2E-02

IoTデバイス向け証明書発行基盤技術の設計と試作

小寺　志保†　　南　圭祐†　　安次富　大介†

(株)東芝　研究開発センター　ネットワークシステムラボラトリー†

1.はじめに

IoTの普及に伴い，証明書を使ってデバイスをセキュアに管理するシステム，技術が議論されている．例えば[1]では，各デバイスに予め埋め込んだ，デバイス単位で一意のブートストラップ用の証明書（IDevID: Initial Device Identifier [2]）でデバイスを認証し，システム内でセキュアな通信を行うための有効な証明書（LDevID: Locally Significant Device Identifiers [2]）を各デバイスに動的に発行する仕組みを定めている．

一方，IoTの文脈とは異なるが，パスワードレスのユーザ認証規格などを定めるFIDO Alliance [3] ではプライバシーの観点から，ブートストラップ時のデバイス(認証器)の認証にデバイス単位ではなく，製品単位で発行した証明書を用いている．実際のIoT向けサービスでも，ホームネットなどの個々のデバイスがユーザと密接に結びつくものが多く，デバイスのセキュリティだけでなく，プライバシーも考慮する必要がある．

そこで本稿では，証明書発行処理におけるプライバシーを考慮した，IoTデバイス向けの証明書発行基盤技術を提案する．具体的には，IoTデバイスに対し，製品単位の証明書をブートストラップに用いて，デバイス単位の証明書を動的に払い出す方針を採用し，その設計と試作について述べる．

2.IoTデバイス向け証明書発行基盤の概要

本章では，IoTデバイス向け証明書発行基盤の概要を述べる．提案するIoTデバイス向け証明書発行基盤は，証明書管理サーバとデバイス管理サーバで構成される．証明書管理サーバはリクエストに応じて証明書を発行する．一方，デバイス管理サーバは製品やデバイス等の情報を登録・管理する．図1にIoTデバイス向け証明書発行基盤の利用フローを示す．図1のProductCertは製品単位の証明書，DeviceCertはデバイス単位の証明書である．

①デバイス製造元は証明書発行と製品情報登録をデバイス管理サーバにリクエストする．
②デバイス管理サーバは，証明書管理サーバに証明書をリクエストする．
③証明書管理サーバはProductCertを発行する．
④デバイス管理サーバはProductCertと製品情報を対応付けて登録し，デバイス製造元に応答を返す．
⑤デバイス製造元は，該当する製品のデバイスにProductCertを埋め込む．
⑥デバイスは証明書発行と自身の登録をデバイス管理サーバにリクエストする．このとき，ProductCertを認証情報として使う．
⑦デバイス管理サーバは，証明書管理サーバに証明書をリクエストする．
⑧証明書管理サーバはDeviceCertを発行する．
⑨デバイス管理サーバはDeviceCertとデバイス情報を対応付けて登録し，デバイスに応答を返す．
⑩デバイスは発行されたDeviceCertを認証情報として，デバイス管理サーバと通信する．

3.API設計

本章では2章で述べた図1の各処理を設計する．図1の①，④，⑥，⑨に示した処理は

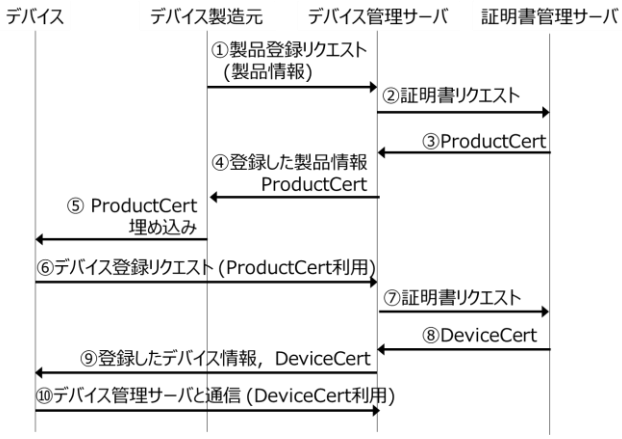

図1：IoTデバイス向け証明書発行処理フロー

"A Design and a Prototype of a Certificate Issuance to IoT Devices", Shiho Kodera†, Keisuke Minami†, Daisuke Ajitomi†
† Network System Laboratory, Corporate Research & Development Center

表 1：証明書発行 API の概要

POST products (POST /v1/products)	
リクエスト	製品情報
認証	ユーザ認証
レスポンス(201)	製品情報， ProductCert，秘密鍵
POST devices (POST /v1/devices)	
リクエスト	デバイス情報，CSR
認証	ProductCert による認証
レスポンス(201)	デバイス情報，DeviceCert

REST API で実現する．表 1 に設計した API を示す．POST products は製品情報の登録処理を実行する．リクエストが①，レスポンスが④に該当する．POST products のリクエストは Web 画面を通じてユーザ(製品製造元の管理者等)が実行することを想定しており，製品情報を入力とする．リクエストの認証はユーザ認証を用いる．証明書発行に使う CSR（ Certificate Signed Request ）と秘密鍵はデバイス管理サーバで発行する．レスポンスには，登録した製品情報，ProductCert，秘密鍵を含める．一方，POST devices はデバイスの登録処理を実行する．リクエストが⑥，レスポンスが⑨に該当する．POST devices のリクエストは個々のデバイスが実行することを想定しており，CSR，秘密鍵共に各デバイスで発行する．入力はデバイス情報と CSR とする．リクエストの認証には ProductCert を用いる．レスポンスには，登録したデバイス情報と DeviceCert を含める．

図 1 の②，③，⑦，⑧に示した処理は，証明書発行に関わる処理であり，EST（ Enrollment over Secure Transport ）[4] で実現する．EST は HTTP（ HTTPS ）経由でクライアント証明書を自動発行するプロトコルである．証明書発行後，デバイス管理サーバは製品・デバイスの ID と，発行した証明書の証明対象の DN（ Distinguished Name ）を対応付けて登録する（図 1 の③，⑧）．ID，DN は一意の値である．

4.実装システム

本章では，3 章で設計した処理の実装について述べる．本稿の実装では，すでに実装されている PSK（ Pre-Shared Key ）ベースのデバイス管理サーバに，3 章で設計した処理を追加した．図 2 に実装したシステム全体を示す．システムは 3 章までに述べた IoT デバイス向け証明書発行基盤と，認証・分散サーバで実現する．認証・分散サーバは，デバイス管理サーバへの負荷分散と，クライアント証明書による認証への対応のために追加した．詳細は後述する．

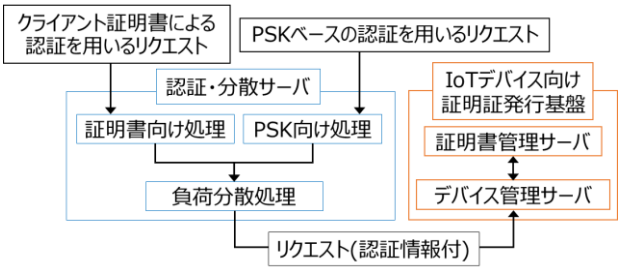

図 2：実装したシステム

証明書管理サーバには，3 章で設計した EST による証明書発行処理を実装した．デバイス管理サーバには，3 章で設計した API を実装し，各証明書と製品 ID，デバイス ID を対応付けてデータベースに登録する処理を追加した．

認証・分散サーバでは，リクエスト負荷分散に加え，リクエストを認証する方法に合わせた認証処理を行う．本稿では，認証・分散サーバに Nginx [5]を用いた．リクエストを認証する方法には，既存の PSK ベースのものと，新たに追加したクライアント証明書ベースのものがある．PSK ベースの認証を用いる場合，リクエストは PSK 向け処理を通り，リクエストの認証情報をそのままヘッダに格納する．一方，クライアント証明書ベースの認証を用いる場合，リクエストは証明書向け処理を通る．証明書向け処理では，クライアント証明書によってリクエストを認証し，認証結果を認証情報としてヘッダに格納する．証明書向け処理によって，負荷分散処理時の TLS 通信の終端にも対応できる．デバイス管理サーバはリクエストを受け取ると，認証情報のヘッダを確認し，認証する．

5.おわりに

本稿では，IoT デバイス向けの証明書発行基盤について述べた．IoT デバイスを証明書によって認証することはセキュリティの面で効果が見込めるが，IoT デバイスがユーザと密な関係にあることを踏まえるとプライバシーの問題が生じる恐れがある．そこで，IoT デバイスのブートストラップ時は製品単位の証明書を使い，IoT デバイスごとにシステム内でのみ有効な証明書を動的に発行する手法を検討した．

参考文献

[1] “Bootstrapping Remote Secure Key Infrastructures (BRSKI)”, draft-ietf-anima-bootstrapping-keyinfra
[2] IEEE 802.1AR, “Secure device identity”
[3] FIDO Alliance, https://fidoalliance.org/
[4] “Enrollment over Secure Transport”, RFC 7030
[5] Nginx, http://nginx.org/