携帯型セキュリティアプライアンス上に HTML5 にて実装した パスワードマネージャの提案と実装

金谷 延幸† 野田 敏達‡ 長谷部 高行‡

†国立研究開発法人 情報通信研究機構
182-8585 東京都小金井市貫井北町 4-2-1
kanaya@nict.go.jp

‡株式会社富士通研究所
211-8588 神奈川県川崎市中原区上小田中 4-1-1
{noda.bintatsu, hasebe.takayuki}@jp.fujitsu.com

あらまし 本稿では、携帯可能なセキュリティアプライアンス上で動作し、HTML5にて実装したパスワードマネージャについて述べる。従来のパスワードマネージャは、複数端末間での共有・バックアップのため、インターネット上のサービスにパスワードを保管・管理しており、このサービスへの不正侵入や運営者の不正の脅威が課題となる。そこで、安全なHTML5アプリを作るための実行基盤「SEHTML5」を適用し、携帯可能な個人用セキュリティアプライアンス上にパスワードを保管・管理することで、「いつでも・どこでも・どんな端末でも」安心して使えるパスワードマネージャを実現した。

Proposal and Implementation of Password Manager in HTML5 on a Portable Security Appliance.

Nobuyuki Kanaya† Bintatsu Noda‡ Takayuki Hasebe‡

†National Institute of Information and Communications Technologies.
4-2-1, Nukui-Kitamachi, Koganei, Tokyo, 184-8795, JAPAN
Kanaya@nict.go.jp
‡FUJITSU LABORATORIES LTD.
1-1, Kamikodanaka 4-chome, Nakahara-ku, Kawasaki, JAPAN
{noda.bintatsu, hasebe.takayuki}@jp.fujitsu.com

Abstract Most of the password manager has a master server on the Internet in order for sharing and backup. The master server becomes a weak point because attackers can crack the server to steal password anytime, anywhere. In our research, we propose a new password manager based on our HTML5 security framework, and implement it on "Pocket-able Security Enhancement router (PSER)", then we can use it with PSER from any devices, any devices anytime anywhere.

1 はじめに

1.1 背景

長年使用されてきたパーソナルコンピュータに代わりスマートフォン、タブレット、スマートテレビ、車載器、スマートウォッチといった、様々なハードウェア・OS で動作する端末が使われ、様々な場所でインターネットに接続し、利用されるようになった。この環境においては、「いつ・どこでも・どんな端末でも」利用できる、アプリケーションやサービスが必要とされている。

この要望は、2014 年に W3C が勧告した HTML5 およびその周辺 JavaScript API にて実現可能と考える。HTML5 では、機能が大幅に強化され、例えば位置情報の取得、ファイルアクセス、双方向ソケット通信、データ格納などがブラウザ標準機能となった[1]。これらの拡張により、従来は PC 上のプラットフォーム固有のアプリとして実装されるのが当たり前であった、文書・プレゼンテーション作成や表計算といった事務用アプリケーションや、3D ゲームアプリケーションが、Web ブラウザ上で動作する HTML5 アプリとして実用化されている。このように、HTML5 と Web ブラウザは、様々な端末で動作するアプリケーションを実現するクロスプラットフォームなアプリ実行環境進化している。

従来の3階層モデルに代表される Web アプリケーションは、シンクライアントを構成し、端末側では UI のみを担当し、データやロジックをほとんど持たないため、端末側のセキュリティリスクは低いとされてきた。一方 HTML5 にて実装され、Web ブラウザ上で動作するアプリケーションは、複数のサーバと通信する分散クライアント・サーバ型を構成する。端末側にロジックとデータをダウンロードし、端末側が主体となってアプリが実行されるため、このロジックとデータを保護する必要がある。しかしながら、これまで HTML5 アプリに対するセキュリティ技術は少なく、そのリスクが高まっている[2]。例えば、独立行政法人情報処理推進機構(IPA)は、DOM Base XSS と呼ばれる、Web ブラウザ側で発生する Cross Site Scripting 脆弱性の急増を報告している[3]。

そこで、富士通研究所は、安全な HTML5 アプリを作るための実行基盤「SEHTML5」を開発した[4]。SEHTML5 は、「sandbox 属性」を応用した隔離環境を実現することを特徴とする。フレームの sandbox 属性を設定すると、自身が動作するフレームのみならず、自身のブラウジングコンテキストおよびその子孫のブラウジングコンテキストが固有の独立したオリジンとして扱われ、他のフレームに対する操作や、他のフレームからの操作が無効となる。この機構を使うことで、HTML5 アプリの脆弱性が発生した場合であっても、その影響を最小限に抑えることができる。

さらに、我々はこの SEHTML5 を使い、パスワードマネージャを実装した[5]。我々の実装では、従来の組み込みのパスワードマネージャと異なり、OS やブラウザを選ばず、事前のインストールなしに、「いつでも・どこでも・どんな端末でも」利用することができる。さらに、SEHTML5 によりセキュリティ強化をすることで、パスワードマネージャを他の Web アプリから隔離実行し、その安全性を確保している。

我々のパスワードマネージャは、SEHTML5 応用の一例であるが、この他にも HTML5 にて開発し、SEHTML5 を適用することで、「いつでも・どこでも・どんな端末でも」安心して使うことできる安全なアプリケーションを、容易に実現できると考える。

1.2 課題

パスワードマネージャでは、インターネット上に配備される複数端末間でのバックアップ・同期・共有に使われるサーバに対する脅威が課題となる。例えば Firefox 標準のパスワードマネージャでは Firefox Sync サーバに[7]、Mac OS X の Safari では keychain を介し iCloud 上に保存・共有される[8]。また、インターネットに配備した Web アプリにパスワードを保管・管理する「Web ベース型」のパスワードマネージャがある。この Web ベース型パスワードマネー

ジャは、「いつ・どんな端末でも」利用できるが、逆に攻撃は容易であり、侵入を許すことは、非常に深刻な問題となる。たとえば、パスワード一元管理ツールを提供する米 LastPass は、ハッキングを受けて利用者情報の一部が漏洩したと発表している[9]。この事件では、暗号化されたデータやパスワードそのものは漏洩していないとされるが、Web 型のパスワードマネージャの安全性、さらにはインターネットを介したセキュリティ機能提供の危険を実感させた。

さらに、既発表における我々のパスワードマネージャにおいても、SEHTML5 配備サーバに対する攻撃が、最も高いリスクと考える。SEHTML5 では、まず全体を管理し信頼点として機能する SEHTML5 アプリが起動することで安全性を確保する。この SEHTML5 の起動元となる Web サーバが攻撃された場合、全 HTML5 アプリに対する脅威となる。

このように、セキュリティの信頼点をインターネット上に配備した場合、「いつ・どこでも・どんな端末でも」セキュリティ機能の安全性を享受できる一方、攻撃者にとっては信頼点に対し「いつでも・どこからでも」攻撃できることを意味する。従って、信頼点に対する攻撃のリスクを低減させることが必要となる。

1.3 関連研究

HTML5 アプリのセキュリティに関する研究として、サーバサイドで動作する従来の Web アプリに対するセキュアな開発手法をベースとした研究がある。HTML5 アプリ開発者が脆弱性を作りやすいポイントや、チェックリスト、対策方法などが報告されている[2][3]。しかしながら、ブラウザ拡張をインストールせず、Web ブラウザの標準機能のみで HTML5 アプリの隔離実行環境を実現する研究は、我々の研究のほかにないと考える。

商用の Web ベース型のパスワードマネージャに対し、Zhiwei Li[10]らは、セキュリティ分析をしており、その全てに4種類の脆弱性を発見している。これらの脆弱性の本質的な原因は、パスワード入力に用いられる JavaScript プログラムが、インターネットに公開された Web パスワードアプリと通信するという Web ベース型の構成に起因しており、Web ベース型が脆弱性であることを示している。

2 我々のアプローチ

2.1 基本方針

従来のパスワードマネージャ、および既発表における我々のパスワードマネージャでは、「いつでも・どこでも・どんな端末でも」使えるようにするため、信頼点をインターネットに公開していたが、これがセキュリティ上のリスクとなっていた。そこで、信頼点をインターネットに公開せず、いつでもどこでも使えるコンセプト Pocket-able Security Enforcement Router (PSER)にて実現することで、信頼点に対するリスクを減らす。

我々は、「いつでも・どこでも、どんな端末」でも使えるセキュリティ機能を提供する個人用携帯セキュリティアプライアンス PSER を提案した[11]。PSER のコンセプトは、利用者が必要とする自分用のセキュリティ機能を PSER 内に集約・持ち運び、セキュリティ機能を利用端末へ提供する。例えば、パスワードや暗号鍵を保存し、暗号機能や、認証機能を周辺のハードウェアに提供する。この PSER さえ適切に管理すれば、「いつでも・どこでもどんな端末からも」セキュリティ機能を安全に利用することが可能となる。

本稿では、この PSER のコンセプトをパスワードマネージャと SEHTML5 に適用する。

- PSER 内にパスワードのバックアップ・同期機能を実装
- PSER 内に Web サーバを実装し、SEHTML5 を配備

以上により、信頼点をインターネット上に晒さず、他者に委託する必要もないため、信頼点に対するリスクを低減させることが可能となる。

2.2 基本構成

本稿におけるパスワードマネージャの構成を「図1.基本構成」に示す。

1. 代理入力ブックマークレット
   Webブラウザに表示されたログインフォームにパスワードを代理入力する。ブックマークレットとは、ブラウザのブックマークに登録した JavaScript プログラムのことであり、Web ブラウザへのプラグイン等のインストールなしに、手軽に JavaScript プログラムを実行させることができる。
2. パスワード管理クライアント
   代理入力するパスワードを、パスワード入出力 Web API 経由で取得し、パスワード代理入力機能に渡す。SEHTML5 管理アプリによりラップされ、その制御下で実行される。
3. パスワード入出力 Web API
   利用端末から PSER に対するインタフェースであり、PSER 内に蓄えられたパスワードをパスワード管理クライアントに返す。
4. パスワードマスター管理アプリ
   PSER の画面からアクセスし、PSER 内に蓄積されたパスワードの操作に使う。
5. SEHTML5 管理アプリ
   上記の HTML5 アプリ保護する JavaScript のライブラリであり、PSER 上に配備され、利用端末上のブラウザにダウンロードされ実行される.

なお、既発表[11]における PSER は、HTTP を中継する proxy として動作し、HTTP リクエストの Authorization ヘッダに ID とパスワードを埋め込むことで認証を代行する機能を実装していた。本稿では、ブックマークレットを利用したパスワードフィールドの代理入力という形態なので、既発表では対応しない、フォーム認証や TLS 通信にも対応し、より応用範囲が広がる。

PSER を利用する際は、利用したい場所の利用したい端末の近傍に PSER を持ち込み、利用端末をネットワークでつなげ、利用端末から PSER にアクセスし、利用端末から、PSER 内のパスワード管理アプリを利用する。この時、標準仕様の Web ブラウザさえあれば、どんな端末であってもパスワードマネージャを利用することができる。

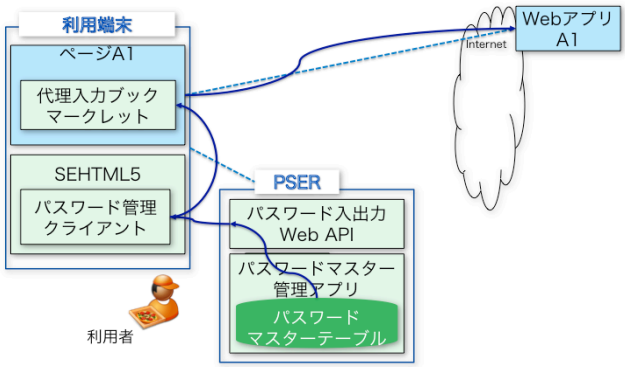

図1：基本構成

2.3 動作の仕組み

本提案によるパスワードマネージャの利用手順を示す。まず、事前準備として、利用端末の Web ブラウザより、パスワード管理クライアントの初期化を行う。以下に初期化の手順を示す。

1. 利用者は、利用端末と PSER をネットワークに接続
2. 利用者は、利用端末の Web ブラウザから、パスワード入出力 Web API にアクセス
3. パスワード入出力 Web API は、利用者と利用端末を認証し、パスワード管理クライアントの URL をレスポンス
4. 利用者端末上のブラウザは、SEHTML5 管理アプリによりラップされたパスワード管理クライアントをダウンロード
5. ダウンロードが完了すると、Web ブラウザ上にて実行・初期化

初期化が完了すると、次に利用者はパスワード入力対象となる Web アプリケーションにアクセスし、Web ブラウザ上に認証フォームが表示される。パスワードマネージャ利用時の処理は、以下の通り。

1. 利用者は パスワード代理入力機能ブックマークレットを起動
2. パスワード代理入力機能ブックマークレットは、パスワード管理クライアントに対しパスワードを要求
3. パスワード管理クライアントは、ブックマークレットからの要求を検証、正当であれば、パスワード入出力 Web API にパスワード

を要求
4. パスワードマスター管理 Web アプリは、利用者に自動入力の許可を要求
5. 利用者が承諾すると、パスワード入出力 Web API は、パスワードをパスワード管理クライアントに返答
6. パスワード管理クライアントが、ブックマークレットにパスワードを返答
7. ブックマークレットが、パスワード入力対象となるフォームにパスワードを入力

3 詳細設計と実装

本節では、先で述べた基本方針に基づき、詳細な設計と実装について議論する。

3.1 詳細設計

本パスワードマネージャでは、セキュリティの観点から以下を考慮した。

1. 利用端末からの最初のアクセス
2. パスワード管理クライアントとパスワード入出力 Web API(Web アプリ)間の通信

以下に、課題と対策について述べる。

3.1.1 利用端末からの最初のアクセス

利用端末から最初のアクセスでは、利用者が意図した端末からのアクセスのみに限定するために、利用者と利用端末を認証する手段が必要である。多くのパスワードマネージャでは、利用者が記憶したパスワードを入力することで、認証を実現する。PSER では、「ペアリング」と呼ばれる方式を用いる。ペアリングとは、PSER 上に表示されるランダムな文字列を、利用端末から入力する方式であり、PSER の既発表でも用いられている方式である。この方式の利点は、利用者にパスワードの記憶を強いることなしに認証できる点にある。PSER を信頼点として、常に携帯し、利用したい端末の近傍に持ち込み、PSER を直接操作できるというコンセプトが可能としている。

ペアリングの手順は、以下のとおり。

1. 利用者は、PSER の画面ロックを解除し、パスワードマスター管理アプリにアクセス
2. パスワードマスター管理アプリは、ランダムな一時鍵 Tk を生成、画面上に表示
3. 利用者は、利用端末の Web ブラウザからパスワード入出力 Web API にアクセス
4. パスワード入出力 Web API は、利用端末に対し、Tk を要求
5. 利用者は、PSER に表示された Tk を利用端末のブラウザより入力、ログイン
6. パスワードマスター管理アプリは、Tk を照合、正しければ利用端末にパスワード管理クライアントをレスポンス
7. 利用端末は、Web ブラウザにパスワード管理クライアントをダウンロード

このように、我々の PSER によるペアリングという方式は、信頼点である PSER が利用端末の近傍にありかつ双方に UI をもつという特徴を活かすことで、UI を介したセキュアな鍵交換チャネルを構成することを可能とする。

さらに、安全にするために、以下を考慮した。

- パスワードマスター管理アプリを localhost からのアクセスに限定する。
- PSER 上の画面にログインの成否などのログを表示させ、利用者に不正アクセスの有無を確認させる。
- Tk の有効期限を十分短くする。
- 画面のロックの解除の安全性・利便性については、PSER を実装する端末のハードウェアの機能を活用する。例えば、虹彩認証機能を備えた端末であれば、PSER に十分な安全性を担保できる。
- Tk の入力を QR コード等を使って入力可能にし、Tk が十分な長さであっても、利便性を確保する。

以上により、より安全でかつ、利便性の高い端末認証を実現することができる。

3.1.2 パスワード管理クライアントとパスワード入出力 Web API 間の通信

パスワード管理クライアントとパスワード入出力 Web API 間の通信に対する安全性を確保する方式として、以下の3点を検討する。

- 方式 1 セキュアな通信層の利用
- 方式 2 TLS の利用
- 方式 3 逆ペアリングによる独自実装

以下順に説明する。

1 番目のセキュアな通信層を利用する方式では、PSER と利用端末の間を Wi-Fi 等にて直接接続しており、WPA2 などの十分安全なプロトコルと鍵を用い、必要な安全性が確保される。

2 番目の TLS を利用する方式では、PSER 内の Web サーバに、公開鍵証明書をインストールし、利用端末から TLS を使いアクセスする。ブラウザにププレインストールされたルート CA から発行された証明書を用いる事ができれば、より安全かつ利便性も確保できる。しかし、現実的には自己署名証明書による運用が想定される。この場合、PSER 上に公開鍵証明書のフィンガープリントを表示し、利用端末にて確認することで安全性を確保できる。

3 番目の、逆ペアリングによる独自実装による方式では、パスワード管理クライアント端末側で最初に実行された際に、乱数から一時鍵 Tkc を生成、これを端末側に表示し、パスワードパスワードマスター管理アプリに入力することで、通信に使われるデータの暗号鍵を交換する。ただし、この方式では通信の秘匿を実現きるが、パスワード管理クライアントの改竄には弱いため、有効性は低い。

以上の検討結果より、もっとも実用的かつ安全性の高い方式は、方式 1 のセキュアな通信層を用いる方式であり、PSER をルータとして使う場合がもっとも安全であると言える。

3.2 PSER の実装

今回は、パスワードマスター管理アプリを含む PSER としての UI を、ネイティブアプリとして実装せず、移植性を考慮して Web アプリとして実装した(図２)。PSER に対する操作は、PSER の画面上で動作する Web ブラウザから localhost で動作する Web アプリにアクセスすることで操作し、Tk の表示やログ表示等もすべて Web アプリとして構成した。

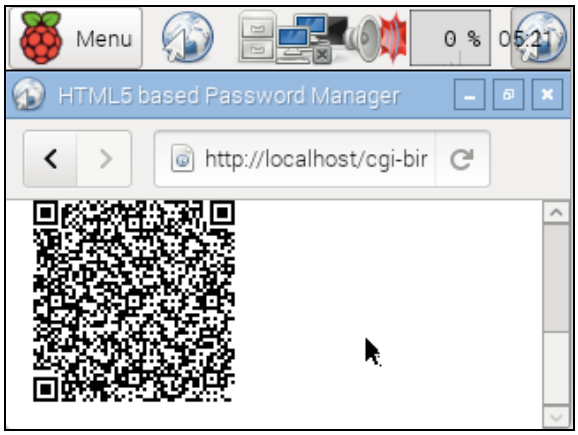

図 2: PSER に表示された
パスワードマスター管理アプリ画面

上記を実現するため、PSER に Web ブラウザと、HTTP サーバと、CGI が実行出来る環境、及び Perl の実行環境を用意する。HTTP サーバのコンテンツとして、先の発表で用いたパスワードマネージャアプリー式をベースに改良を行い、PSER のハードウェア向けに移植した。その中で SEHTML5 の導入は容易であった。SEHTML5 実装の実体は、静的な HTML5 と JavaScript ライブラリで構成しているため、HTTP サーバのコンテンツディレクトリに配置するだけでよく、コードの変更なしに、動作させることができた。

3.3 PSER のハードウェア

PSER を以下2つのハードウェア上に実装した。

1. Raspberry Pi+タッチセンサ付き LCD 液晶(図2)
2. Android スマートフォン

「Raspberry Pi」は、持ち運べる端末を簡単に実現できると考え、実装を試みた。なお、タッチセンサ付き LCD 液晶には、4D SYSTEMS の 4DPi-32 を用いている。Raspberry Pi では、

図 3:Rasperry PI＋LCD
（4D SYSTEMS のホームページより引用）

標準的なパッケージが充実しており、PSER 本来のコンセプトである「セキュリティの信頼点を集約し持ち運び可能にする」を実現し、様々なセキュリティ機能を拡張していくには、有益なプラットフォームと考える。

Android スマートフォンでの実現は、より現実的なハードウェア上での実現が必要と考え、実装を試みた。本稿では、Android のネイティブアプリではなく、HTTP サーバと CGI で構成されているが、Android では必要な環境が用意されておらず、選定と整備に手間がかかっている。HTTP サーバに軽量化のため busybox を改変した htttpd を用いた。改良には、CGI の実行パスの処理の変更、実行制御や、接続先制限が含まれている。さらに、Android 端末に sshd とクロスコンパイルした Perl をインストールし、実行環境を整えた。

3.4 実験環境

この PSER を利用したパスワードマネージャの実験環境を、利用者が認証対象 Web アプリ A1, A2 に対してアクセスする想定で構成した。

なお、PM、A1、A2 は、それぞれホスト”PM”、“A1”、”A2”とし、異なるオリジンで動作させた。また、A1 と A2 は簡単な CGI として実装し、ユーザ ID/パスワードをフォームから入力させ、プログラム内の ID/パスワードと照合し、認証に成功すると初期メニューがレスポンスされる。また、Windows7、iOS の上の端末を使い、ブラウザは IE、Firefox、safari、Chrome を利用し、パスワードマネージャの動作を確認することができた。

4 評価

本節では、メジャーな Web ブラウザで利用できる、ブラウザ標準組み込み型と Web ベース型について評価を行う。

4.1 ブラウザ標準組み込み型との比較

ブラウザ標準型である Firefox のパスワードマネージャと比較した場合、複数端末での同期機能に対する安全性の面で優位となるが、利便性において本提案手法は劣る。

Firefox Sync や iCloud によるブラウザ標準の同期機構では、暗号化されているとはいえ、インターネット上の第三者によって運用されおり、これが攻撃される、もしくは運営者の内部不正等によって、パスワードが漏洩する危険性があり、運用での不安が拭い切れない。我々の手法はブラウザ標準組み込み型に対し、利用できる OS やブラウザを選ばない点、パスワードを第三者に預けない点において、優位と考える。

4.2 Web ベース型に対する評価

文献[10]にて評価の対象とされている Web ベース型と比較した場合、本提案はオフライン実行の利便性とパスワード管理クライアントとの通信の安全性において優位であると考える。

まず、Web ベース型に共通の課題として、パスワードを保存し管理する機能がインターネット上の第三者によって運用されるという根本的な欠点がある。ブラウザ標準型では、同期機構を使わずともパスワードマネージャ単体で使うことができるが、Web ベース型ではパスワードの代理入力に必ず通信が必要となる。

さらに、文献[10]で Web ベース型に対する攻撃として述べられている「Bookmarklet Vulnerabiritlies」と「UI Vulnerabiritlies」に対し、本提案では根本的な対策が可能である。この攻撃は、ブックマークレットが悪意あるサイトで実行された場合に発生する攻撃であり、ブックマークレットからパスワード管理 Web アプリの通信を偽装することで、他サイトのパスワー

ドを搾取する。一方、本提案では PSER 上のパスワードマスター管理アプリがユーザに対する確認のダイアログを出すことで、たとえ対象となる Web アプリに対する XSS 攻撃等が発生した場合であっても、確実に利用者に警告を出すことができ、利用者の意図しないパスワードの代理入力を防ぐことができる。

5 まとめ

本稿では、携帯可能なセキュリティアプライアンスに配備され、いつでも・どこでも・どんな端末でも安心して使える、安全な HTML5 アプリを作るための実行基盤「SEHTML5」が適用されたパスワードマネージャについて提案した。

従来の、パスワードマネージャ共通の課題として、複数端末間でのパスワードバックアップ・同期・共有に使われ、インターネットに配備されるパスワードの管理機能に対する脅威がある。

そこで、我々はパスワードを携帯可能な個人用セキュリティアプライアンス PSER 上に保管する方式を提案した。我々の PSER は、セキュリティ上の信頼点を利用端末の近傍に持ち込み、「いつでも・どこでも」使えるセキュリティ機能を利用端末に提供する。また、HTML5 にて実装することで、「どんな端末」でも動作するアプリを実現することができる。この組み合わせにより、「いつでも・どこでも・どんな端末でも」使える、より安心なパスワードマネージャを実現することができた。

今後の課題として、SEHTML5 と PSER の組み合わせにより、より広いセキュリティ機能の実装が考えられる。この構成において、暗号機能や、証明書認証、SAML などのセキュリティ上機微となる処理を代行するような機能を PSER 内で完結させることで、「いつでも、どこでも、どんな端末からも、好きなときに使える」より安全な HTML5 アプリケーションを実現できると考える。

参考文献

[1] World Wide Web Consortium(W3C), "HTML5 A vocabulary and associated APIs for HTML and XHTML",

[2] 独立行政法人情報処理推進機構(IPA), 「HTML5 を利用した Web アプリケーションのセキュリティ問題に関する調査報告書」, https://www.jpcert.or.jp/research/html5.html

[3] 独立行政法人情報処理推進機構(IPA) IPA テクニカルウォッチ 『DOM Based XSS』に関するレポート, https://www.ipa.go.jp/about/technicalwatch/20130129.html

[4] 野田 敏達, 金谷 延幸, 長谷部 高行, 「HTML5 セキュリティ強化実行基盤の提案と実装」, SCIS2015

[5]金谷 延幸, 野田 敏達,長谷部 高行, HTML5 セキュリティ強化実行基盤「SEHTML5」を使ったパスワードマネージャの提案と実装, SCIS2015

[7] Mozilla.org, 「Firefox のパスワードマネージャ」, https://support.mozilla.org/ja/

[8] Apple Inc., "Keychain Services Concept" , https://developer.apple.com/library/ios/navigation/

[9] LasstPass, LastPass Security Notice, https://blog.lastpass.com/ja/2015/06/lastpass-security-notice.html/

[10] Zhiwei Li, Warren He, Devdatta Akhawe, Dawn Song, The Emperor's New Password Manager: Security Analysisof Web-based Password Managers 23rd USENIX Security Symposium, https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/li_zhiwei

[11]野田敏達,海野雪絵,金谷延幸,大久保隆夫, 「個人用セキュリティアプライアンスの提案」, 第 154 回 DPS・第 60 回 CSEC 合同研究会