実験環境

また、米国 NIST が公開している PKITS[19]のようにパス検証テストのための公開リポジトリやテストデータの参照、テストデータの配布サーバとして実験環境を公開することも可能である。

(d) 結論

平成 15 年度成果となるパス検証テストのテストデータは、データベースに保持されており、本データを修正し同じテスト環境を容易に再構築することができるため再利用性が非常に高いといえる。

⑹ 協調作業環境としてのテストツール

平成 15 年度追加されていたテストケースである UTF8 CJK に関するテストケースではチャイニーズ台北が主体となりテストケースの設計を行ったが、テストツールサーバへのデータ登録に関しては日本とチャイニーズ台北が協調して作業を行い、テストツール上の日本側リポジトリにデータを登録し実験を行った。実験結果に問題があった場合にはデータベースに登録されているデータをウェブブラウザでお互いに確認しながらデータの誤りやテストケースの設計ミスなどを発見した。パス検証テストにおけるテスト項目の設計や参照における協調作業環境としてのテストツールについて考察する。

(a) ウェブブラウザによるテストデータの編集・参照

証明書、証明書失効リスト、リポジトリ情報等のパス検証テストデータはデータベースにより管理されており、汎用のウェブブラウザから編集・参照することができる。これにより各国テストデータ設計者が協調してテスト設計を行うことができる。

(b) テストデータの追加変更および参照のアクセス権限の設定

あらかじめ定められたテストケース設計者といった権限のある各国のユーザがウェブブラウザにより協調しながらテストデータ追加、変更することができる。また、テストデータ参照の権限も設定することができる。

(c) 実験とテストデータ修正のターンアラウンドの短縮

テストデータは数が多く、データ作成の際、登録の誤りといったミスにより誤ったデータを作成してしまう可能性がある。実験データである証明書や証明書失効リスト、リポジトリを見なくともウェブブラウザで確認することにより問題解決する場合もあり、これをオンラインで修正することができるため、修

[19] http://csrc.nist.gov/pki/testing/x509paths.html

正から実験までのターンアラウンドを短縮することができる。
結果的に実験を効率的に進めることができる。

(d) 結論

国際的な PKI 相互接続実験において、テストデータを共有し、共同でテストケースを開発する必要がある場合、本テストツールのようなアクセシビリティの高い環境を利用することが望ましい。

(7) パス検証テストの実施環境に関する考察

パス検証テストガイドラインは、テスト実施環境としてパス検証テストツールの環境を前提としている。しかし、パス検証テストガイドラインの前提とは異なるテスト環境で、利用者がテストを実施する可能性がある。本項では、パス検証テストガイドラインの前提と異なった環境でパス検証テストを実施した場合の問題点を考察する。

(a) パス検証テストの実施環境

想定できるパス検証テストの実施例を、表 14.7　パス検証テストの実施例に示す。

表 14.7　パス検証テストの実施例

パターン	パス検証テストツールの使用	JKSTIWG が用意するもの	利用者が用意するもの	備考
0	有り	- ガイドライン - テスト環境(ツール)	- 検証モジュール	パス検証テストガイドラインで想定しているテスト環境。実証実験により、有効性を検証している。
1	有り (LDIF 生成機能)	- ガイドライン - LDIF	- 検証モジュール - リポジトリ	(例) 利用者がパス検証テストツールへアクセスできない環境でテストを実施したい場合(企業内 PKI 等)。
2	なし	- ガイドライン	- 検証モジュール - リポジトリ - 認証局(証明書発行ソフトウェア)	(例) 利用者が、既存の PKI を利用してパス検証テストを実施したい場合。

3	有り (パス検証テストツールからテスト用証明書を取得)	- ガイドライン - パス検証テスト用証明書	- 検証モジュール	(例)利用者が、検証モジュールについて、リポジトリへアクセスしてパス検証に必要な証明書を収集する処理を評価対象外とし、パス検証テストを行いたい場合。

パターン 1 は、利用者がパス検証テストツールのリポジトリ情報を LDIF 形式でエクスポートして、利用者側が準備したリポジトリにインポートするパターンである。例えば、利用者が企業内 LAN の内部でパス検証テストを実施したいが、ファイアウォールなどに妨げられてパス検証テストツールのリポジトリへアクセスできないときがこれにあたる。

パターン 2 は、利用者がパス検証テストガイドラインを参照して、利用者が準備した認証局から証明書を発行し、その証明書をリポジトリへ登録してパス検証テストを実施するパターンである。例えば、既存の PKI 環境を利用してパス検証テストを実施したいときがこれにあたる。この場合、パス検証テストツールが提供する環境(パターン 0)を基準と考えると、利用者はパス検証テストをパターン 2 で実施した結果と、パターン 0 で実施した結果を比較することで、パターン 2 で利用者が準備した環境と基準とする環境との間の差異を確認できる。

パターン 3 では、利用者がパス検証テストツールからパス検証テスト用証明書をダウンロードする。そして、証明書パスを構成する中間証明書及び CRL 等を検証モジュールに入力し、問題なくパス検証できるか確認する。つまり、パターン 3 では、検証モジュールが証明書及び証明書失効リストを集めるためにリポジトリへアクセスする必要がない。この場合、利用者は検証モジュールについて、リポジトリへアクセスして必要な証明書を収集する処理を除くパス検証処理を評価することができる。

(b) パターン 1 及びパターン 2 のテスト実施例における問題点の検討

パターン 1 及びパターン 2 のとき、以下のような問題が発生する可能性がある。

- パス検証テストガイドラインに記載している実験ができない。
  (実験実施が不可能な場合)
- 利用者が実験した際、パス検証テストガイドラインが示す実験結果期待値と違った結果が得られたとしても、その原因が証明書検証モジュールにあるとは特定できない。
  (実験結果の適切な評価が不可能な場合)

(i) 実験実施が不可能な場合

実験実施が不可能な場合の原因としては、以下のような理由を挙げることができる。

- 利用者が実験に使用する認証局が、パス検証テストガイドラインで規定しているプロファイルの証明書を発行することが出来ない。
- 利用者が実験に使用するリポジトリがパス検証テストで使う証明書の登録が出来ない。

(ii) 実証実験の適切な結果評価が不可能な場合

実証実験の適切な結果評価が不可能な場合の原因としては、以下のような理由を挙げることができる。

- 認証局が正しい ASN.1 構造の証明書を発行していない。[20]
- リポジトリにアクセスするときに、“;binary”オプションを付けた場合又は付けない場合のリポジトリの動作が RFC に準拠していない。[21]

(c) パターン 3 のテストの考察

パターン 3 のとき、前述したパターン 1 及びパターン 2 のような問題は発生しない。本パターンは既に認証パスの検証に必要な証明書及び証明書失効リストのファイルをローカルファイルとして取得しており、パス構築ではなく狭義のパス検証にフォーカスしたテストを行うことが可能である。

(d) 結論

パス検証テストツールを活用しない場合は、テスト結果を正確に評価することが出来ない可能性がある。よって、パス検証テストを実施する環境としてはパス検証テストツールを活用するべきである。

[20] 本実験報告書「14.2.1(3)証明書の ASN.1 構造及びエンコーディング方法に関する考察」を参照

[21]平成１４年度情報セキュリティ対策推進事業「電子商取引(ＥＣ)技術基盤の相互運用性に関する調査研究」実証実験報告書　実証実験編「18.2.1(5) (5) cRLDP(iDP)内の LDAPURI における attribute、及び binary オプションの有無」を参照

14.2.3 アプリケーションインターフェース仕様

(1) スロットを管理する PKCS#11 ライブラリについて

今回の実証実験の中で、日本から韓国アプリケーションを使用した検証において、日本クライアントのトークン中の鍵を韓国アプリケーションが読み出せないというエラーが発生した。その原因は、PKCS#11 ライブラリが管理するスロットの数の制限に関する認識が異なっていたことであり、共通仕様とするための大きな阻害要因となることから、PKCS#11 ライブラリが管理するスロットの数について考察する。

(i) 日本と韓国の双方のモデル

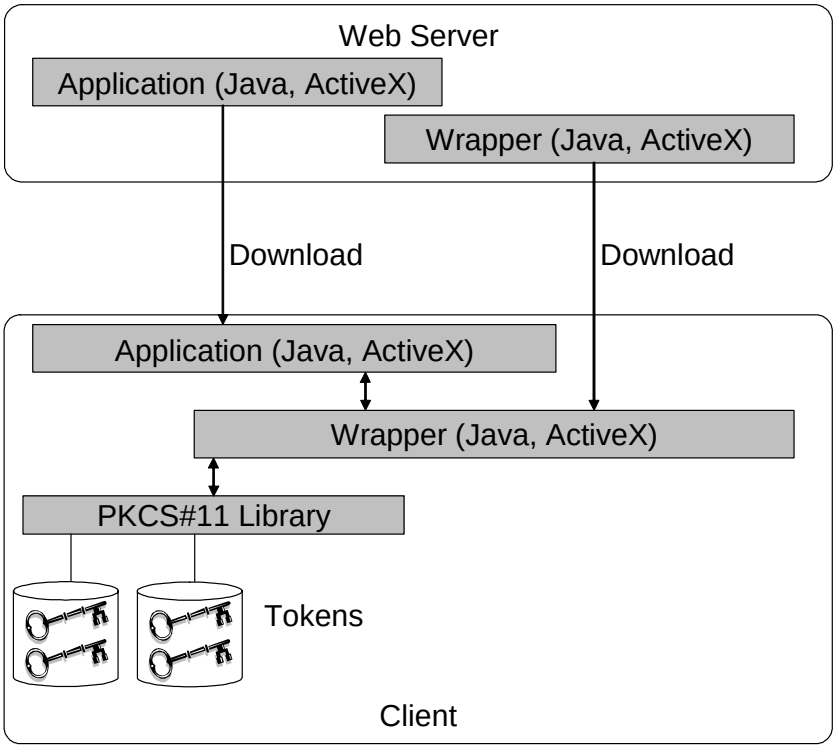


図 14.10　日本のアプリケーションモデル

日本のアプリケーションモデルを「図 14.10　日本のアプリケーションモデル」に示す。

日本のクライアント環境では、PKCS#11 ライブラリは、使用するアプリケーションサービスごとに異なることが前提である。

トークンは、利用者 1 人に 1 つが割り当てられ各個人が管理する。

ある 1 つのアプリケーションサービスを複数人が 1 台の端末から共通に利用することが想定されるため、1 つの PKCS#11 ライブラリが管理することが可能なトークンは複数に対応できるところが特徴である。

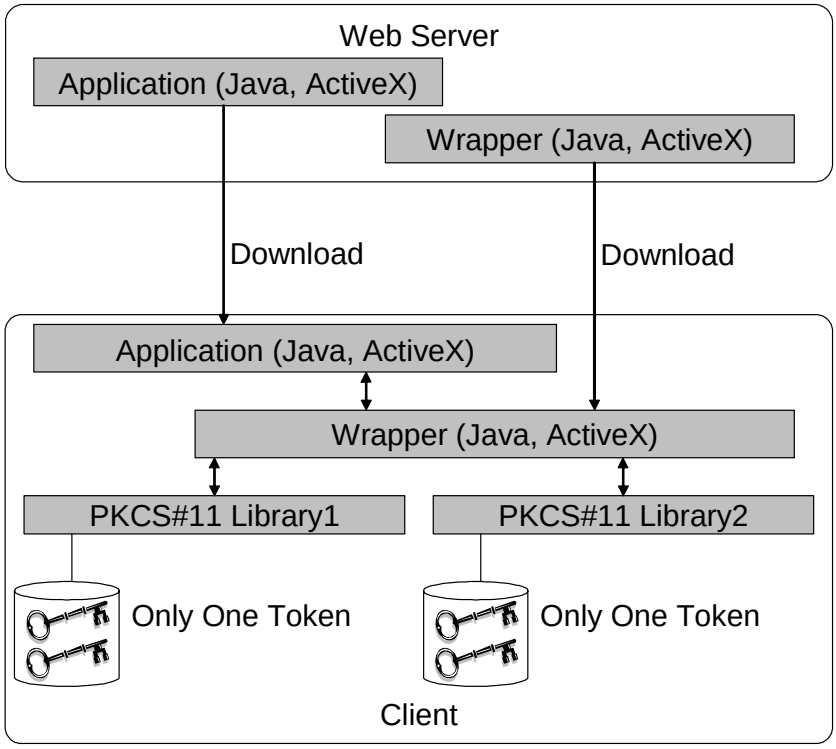


図 14.11　韓国のアプリケーションモデル

韓国のアプリケーションモデルを「図 14.11　韓国のアプリケーションモデル」に示す。

韓国のクライアント環境でも、PKCS#11 ライブラリは、使用するアプリケーションサービスごとに異なることが前提である。

トークンは、利用者 1 人に 1 つが割り当てられ各個人が管理する点も日本と同様である。

日本と異なる点は、ある 1 つのアプリケーションサービスを複数人が 1 台の端末から共通に利用することが想定されていない点である。従って、1 つの PKCS#11 ライブラリが管理することが可能なトークンは 1 つのみであることが特徴である。

(ii) 想定される利用場面

近年アジア圏の各国／地域でも一人一台の PC 環境が整備されてきているが、利用者がアプリケーションサービスを利用する場合の環境として、複数人が 1 台の PC を共通で利用するケースはまだ残っていると思われる。また、1 台の PC に複数のトークンを格納する場面として、社内のアプリケーションサービスを部門内の共用 PC から利用し、1 台の PC に利用者のトークンを複数格納するケースは十分に考えられる。

1 つの PKCS#11 ライブラリが 1 つのトークンしか管理できない仕様とすることは共用 PC からの利用を阻止するものであり、PKI を活用するアプリ

ケーションサービスの普及を妨げる結果にもなりかねない。

(iii) セキュリティ面

共用 PC を複数人で利用すること自体が、個人情報の漏洩等の原因となる可能性があり、セキュリティ面で問題があることは確かである。

焦点を絞り、1 つの PKCS#11 ライブラリで複数のトークンを管理する場合の鍵情報の漏洩についてセキュリティ面から考察する。

問題となるのは、一人の利用者があるアプリケーションサービスを利用している時、同じ PKCS#11 ライブラリ管理下の他のトークンに格納されている鍵情報を取得することが可能かどうかである。

トークンに格納される鍵情報には以下のものがある。

- 利用者の秘密鍵
- 利用者の秘密鍵とペアとなる公開鍵証明書

秘密鍵へアクセスするには、その利用者のみが知る PIN を入力して PKCS#11 ライブラリ経由で利用者のトークンへログインする必要がある。そのため、目的とする秘密鍵の PIN が漏洩しない限り秘密鍵が漏洩する危険性は無い。

一方、公開鍵証明書へアクセスするには、PKCS#11 ライブラリの C_findObjects 関数を使用することにより、利用者の PIN によるログインが無くても取得することが可能である。もともと公開鍵証明書は公開される性質の情報であるため、利用者以外の人に知られて問題になることは無い。また、公開鍵証明書から公開鍵を取り出し、そこから秘密鍵を割り出すことは事実上不可能であることから問題は無いと判断できる。そもそも、秘密鍵割り出しの危険性がある場合、PKI の信用性そのものが危なくなる。

以上のことから、1 つの PKCS#11 ライブラリで複数のトークンを管理していてもセキュリティ面での問題は無いと思われる。

(iv) 一般的な考え方

PKCS#11 の仕様書である「PKCS #11 v2.11: Cryptographic Token Interface Standard (RSA Laboratories) Revision 1 ¾ November 2001」では、アプリケーションモデルについて以下のように述べられている。

6.2 General model
Cryptoki's general model is illustrated in the following figure. The model begins with one or
more applications that need to perform certain cryptographic operations, and ends with one or

more cryptographic devices, on which some or all of the operations are actually performed. A
user may or may not be associated with an application.
（中略）
Cryptoki provides an interface to one or more cryptographic devices that are active in the
system through a number of “slots”. Each slot, which corresponds to a physical reader or other
device interface, may contain a token. A token is typically “present in the slot” when a
cryptographic device is present in the reader. Of course, since Cryptoki provides a logical view
of slots and tokens, there may be other physical interpretations. It is possible that multiple slots
may share the same physical reader. The point is that a system has some number of slots, and
applications can connect to tokens in any or all of those slots.
（後略）

図 14.12　アプリケーションモデルに関する記述

この記述によると、1 つの PKCS#11 ライブラリが複数のスロットを管理することを自然なことと考えており、更には、複数のアプリケーションが 1 つのスロットを共有することもあり得る。

1 つの PKCS#11 ライブラリが管理するスロットの数は、PKCS#11 の仕様から結論を出すというよりは、アプリケーションの運用要件から導かれるものと思われる。

(v) 結論

以上より、1 つの PKCS#11 ライブラリが複数のトークンを管理することについて問題はなく、PKI 及び PKI を活用するアプリケーションサービスの普及のためにも、1 つの PKCS#11 ライブラリが複数のトークンを管理することを可能とすべきであると考える。

ただし、この問題は、アプリケーションサービスの運用者における運用ポリシに依存するところが大きいため、アプリケーションインターフェース仕様の中では本件について規定せず、現状のままとする。

(2) 暗号化アルゴリズムについて

今回のアプリケーションインターフェース仕様における暗号化及び鍵のラップ処理で使用する暗号アルゴリズムとして、トリプル DES を使用した。PKCS#11 の中では、更に入力データ長の 8 バイトに満たない分を埋める（Padding）処理を行うかどうかで 2 種類の暗号アルゴリズムパラメータがある。

どちらの暗号アルゴリズムパラメータを使用すべきかを考察する。

(i) CKM_DES3_CBC について

CKM_DES3_CBC は、暗号化及び復号処理、鍵のラップ及びアンラップ処理においてトリプル DES のアルゴリズムを用いるための PKCS#11 のパラメータである。トリプル DES のアルゴリズムでは、処理対象となる入力データを 8 バイトの倍数の単位で処理するため、入力データ長が 8 バイトの倍数で無い場合、Padding 処理が必要となる。

このパラメータではアプリケーションが Padding 処理を行う必要があり、開発者の負荷及び実行時のアプリケーション負荷が高くなるというデメリットがある。特に Padding 処理を知らない開発者の場合、開発効率がかなり下がる危険性もある。一方で、CKM_DES3_CBC は多くの製品がサポートしていると思われるため、汎用性を高めるためにはこのパラメータを使用することがいいと思われる。

(ii) CKM_DES3_CBC_PAD について

CKM_DES3_CBC_PAD は、暗号化及び復号処理、鍵のラップ及びアンラップ処理においてトリプル DES のアルゴリズムを用いるための PKCS#11 のパラメータである。この点は CKM_DES3_CBC と同様であるが、このパラメータでは、Padding 処理を PKCS#11 ライブラリが自動的に補うため、アプリケーション開発負荷を下げるとともにアプリケーション実行時のアプリケーション負荷を下げることが可能であるメリットがある。アプリケーション開発者は、Padding 処理の実装方法を全く知らなくても効率的に開発を進めることが可能である。一方、PKCS#11 ライブラリの実装を行う際に Padding 処理も実装する必要があり、CKM_DES3_CBC のパラメータと比較すると実装されない可能性が高く、汎用性を損なう可能性がある。実際、今回の実験の中で、Padding 処理に対応していない国が 2 カ国あり、現状では汎用性を損なう可能性は高いと思われる。

(iii) アプリケーションで使用すべき暗号アルゴリズムパラメータについて

トリプル DES アルゴリズムを使用するに当たって、Padding 処理をアプリケーションと PKCS#11 ライブラリのどちらで行うべきかについて述べる。

今回の実験では、実験参加国の中に CKM_DES3_CBC_PAD をサポートし

ない PKCS#11 ライブラリを使用する国があったため、CKM_DES3_CBC を使用することとしたが、世の中の PKCS#11 ライブラリで CKM_DES3_CBC_PAD がサポートされないものがどれほどあるかは未知数であるため、汎用性を損なう可能性を推し量ることは困難である。

一方、PKCS#11 の仕様書である「PKCS #11 v2.11: Cryptographic Token Interface Standard (RSA Laboratories) Revision 1 ¾ November 2001」では、CKM_DES3_CBC_PAD について以下のように述べられている。

12.21.4 General block cipher CBC with PKCS padding
Cipher <NAME> has a cipher-block chaining mode with PKCS padding, "<NAME>-CBC with PKCS padding", denoted CKM_<NAME>_CBC_PAD. It is a mechanism for singleand multiple-part encryption and decryption; key wrapping; and key unwrapping with <NAME>. All ciphertext is padded with PKCS padding.
It has a parameter, an initialization vector for cipher block chaining mode. The initialization vector has the same length as <NAME>'s blocksize.
The PKCS padding in this mechanism allows the length of the plaintext value to be recovered from the ciphertext value. Therefore, when unwrapping keys with this mechanism, no value should be specified for the CKA_VALUE_LEN attribute.
In addition to being able to wrap and unwrap secret keys, this mechanism can wrap and unwrap RSA, Diffie-Hellman, X9.42 Diffie-Hellman, EC (also related to ECDSA) and DSA private keys (see Section Error! Reference source not found. for details). The entries in Table 99 for data length constraints when wrapping and unwrapping keys do not apply to wrapping and unwrapping private keys.

図 14.13 CKM_DES3_CBC_PAD に関する記述

これによると、Padding 処理に関する扱いが特別なものではなく、アプリケーション開発の効率化を図ることが可能な便利な機能として掲載されており、標準でサポートすることがいいと思われる。

(iv) 結論

Padding 処理に未対応な PKCS#11 ライブラリがどれだけあるかによるため、ここですぐに結論を出すことは時期尚早であると思われる。状況を見据えた上での対応が必要になる。アプリケーション開発者にとっては Padding 処理を認識している開発者は現状ではまだ少ないと思われる。この点では CKM_DES3_CBC を対応する仕様が最適であると思われる。しかし、将来

的には、アプリケーション開発のための仕様としては、開発効率を上げることが可能な CKM_DES3_CBC_PAD をサポートするべきである。

以上より、今回策定する仕様としては現状とおり CKM_DES3_CBC とする。

(3) 実験結果について

アプリケーションインターフェース仕様の有効性に関する実験の中で、幾つかエラーが発生した。アプリケーションインターフェース仕様をより有効なものとするため、発生したエラーの原因をアプリケーションインターフェース仕様へ追加すべきかどうかについて考察する。また、実験結果を受けて行ったヒアリングについて、機能追加に関する指摘があった。この指摘を今後のアプリケーションインターフェース仕様へ反映すべきかどうかについて考察する。

(a) 属性設定におけるアプリケーションバグエラーについて

鍵オブジェクト及び証明書オブジェクトで設定する属性について、その設定方法に問題があり、エラーが発生した。

これは、トークンに格納するオブジェクトの属性項目数が非常に多く、オブジェクトの種類によってサポートする属性項目が違う等設定が複雑である点が原因と思われる。今回、アプリケーションの属性設定方法を修正することで、テストを完了させている。

本件は、現在のアプリケーションインターフェース仕様の記述を見やすさを向上させるための修正が必要になる事項と思われる。今後検討を行い、仕様に反映させる必要がある。

(b) クライアント環境におけるソフトウェアバージョンの不整合に関するエラーについて

Java で開発されたアプリケーションでは､クライアント環境に対しても同じバージョンの Java 環境を要求する。今回、サーバ側アプリケーションとクライアント側実行環境で Java のバージョン不一致による動作異常が発生した。日本のアプリケーションは Java1.3.1 で構築しており、このアプリケーションに対して Java1.4.1 のクライアントからアクセスすると、画面が正常に表示されない等の顕著な不具合、或いは機能動作が正常に動作しない等の見えづらい部分での不具合が発生した。

今回、アプリケーションとクライアントの Java のバージョンを揃えることでテストを完了させている。

本件はアプリケーションの運用において規定されるべき事項であるため、アプリケーションインターフェース仕様の範疇では無い。従って、アプリケーションインターフェース仕様は現状のままとする。

(c) PKCS#11 ライブラリにおける管理オブジェクトの相違に関するエラーについて

トークンで管理するオブジェクトの相違によりエラーが発生した。

トークンで管理するオブジェクトには主に秘密鍵オブジェクト、証明書オブジェクト、公開鍵オブジェクトの 3 種類がある。この内、公開鍵オブジェクトに関する管理の相違からエラーが発生した。

具体的には、トークンで公開鍵オブジェクトが管理されていることを前提としたアプリケーションに対して、秘密鍵オブジェクトと証明書オブジェクトしか管理していないトークンを使用したクライアントがアクセスした場合、公開鍵を必要とする処理でアプリケーションは公開鍵を取得することができずにエラーとなる。

アプリケーションが公開鍵オブジェクトを取得する方法には以下の 2 とおりがある。

1) トークン中で公開鍵オブジェクトが管理されており、アプリケーションは直接公開鍵オブジェクトを取得する。
2) トークン中では秘密鍵オブジェクトと証明書オブジェクトのみが管理されており、アプリケーションは一旦証明書オブジェクトを取得した後、証明書から公開鍵を取り出し、公開鍵オブジェクトを作成する。

1）の場合、アプリケーション開発の効率化を図ることが可能であり、実行時のアプリケーション負荷を抑えることが可能であるメリットがある。一方、認証局が PKCS#12 形式で証明書を発行した場合、トークンに証明書を格納するのはアプリケーション又は利用者の作業となり、証明書から取り出した公開鍵の正当性を認証局が保証できなくなる危険性がある。

b）の場合、認証局が保証する証明書から公開鍵を取り出すことで、公開鍵の正当性を保つことが可能である。一方でアプリケーション開発の負荷とアプリケーション実行時の負荷を上げるデメリットがある。

PKI を活用するアプリケーションを構築する場合、PKI の信頼性を損なう運用は避ける必要があり、トークンで管理するオブジェクトは秘密鍵オブジェクト、証明書オブジェクトのみとすることが必要である。

今回、アプリケーションにおいて、証明書オブジェクトから公開鍵を取得し、公開鍵オブジェクトを作成する修正を行うことでテストを完了させている。

本件は、アプリケーションインターフェース仕様の中で規定すべき事項であると思われる。今後検討を行い、仕様に盛り込む必要がある。

(d) 機能追加に関する指摘について

今回のアプリケーションインターフェース仕様では、PKCS#11 を使用する

場合にアプリケーションの要件から必要となる機能を網羅することができた。

一方、「14.1.3(1)アプリケーションサービスの国際的利用における共通仕様の確立」で挙げたアプリケーションで必要となる 8 機能のうち今回盛り込めていない機能もある。特にその中で重要であり、ヒアリングからも必要機能として挙げられている署名者証明書の有効性検証とアプリケーション環境の正当性保証機能について考察する。

(i) 署名者証明書の有効性検証について

署名を検証する場合、署名者が作成したハッシュ値と検証者が作成するハッシュ値の比較による改ざん検出と併せて署名者の秘密鍵の正当性について署名者の公開鍵証明書を検証することにより間接的に検証する必要がある。

パス検証では、署名者と検証者の間の認証パスの構築とそのパスの有効性検証を行うが、アプリケーションインターフェース仕様の対象である PKCS#11 は証明書の有効性検証に関する機能を持っていない。この機能については CryptoAPI や Java の機能を使用することになると思われる。また、証明書検証に関する共通のアプリケーションインターフェースを策定し、ベンダー固有製品等を使用して証明書検証を行う方法も考えられる。

CryptoAPI や Java は証明書検証の機能を有しているが、検証項目については注意が必要である。

RFC3280 や日本の「政府認証基盤　相互運用性仕様書」で検証項目について述べられているが、CryptoAPI や Java はその全てを網羅しておらず、従って、必要となる検証項目に関する検討が必要である。更に、足りない検証項目がある場合、CryptoAPI や Java でどのように実現し、国際間の利用者でどのように共通化を図るかを検討する必要がある。

(ii) アプリケーション環境の正当性保証について

アプリケーション環境には大きく 2 つの要素がある。1 つはアプリケーションであり、もう 1 つはクライアント環境である。国際間で利用されるアプリケーションは Web ベースが想定され、アプリケーションとしては Java や ActiveX が利用されることが想定される。

アプリケーションの正当性を保証するためには、サーバ証明書によるサイト認証、Java や ActiveX のモジュールに対するコード署名という方法がある。

問題はクライアント環境である。アプリケーションを利用するクライアントの環境には以下のものがある。

- OS
- Web ブラウザ
- アプリケーション利用のためのミドルウェア

・ アプリケーションの環境設定ファイル
・ PKI ライブラリ等の PKI コンポーネント

これらのクライアント環境の正当性を保証するにあたり、以下のことが問題となる。
・ 誰がクライアント環境を保証するのか。
・ アプリケーションが保証するとした場合、どの範囲を保証すればいいのか。
・ アプリケーションが保証する場合、利用者による環境構築、利用開始から利用終了までのどのタイミングで保証するか。
・ どのような手段により保証するか。

これらのクライアント環境のうち、アプリケーションが責任を持つ必要がある箇所がある。

アプリケーションの環境設定ファイルは、利用者の状況に合わせてアプリケーションが作成するものであり、その内容が改ざんされていないことをアプリケーションが厳密に確認し、正当性を保証する必要がある。更には、アプリケーションで利用者が使用する PKI コンポーネントについてもアプリケーションが正当性を保証した方がいいと思われる。

このアプリケーションの環境設定ファイルと PKI コンポーネントについて、どのタイミングでどのような方法により正当性を保証するかについては相当な検討を要するためここで結論を出すことはできな。いくつかの概案を提示する。

(ｲ) アプリケーションの環境設定ファイルについて

環境設定ファイルをクライアント PC 内に持つ場合、1 台の PC で 1 つの環境設定ファイルがアプリケーションにより構築される。複数人の利用者が 1 台の PC を共通に利用する場合でも、環境設定ファイルに更新があった時点でアプリケーションがアプリケーション自身を証明できる証明書により電子署名を付与することで環境設定ファイルの改ざんを検出することが可能となり、正当性を保証することが可能となる。

環境設定情報をアプリケーション（サーバ側）で管理することも可能である。利用者認証により利用者が使用する環境情報を特定する方法をとることで、環境設定ファイルがクライアントで必要なくなり、環境設定ファイルに対する署名付与が不要となる。クライアント環境は単純化されるが、アプリケーションの構造が複雑になるデメリットもある。

(ﾛ) PKI コンポーネントについて

PKCS#11 ライブラリ等の PKI コンポーネントについては、利用者或いは PC 管理者が正当性を保証するものであり、アプリケーションが正当性を保証するものでは無いという考え方もある。

アプリケーションが正当性を保証する場合、使用する PKI コンポーネントに対してアプリケーションが署名を付与する等の方法をとることになるが、1 つの PKI コンポーネントを複数のアプリケーションが共通で使用する場合、署名を付与するという方法は単純には成り立たなくなる。具体的な方法は見つけづらい。

以上から、クライアント環境の保証については、アプリケーションインターフェース仕様の範疇を大きく超える箇所があり、必要性や方法についての世界レベルでの検討や標準化が必要であると思われる。