一般
投稿論文

Webアプリからの遠隔操作に対応したスマートTV向け不正操作防止技術

磯崎 宏[†1]　金井 遵[†1]

[†1] (株) 東芝

ネット接続機能と，アプリを自由にダウンロードして実行する機能を備えたデジタルテレビ，いわゆるスマートTVが普及しつつある．スマートTVの操作機能をスマートフォンやタブレットのようなモバイル端末のWebアプリから操作できるように公開し，ネットワークを経由して操作できるように構成すれば利便性の向上だけでなく，Webサービスと連動したさまざまなアプリの実現が期待できる．一方で，悪意あるWebアプリから不正にスマートTVを操作される危険性もある．そこで本稿ではスマートTVの不正操作を防止するセキュリティアーキテクチャを提案した．提案アーキテクチャでは，モバイル端末上で動作するWebアプリに証明書を付与し，証明書の検証が成功したWebアプリにのみスマートTVの操作権限を与えることで，スマートTVの操作を正当な手続きを踏んで開発したWebアプリに制限することができるため，ユーザは各自のモバイル端末から快適に安心してスマートTVを楽しむことが可能になると期待できる．

1. はじめに

ネット接続機能と，アプリを自由にダウンロードして実行する機能を備えたデジタルテレビ（DTV），いわゆるスマートTVが普及しつつある．スマートTVでは画面操作が複雑化しており，赤外線リモコンを使った従来のボタン操作では限界がある．そこで，タブレットやスマートフォンなどのモバイル端末を使い，リモコン代わりに利用できれば，使い勝手の向上が期待できる．ところが，モバイル端末ではAndroid™ ☆1やiOSといったように実行環境が多様化しており，アプリ開発者は実行環境ごとにアプリを開発しなければならず，開発コストが高くなってしまう．そこでアプリをHTML5（HyperText Markup Language 5）やCSS3（Cascading Style Sheets level 3），JavascriptのようなWebアプリで実現すれば，実行環境に依存せず統一的に開発することができる．さらに多くのWebブラウザではWebSocketと呼ばれる通信機能をサポートしているため，DTV上にWebSocketサーバを実装し，モバイル端末のブラウザで実行されるWebアプリからネットワーク経由で操作コマンドを送信すれば，WebアプリからDTVの操作が実現できる．

しかし，ここで問題となるのはセキュリティである．従来技術では，プロプライエタリな通信プロトコルを独自に定義してDTVとの通信方式を非公開とするか，あるいはWebアプリから操作可能なようにDTVの機能を公開していたとしても，DTVの機能にアクセス可能なWebSocketサーバへの接続をローカルなWebアプリに制限したり，パスワードによるユーザ認証機能を持たせたりしていた．しかし，WebSocketサーバの接続をモバイル端末などの外部端末に公開した場合，Webアプリは単なるWebコンテンツと区別がつかず，一般のWebコンテンツからもDTVが操作可能となってしまうため，モバイル端末のブラウザでWebコンテンツを閲覧しただけでDTVが意図せず操作されてしまう危険性がある．

そこで本稿ではWebアプリからスマートTVを操作する上で生じるセキュリティ上の脅威を整理し，スマートTVの操作を正規のWebサーバからダウンロードされたWebアプリに制限するセキュリティアーキテクチャ“Double-RACLITE™”について提案する．

2. 想定するモデル

モバイル端末上のWebアプリからWebSocketによってDTVを操作する際の想定モデルを述べる（図1）[1].

まずユーザが，モバイル端末の実行環境に固有のアプリ配布マーケットから各自のモバイル端末にモバイルアプリをダウンロードし，インストールする（手順1）．ユーザがそのモバイルアプリを起動すると，モバイルアプリは内部的にブラウザを起動させ，Webアプリを実行

☆1 Androidは，Google Inc.の商標または登録商標．

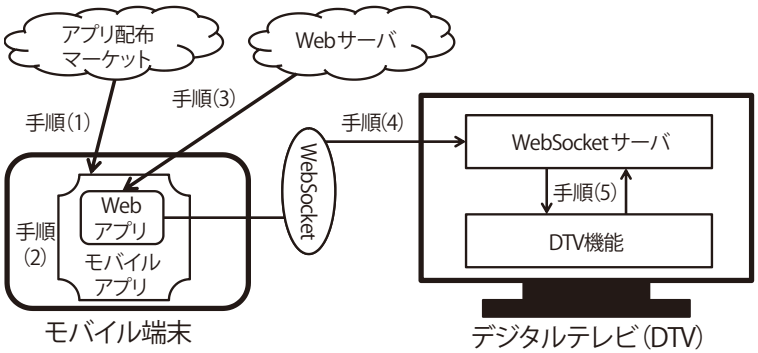

図 1　想定モデル

する（手順2）．Webアプリはモバイルアプリに含まれていてもよいし，動的にWebサーバから取得してもよい（手順3）．ユーザはモバイル端末上のWebアプリからWebSocket経由でDTVを操作する（手順4）．DTVではWebSocketサーバが動作しており，WebSocketサーバを介してDTV機能に操作を要求したりテレビ機能からの通知を受けとったりする（手順5）．

このように構成することで，ユーザの所有するモバイル端末に幅広く対応することができるほか，Webサーバ上のWebアプリを更新するだけでWebアプリの機能追加や機能変更を行うことができるためユーザがアプリをアップデートする手間を省くことができる．

3. セキュリティ上の脅威

3.1　不正アプリにより想定される被害

DTVの機能を外部端末から操作できるようにWebSocketサーバとして公開することで，操作性の向上やWebコンテンツとの連動ができるようになる反面，悪意あるWebアプリによってさまざまな問題を引き起こす危険性がある．

ユーザに被害を与える具体的な不正アプリの例を以下に示す．

- ボリューム操作・コンテンツ削除アプリ：DTVのボリューム調節をモバイル端末から操作できる機能をDTV機能が提供したとする．悪意あるWebサイトの所有者はいたずら目的でボリュームを最大あるいは最少にする命令を含めたWebサイトを構築し，そのWebサイトを訪れたユーザのDTVのボリュームを変更してしまうかもしれない．また，HDDを内蔵し，テレビ番組を録画する機能を備えたDTVが普及している．HDDに蓄積されたコンテンツの削除命令をDTV機能が提供した場合，悪意あるWebサイトの所有者はいたずら目的でDTVの内蔵HDDに蓄積されたすべてのコンテンツをユーザの確認なしに消去する命令を含め，そのWebサイトを訪れたユーザのコンテンツを消去させてしまうかもしれない．さらに，録画予約機能を提供した場合，いたずら目的で大量の録画予約を行って内蔵HDDをあふれさせ，本来ユーザが予約したい番組の録画をブロックさせてしまうかもしれない．
- リモート操作アプリ：DLNAガイドライン[2]に準拠し，家庭内ネットワークに接続されたレコーダと連携する機能を備えたDTVが普及している．DLNA機能をWebアプリから利用可能にするためにDTVがWebSocketサーバとしてDLNA機能を公開すれば，家庭内ネットワークに接続されている機器からコンテンツリストを収集し，コンテンツの蓄積場所に依存せずシームレスにDTVで再生可能なリストを表示するWebアプリを実現することができ，ユーザ利便性の向上が期待できる．一方，悪意あるWebアプリはDTVだけでなくDLNAで接続された機器まで不正に操作してユーザを混乱させるかもしれない．このようにDTV以外のほかの機器に二次被害を与える危険性がある．
- プライバシ情報取得アプリ：現在視聴中のチャンネル名や，内蔵HDDに蓄積されたコンテンツリストをモバイル端末から取得できる機能をDTV機能が提供すれば，現在視聴中のチャンネルに関連したWebコンテンツを表示するといったようにモバイル端末とDTVの連携サービスを提供することができる．一方，悪意あるWebサイトの所有者は，それらの情報を収集する命令を含め，そのWebサイトを訪れたユーザの許可なく，ユーザがどのような番組に興味があるのかといったユーザの嗜好情報を勝手に収集するかもしれない．

3.2　被害をもたらす技術的要因

DTVやレコーダの機能を外部端末から操作する機能を備えた製品は従来から存在しており，上述した脅威は以前にも存在する可能性はあった．しかし，実行環境に依存したアプリがプロプライエタリなプロトコルで通信しDTVを操作する従来技術と比較し，アプリがWebアプリとなると固有の問題が発生するため，セキュリティ上の脅威は増すと考えられる．その技術的な要因を以下にまとめた．

- ブラウザからも実行可能：Webアプリは単なるWeb

コンテンツである．想定モデルでは利便性の観点により専用のモバイルアプリからWebアプリを起動しているが，技術的にはブラウザからWebアプリを直接実行可能である．もし不正なWebコンテンツの中にDTVの操作命令が含まれていた場合，ユーザが通常のWebブラウジングをしている最中に意図せずDTVを制御される危険性がある．ユーザは，Webサイトに悪意ある命令が含まれているかどうかを事前に知る方法もなく，攻撃者はユーザに対して不正な命令が含まれたWebサイトであることをわざわざ知らしめる必要もないため，どのWebサイトに不正な命令が含まれているか知ることも困難であり，いつ問題が起きたのかを突き止めることもできない．ユーザは攻撃に気が付かずDTVの故障によって誤作動したのではないかと誤解する危険性もある．

- アプリの実行可否判定が困難：PCやAndroid™などのモバイルOSにはアプリのインストール時や起動時にアプリの開発元やアプリに付与する権限をユーザに明示的に確認させるアクセス制御の機能が備わっている．一方，WebアプリではURLの遷移でアプリが切り替わり，インストールや起動という明示的な操作が行われないため，ユーザが実行可否の判定を行う機会がない．
- 不正アプリが容易に作成可能：デスクトップアプリの場合，アプリは実行バイナリとして配布されるため，処理内容を解析したり通信プロトコルを解析したりするには一定以上のスキルが必要だが，Webアプリはスクリプト言語で記述されているためソースコードを容易に解析することができてしまう．Webアプリを採用することでアプリ開発が容易になった反面，攻撃者にとって悪意あるWebアプリを作成する敷居が低くなっている．

第2章で述べたWebアプリからDTVを操作するモデルを実現するには，これらWebアプリ固有の課題を解決する手段を提供する必要がある．

また，スマートTVにはサーバからアプリをダウンロードし，DTVにインストールしてDTVの機能やサービスを選択的に拡張できるものもある．本稿ではWebアプリからDTVをネットワーク経由で操作する際のセキュリティ脅威に着目しており，DTVにアプリを直接インストールする際に生じる脅威はスコープ外とする．なお，不正なアプリがDTVにインストールされることを防止するために，サーバに登録するアプリを認証されたものに限定したり，アプリに署名を付与してインストール時に検証したりする対策が施されている．

4. システムアーキテクチャ

4.1 全体システムアーキテクチャ

図2に全体システムアーキテクチャを示す．基本的な考え方は，ユーザが許可した正規Webアプリにのみ DTVの機器操作を許可する証明書（DTV操作トークン）を与え，Webアプリは機器操作命令の送信に先立ちDTV操作トークンを使ってDTVと認証処理を行う．DTVは認証が成功したWebアプリからのみ機器操作命令を受け付けるというものである．以下に主要コンポーネントの概要を示す．

- DTV操作トークン発行者
  DTV操作トークンを管理しており，アプリ開発者からの要請に基づいてDTV操作トークンを発行する機関．
- アプリ開発者
  モバイルアプリとWebアプリを実装する開発者．モバイルアプリはプラットホーム固有のアプリ配布マーケットに，WebアプリとDTV操作トークンはWebサーバにアップロードする．
- モバイルアプリ
  プラットホーム固有のアプリ配布マーケットからダウンロードしてインストールするアプリケーション．ブラウザを起動し，後述するStartupURLで示されるWebアプリを実行する．モバイルアプリはプラットホーム固有の方式で署名による改ざん防止

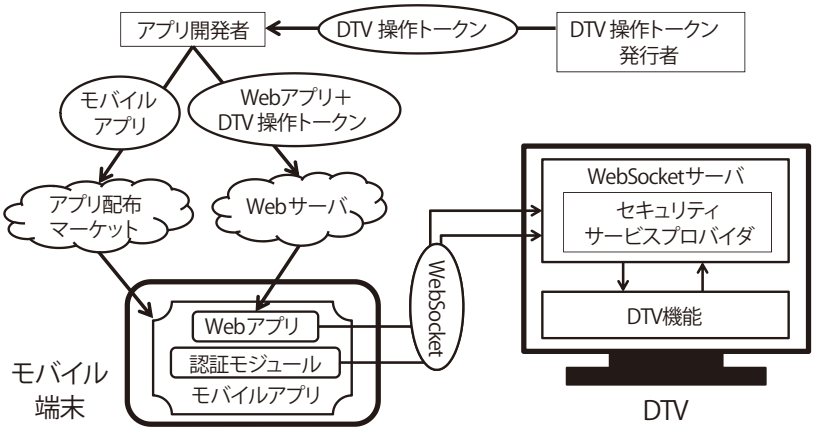

図2　全体システムアーキテクチャ

や，インストール時・バージョンアップ時のユーザ確認を行い，セキュリティを確保しているが，本稿ではそれらの仕組みをそのまま流用することを想定している．

- 認証モジュール
  Webアプリが DTV と認証処理を実行するために必要なモジュール．モバイルアプリとともに配布される．認証処理については後述する．
- セキュリティサービスプロバイダ
  Webアプリから WebSocket 経由で送信される機器操作命令の実行に先立ち，モバイル端末の認証モジュールと認証処理を実行し，以降に送信されてくる機器操作命令が正当か否かを判断するモジュール．機器操作命令を送信してくる Web アプリが正当なものであるかを DTV 操作トークンで判断する処理も行う．DTV の WebSocket サーバ内に組み込まれる．

4.2　処理フェーズ

次に，本アーキテクチャの動作を時系列で説明する．本アーキテクチャは以下の5つのフェーズから構成される．

(1) アプリ開発フェーズ（図3）
アプリ開発者はまず Web アプリのトップページの URL である StartupURL を DTV 操作トークン発行者に通知する（手順1）．DTV 操作トークン発行者は Web アプリ固有 ID（APPID），Origin，Permission を生成し，これらの情報に対してトークン署名鍵で公開鍵署名を施した DTV 操作トークンを生成し，アプリ開発者に送付する（手順2）．Origin とは StartupURL のうちドメイン名の部分を切り出した文字列である．Permission については後述する．アプリ開発者はアプリを開発する際に，DTV 操作トークン発行者に通知した StartupURL でモバイルアプリから Web アプリが実行されるように構成しておく必要がある．また，Web アプリは手順2で取得した Web アプリ固有 ID を Web アプリから参照可能なように構成しておく必要がある（手順3）．そして，モバイルアプリをアプリ配布マーケットに登録する（手順4）．さらに Web アプリと DTV 操作トークンを Web サーバにアップロードする（手順5）．なお，Web アプリは手順1で申告した StartupURL で示される URL に配置する必要がある．

(2) DTV セットアップフェーズ
ユーザが赤外線リモコンを用いて DTV を操作し，数字と文字列からなるペアリングコードを DTV に登録する．このペアリングコードはモバイルアプリから DTV にアクセスを許可する際のパスワードとして利用される．ペアリングコードのサイズにアーキテクチャ上の制約はないが，推測困難とされる長さと複雑さを要求することが望ましい．また，ユーザが初期値の変更を怠ったときに備え，工場出荷時に DTV ごとに別々の値が割り当てられていることが望ましい．

(3) モバイルアプリインストールフェーズ
ユーザがプラットホーム固有のアプリ配布マーケットからモバイルアプリをダウンロードしてモバイル端末にインストールする．

(4) アプリ初回起動フェーズ（図4）
ユーザがモバイルアプリを初めて起動する際の処理である．ユーザがモバイルアプリを起動すると，モバイルアプリはブラウザを呼び出して，前節で述べた StartupURL で示される場所から Web アプリを取得する（手順1）．まずこの Web ページで DTV 操作トークンがブラウザの Cookie として保存されているかどうかを確認する．初回起動時には DTV 操作トークンは存在しないため，StartupURL で示

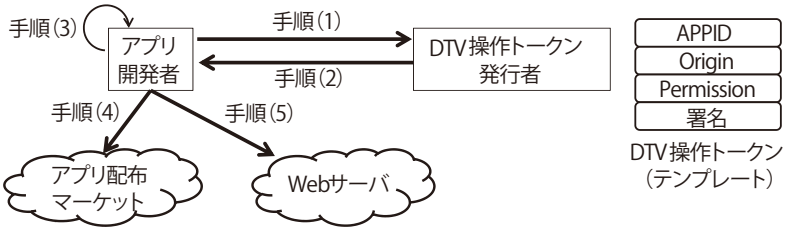

図3　アプリ開発フェーズの各ステップ

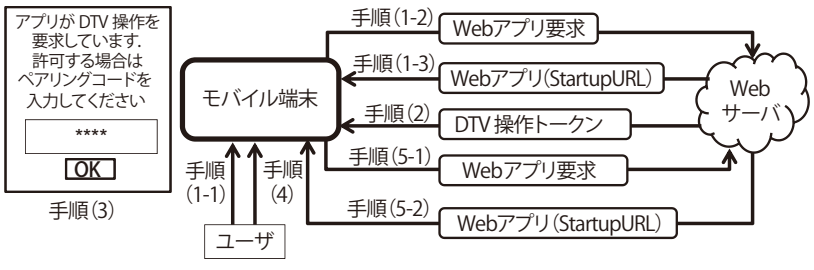

図4　アプリ初回起動フェーズの各ステップ

されるWebページはWebアプリ配布サーバからDTV操作トークンを取得し，Cookieに保存する（手順2）．さらに，このWebページでペアリングコードを入力する画面を表示させ，ユーザにモバイル端末からペアリングコードを入力するよう促す（手順3）．ユーザがペアリングコードを入力する（手順4）と，そのアクションとしてペアリングコード入力ページは再びStartupURLに遷移する．このとき，ペアリングコードの値がURLフラグメント識別子によってStartupURLに受け渡される．StartupURLで示されるWebページはペアリングコードをCookieとして保存する（手順5）．なお，URLフラグメント識別子のため，ペアリングコードはネットワーク上を送信されることはなくペアリング入力ページからStartupURLで示されるWebページに受け渡すことができる．

(5) DTV操作フェーズ

ユーザがWebアプリからDTVを操作するフェーズである．まず，Webアプリはアプリ初回起動フェーズでCookieに保存したDTV操作トークンとペアリングコードを取得し，認証モジュールに対してDTVと認証処理を行うように指示する．認証モジュールはDTVとWebSocketのコネクションを確立して認証処理を行う．認証処理については後述する．認証処理が成功すると，WebアプリはDTV操作命令をDTVに送信する．DTVは受信したDTV操作命令からDTV機能を呼び出す．

最後にモバイルアプリの認証モジュールとDTVのセキュリティサービスプロバイダ間で行われる認証処理について述べる（図5）．

まずWebアプリは認証モジュールにDTVと認証処理を行うようにJavaScript APIで指示する（手順1）．この際，CookieからDTV操作トークンとペアリングコードを読み出し，認証モジュールに渡す．認証モジュールはDTVとWebSocketコネクションを確立し，認証要求とともにDTV操作トークンを送信する（手順2）．DTVのセキュリティサービスプロバイダはDTV操作トークンの署名を検証する．検証が成功した場合に限り，チャレンジ乱数を生成して認証モジュールに送信する（手順3）．認証モジュールは認証要求を呼び出したWebアプリのOriginがDTV操作トークンに含まれるOriginと一致するか検査する（手順4）．検査が成功した場合に限り，受信したチャレンジ乱数とWebアプリから受け取ったペアリングコードと鍵からワンタイムトークン（OTT）を生成し，Webアプリにその値を渡す（手順5）．WebアプリはOTTをDTVに送信する．DTVのセキュリティサービスプロバイダは自身でチャレンジ乱数，ペアリングコードと鍵からOTTを生成し，送信されてきた値と一致するか確認し，その結果をWebアプリに返す（手順6）．Webアプリは認証が成功したと判断し，DTV操作命令をDTVに送信する（手順7）．セキュリティサービスプロバイダはOTTの検証が成功した場合に限り，DTV操作命令を受け付ける（手順8）．なお，OTTはWebSocketコネクションが接続されている限り有効であるため，WebアプリはWebSocketコネクションが切断されるまで認証処理を行う必要なく繰り返しDTV操作命令を送信することができる．

4.3 DTV操作トークンとペアリングコード

DTV操作トークンは，DTVを操作するWebアプリが正規の開発者によって正規のWebアプリ配布サーバから配信されたものであることをDTVに伝えるための情報である．DTV操作トークンには以下の情報が含まれる．

- アプリ固有ID（APPID）
  DTV操作トークン発行者が割り当てるWebアプリ固有のID．
- Origin

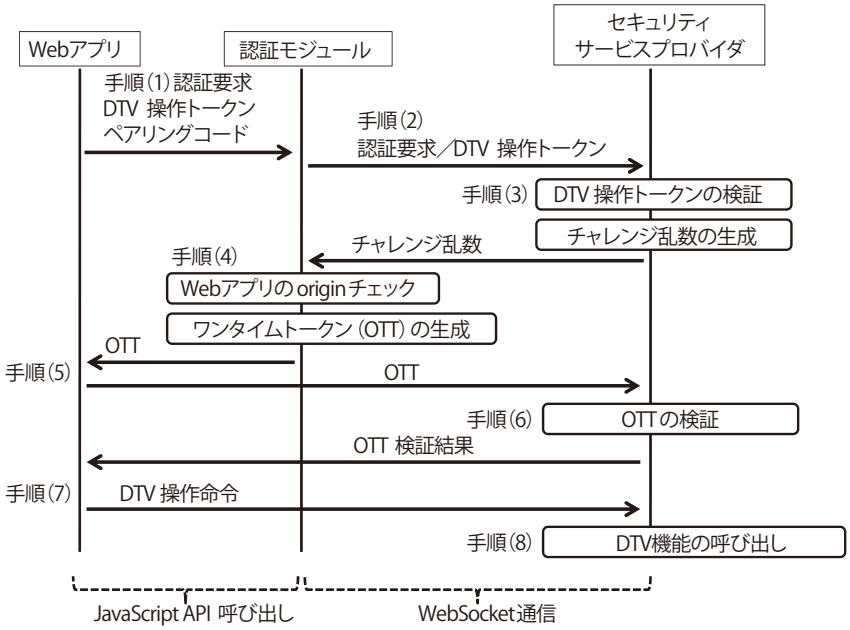

図5　認証処理の各ステップ

Webアプリの StartupURLのドメイン名.

- Permission
  Webアプリに許可されたDTV操作命令のセット．DTV操作命令をカテゴリ分けし，危険度に合わせたレベルを設定して利用可能なWebアプリを制限することができる．たとえばDTVを操作するAPIとしてチャンネル変更命令，ボリューム変更命令，録画予約命令が定義されていた場合，APPIDが1番のWebアプリはチャンネル変更命令とボリューム変更命令を許可するが，APPIDが2番のWebアプリはチャンネル変更命令のみ許可するといったことが定義できる．
- 署名
  APPID，Origin，Permissionに対する署名．署名はDTV操作トークン発行者が持つトークン署名鍵（秘密鍵）によってRSAや楕円曲線暗号などの一般に広く利用されている公開鍵暗号を用いて生成される．RSAや楕円曲線暗号は十分な鍵長を設定すれば計算量的安全性が確保されているため，トークンが不正に作成されることはない．DTVのセキュリティサービスプロバイダは，このトークン署名鍵に対応するトークン検証鍵（公開鍵）を有する．

ペアリングコードは，DTVの所有者がWebアプリに対してDTV操作を許可したことを示す情報である．アプリ初回起動フェーズにてCookieとして格納され，DTV操作フェーズにてCookieから読み出されて認証モジュールに渡される．認証モジュールはペアリングコードを直接DTVに送信するのではなく，OTTとして送信するため，ペアリングコードがネットワーク上を平文で送信されることはない．これにより，ネットワーク上でペアリングコードが盗聴されることはない．

一般的にCookieは読み出し範囲が規定されており，デフォルト値では保存したWebページのみ参照することができる．ペアリングコードはDTVを操作する際に必要な情報であるため，アクセス可能なWebアプリを必要最小限に制限する必要がある．StartupURLで示されるWebページと，DTV操作フェーズで使われるWebページが異なるWebページである場合，StartupURLで示されるWebページはDTV操作フェーズで使われるWebページでペアリングコードを読み出すことができるようにCookieを保存する際にCookieの読み出し範囲を適切に設定する必要がある．

5. セキュリティ機能の検証

本アーキテクチャで実現したセキュリティ機能を検証する．本アーキテクチャはユーザのDTVが不正操作されないようにするための仕組みを提供するとともに，開発者にとっても負担の少ないアーキテクチャとなっている．一方で，本アーキテクチャでは解決できない課題もある．以下にそれらを示す．

5.1 ユーザ側の利点

- 不正Webアプリによる DTV操作を防止：認証処理の中でDTVのセキュリティサービスプロバイダはDTV操作トークンに含まれるOriginとWebアプリのOriginが一致するかどうか検査処理を行っている．したがって，仮にペアリングコードが攻撃者に盗まれたとしても，許可されていないWebサーバ上に置かれたWebアプリからDTVが不正に操作されることはない．また，仮に不正Webアプリが正当なWebアプリと同じドメインに配置されていたとしてもCookieの読み出し範囲が制限されているため不正Webアプリによるペアリングコードの取得を防止できる．このように，本アーキテクチャではペアリングコード，Originによるドメインのチェック，Cookieの読み出し範囲制限といったように多重の対策が施されている．
- 不正ユーザによるDTV操作を防止：Webアプリが DTVと認証を成功させるにはDTVに設定されたペアリングコードと同じ値をWebアプリに入力する必要がある．仮に，DTVの所有者ではない不正なユーザがDTVの所有者になりすましてWebアプリを実行したとしても，ペアリングコードの値がDTVに設定された値と一致しなければ認証処理に失敗し，DTVを操作することはできない．
- 操作権限を越えたWebアプリからのDTV操作を防止：DTVの操作権限はDTV操作トークンのPermissionに記載された範囲に制限される．仮に外部Webアプリが書き換えられ，本来許可されていない操作命令をDTVに送信したとしても，DTVのセキュリティサービス側で拒絶するため，不正操作は実行できない．また，Permissionを活用することで，DTVのベンダはパートナー企業に限ってWebアプリからDTVの機能にアクセスを許可するといったビジネスモデルを構築することも可能である．

5.2 開発者の利点

- Webアプリの更新が可能：DTV操作トークンで許可された操作の範囲内であれば，アプリ提供後も任意のタイミングでWebアプリを更新することができるため，従来のパッケージアプリの開発に比べてリリーススケジュールに自由度を与えることができる．
- 外部WebアプリでもDTV操作が可能：DTV操作を行うWebページのドメイン名とDTV操作トークンに含めるOriginの値を一致させておけば，StartupURLで示されるWebアプリの開発者以外の第3者が提供する外部Webアプリであってもを操作することができる．したがって，DTVのベンダがWebアプリを必ずしも自社開発する必要がなく，アウトソース可能であるため開発費用を抑えることができる．さらに，Webアプリの開発者はアウトソース先にトークン署名鍵を提供する必要がないため情報管理コストを抑えることができるだけでなく，DTVを操作するWebアプリを開発する機会をWebアプリ開発ベンダに広く提供し，DTVと連動したさまざまなWebアプリの開発を促進することが期待できる．

5.3 従来手法との比較

遠隔操作アプリからのアクセス制御を実現する従来手法としてパスワードによるユーザ認証がある．しかし，遠隔操作アプリがWebアプリの場合，フィッシングサイトのように正規のWebアプリを装ったサイトからパスワードを不正に取得される可能性があるため，ユーザ認証だけでは不十分である．ユーザ認証を補完する手段としてクライアント認証やMACアドレス認証[3]，2段階認証が提案されており[4]，オンライン認証の実現手段として広く普及している．クライアント認証により，ブラウザを識別することは可能だが，個々のWebアプリを識別する手段ではないため，不正なWebアプリからの操作を防止することはできない．MACアドレス認証は機器の識別手段にすぎず個々のWebアプリの識別はできないため，不正なWebアプリからの不正操作を防止できない．2段階認証ではパスワード以外に認証コードが必要となるため，仮にパスワードが漏えいした場合でも不正操作を防止できるが，2段階認証は実行手順が複雑なためユーザにとって使い勝手が悪い．

また，あらかじめ信頼関係を構築したサービスから別のサービスにアクセス権限をセキュアに移譲する仕組みとしてOAuthが提案されている[5]．OAuthは主としてWebサービス間のアクセス権限の委譲を目的として利用されているが，DTV操作トークン発行者とDTVをサービスとみなせば，OAuthを適用してDTV操作トークン発行者からWebアプリにDTVへの権限を移譲させることで，WebアプリとDTVのアクセス制御に流用することも考えられる．しかし，OAuthはユーザがDTVサービス利用時にインターネット接続が必要となるほか，DTV操作トークン発行サービスを常時稼働させておく必要があるため提案方式と比較して運用コストが高い．

5.4 制約事項

- Webアプリの改変防止には限度がある：DTV操作トークンで許可された操作の範囲内でWebアプリを改変し，ユーザの意図しない操作命令を実行させる攻撃に対しては防止できない．たとえば，録画コンテンツの削除命令が許可されているWebアプリがあったとする．本来は本当にそのコンテンツを削除してよいかユーザに確認する画面を提示してから削除命令を発行するように設計しているにもかかわらず，確認画面をスキップするように書き換えられたとしても，その改変されたWebアプリの実行を防止することはできない．この対策として，TLS（Transport Layer Security）のCRL（Certificate Revocation List）のように[6][7]，失効したWebアプリのアプリ固有IDをリスト化して配布することで，いったん許可したDTV操作トークンを無効化する，いわゆるリボケーションの仕組みが考えられる．
- 同一ドメインに異なるポリシーのWebアプリが設置不可能：DTV操作トークンは公開情報のため，どのようなWebアプリからでも容易にDTV操作トークンを取得することができてしまう．したがって，WebアプリA用のDTV操作トークンをWebアプリBが流用することも簡単にできてしまう．複数のWebアプリが異なるドメインのWebサーバに配布されれば問題ないが，WebアプリBがWebアプリAと同一ドメインに配置される場合はWebアプリBからでもDTVを操作することができてしまう．したがって，DTVの操作を許可していない一般Webアプリを同一ドメインに設置できない．同様の理由で異なるポリシーのWebアプリを同一ドメインに設置することもできない．この制約に対しては，サブドメインを切り，WebアプリAとWebアプリBを別々のサブドメインに配置する運用上の解決手段がある．また，技術的な解決手段として，DTV操作ト

ークン発行時にWebアプリの配置場所を決めておき，DTV操作トークンにWebアプリのURLを含める．そして，認証モジュールでWebアプリのURLを取得してDTVに送信し，DTVのセキュリティサービスプロバイダで一致比較する方法が考えられる．

6. おわりに

DTVをネットワーク経由で操作する機能を持ったアプリをWebアプリとして構成すれば，Webサービスと連携する機能が容易に記述できる反面，一般ユーザにとっては従来のPCのパッケージアプリに比べてアプリを実行したり切り替えたりしているという感覚がなくなり，不正なWebアプリを実行してしまう危険性が高い．このため，従来のパスワードベースの認証に加えて新たなセキュリティ機能が必要となる．そこで，モバイル端末にインストールされた不正なWebアプリからDTVに対する意図しない操作を防止するセキュリティアーキテクチャを開発した．提案アーキテクチャは，暗号方式として一般に広く利用されている公開鍵暗号を利用し，またWebアプリとDTVの通信にはオープンなプロトコルであるWebSocketを活用しており，提案アーキテクチャの実装にあたり，新規開発部分が最小限となるよう考慮している．

モバイル端末上で動作するWebアプリから家庭内ネットワークを経由してDTVを操作するといったモデルはまだ一般的に広く普及していないため，セキュリティの要求条件も明らかではない．そこで，本稿では，まずセキュリティ上の脅威や課題を明らかにした．そして，その要求条件を満たすセキュリティアーキテクチャを示した．さらに，利点や制約条件について分析した．本稿ではスマートTVを題材としたが，本稿で示したプラクティスはスマートTV以外にもモバイル端末を操作しWebアプリで宅内機器を制御するさまざまな利用シーンにおいて活用できる．

クラウドサービスの普及に伴い，今後，宅内機器とモバイル端末を連携させた数多くのアプリやサービスが提供されていくと考えらえる．そのようなアプリやサービスはセキュリティの専門的な知識を持たない一般コンシューマが利用者層の中心になると予想されるため，本稿で示した煩雑な作業を必要としないセキュリティアーキテクチャが活用されると期待している．

参考文献

1) 海邊　裕，深井祐介，嵯峨砂与子：レグザクラウドサービス「TimeOn」のプラットフォーム技術，東芝レビュー，Vol.68, No.5, pp.36-39 (2013).
2) Digital Living Network Alliance: DLNA Guidelines http://www.dlna.org.（2016年4月現在）
3) 本田和博，柳　康裕，本間義久，河崎利信：IP通信機能搭載の住宅設備ネットワークシステム，松下電工技報，Vol.55, No.1, pp.4-9 (2007).
4) Aloul, F., Zahidi, S. and El-Hajj, W. : Two Factor Authentication Using Mobile Phones, IEEE/ACS International Conference on Computer Systems and Applications, pp.641-644 (2009).
5) Internet Engineering Task Force : RFC 5849 The OAuth 1.0 Protocol https://tools.ietf.org/html/rfc5849（2016年4月現在）
6) Internet Engineering Task Force : RFC 5246 The Transport Layer Security (TLS) Protocol Version 1.2 https://tools.ietf.org/html/rfc5246（2016年4月現在）
7) Internet Engineering Task Force : RFC 3280 Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile https://tools.ietf.org/html/rfc3280（2016年4月現在）

磯崎 宏（非会員）hiroshi.isozaki@toshiba.co.jp
2001年慶應義塾大学大学院政策・メディア研究科修了．同年（株）東芝入社．2008年カーネギーメロン大学電気＆コンピュータ工学部修士課程修了．2015年慶應義塾大学大学院政策・メディア研究科後期博士課程修了．博士（政策・メディア）．ホームネットワーク，セキュリティの研究開発に従事．

金井 遵（正会員）jun4.kanai@toshiba.co.jp
2006年東京農工大学工学部情報コミュニケーション工学科卒業．2008年日本学術振興会特別研究員．2009年東京農工大学大学院工学府電子情報工学専攻博士後期課程修了．同年（株）東芝入社．博士（工学）．オペレーティングシステムのシステムソフトウェア，セキュリティ，ネットワークの研究開発に従事．電子情報通信学会会員．

投稿受付：2015年5月29日
採録決定：2015年12月24日
編集担当：串田高幸（日本アイ・ビー・エム（株））