Web サービスごとに隔離されたブラウジング環境の提案

三村　賢次郎 †　新　城　靖 †　張　世 申 †

1. はじめに

Web ブラウザは、ユーザの入力やファイルなど様々な情報をサーバに発信する．その一方で、ユーザの操作ミス、不正な Web ブラウザ拡張モジュールの利用や Web サーバからの不当な要求によってそれらの情報を意図せずサーバへ発信する可能性がある．

本研究では、ユーザが意図しない情報がブラウザから発信されることを防ぐ手段として Web サービスごとに隔離されたブラウジング環境を提案、および実装する．

2. Web サービスごとに隔離されたブラウジング環境

本研究では、図 1 のような Web サービスごとに隔離されたブラウジング環境を提案、および実装する．

- Web サービスごとに仮想環境を用意する．
- 仮想環境ごとにファイシステムは独立である．
- 仮想環境中の Web ブラウザから対応する Web サービスへの接続のみを許可する．
- もし、環境中の Web ブラウザが許可されていないサービスへ接続を試みた場合には、対応表を参照し、それが許可されている環境で動作している Web ブラウザに自動的に引き継ぐ．

Web サービスと仮想環境の対応表の例を表 1 に示す．この例では、対応表にサービスとして国税電子申告・納税システムである e-Tax と Google 検索を定義している．この環境を利用することで、情報の発信の範 囲を限定することができる．具体的に、e-Tax を用い て国税に係る申告・申請・納税をする場合を考える．e-Tax へ様々な申請をする際に必要な情報を隔離され ていない普通の PC のファイルに用意したとする．この状態で他の Web サービスを利用した場合に、操作 ミスや不正な Web ブラウザ拡張モジュールなどの利用によってこの情報が他の Web サーバへ発信されてしまう可能性が考えられる．

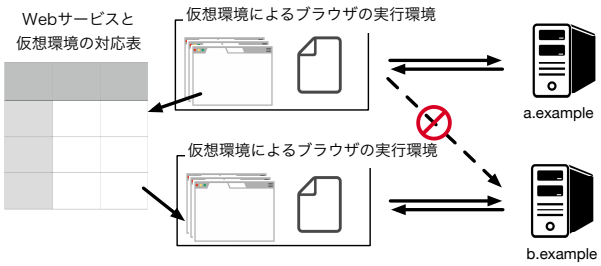

図 1　Web サービスごとに隔離されたブラウジング環境

本研究で提案するブラウジング環境では、仮想環境ごとにファイルは独立している．そのため、例えば他の Web サービスは e-Tax で利用したファイルを利用することはできない．これにより、e-Tax 以外のサービスを利用している際に、e-Tax で用いたデータが外部へ発信される可能性を防ぐことができる．

3. 隔離されたブラウジング環境のコンテナによる実装

本研究では、Web サービスごとに隔離されたブラウジング環境を Docker[1] コンテナを用いて実装する．Docker コンテナは、仮想環境として独立したファイル構成を持つことができる．

本研究では、ブラウザとして Firefox[2] を用いる．この FireFox に WebExtensions[3] の API である WebRequest[4] による拡張モジュールを導入する．WebRequest は、ブラウザ上で作成される HTTP リクエストの作成および、送信の前後でリクエストに関する情報を受け取り、HTTP ヘッダーの編集やリク エストの取り消しを行うことができる．この API を利用することで、実際に Web サーバに接続する前に接

表 1　サービスと仮想環境の対応表の例

サービス名	ドメイン名	コンテナ ID
e-Tax	e-tax.nta.go.jp	1
google 検索	google.jp	2
google 検索	google.com	2
その他	*default*	3

† 筑波大学
University of Tsukuba

続先の情報を取得し、HTTP リクエストをキャンセルすることができる.

また、本研究ではドメイン名とコンテナの対応表をコンテナの外部にサーバとして実装する. ユーザは、この対応表にそれぞれのコンテナが利用できる Web サービスを定義する.

ブラウザが他の Web サー ビスに接続する際に、対応表を用いて接続情報を対応するコンテナへ引き継ぐ. 対応表サーバとコンテナ内のブラウザを WebSocket を用いて接続する. ユーザが URL をたどる操作をするたびに、各コンテナのブラウザは、接続情報として接続先のドメイン名と URL を対応表サーバに渡す. 対応 表サーバは、受け取ったドメイン名を元に表 1 のような表を参照し、接続するコンテナを決める. そして、 そのコンテナで動作している Web ブラウザに対して WebSocket で接続情報を渡す. 受け取った Web ブラウザは、接続情報を元に Web ページを表示する.

現在の実装では、ユーザは対応表サーバーにドメイン名を事 前に設定しておく必要がある. 将来的にはユーザが コンテナとドメイン名の管理を容易にできるようにする. 具体的には、ドメイン名が設定されていない Web サービスに対して新しいコンテナの生成や不要なコンテナの削除などをユーザが対話的に行えるようにする. また、URL のドメイン名の部分だけでなく、パスの部分や POST の内容によっても、コンテナを切り替えられるようにしたいと考えている。

4. 関 連 研 究

Qubes OS[5] は、使用目的別にアプリケーションを仮想マシンを用いて実行することでセキュリティを高 める OS である. この OS では、どのアプリケーショ ンをどの VM で実行するかをユーザが決める. Qubes OS で Web ブラウジングをする際には、接続先の Web サービスごとに自動的にユーザが注意して VM を選 択し実行する必要がある. 本研究では、Web サービス ごとに接続を切り替えるため、ユーザが誤って意図し ない環境で Web サービスへ接続することを抑制する ことができる.

Web ブラウザは高速化のために Web 上の静的なリソースをローカルストレージにキャッシュする. Bin らの研究[6] によれば、このキャッシュの機能を利用して、リソースがローカルからロードされたかどうかを速度の面から推測しユーザが過去に訪れた Web サイトを推測できることを示している. また、Hyungsub らの研究[7] では、同様のキャッシュによる推測を利用してユーザのログイン状態を推測できることを示している. 加えて、ブラウザのウィンドウサイズの時間変化を受動的に監視してユーザがどのタブを閲覧しているかを推測する手法を示している. 本研究の環境を用いることで、キャッシュされるリソースは、同一の Web サービス内に限定されるため Web サービスを跨いだユーザの履歴推測は不可能であると考えられる.

5. ま と め

本研究では、ユーザが意図しない情報を外部へ発信するのを防ぐ手段として、Web サービスごとに隔離されたブラウジング環境を提案する. この環境では、ユーザのミスや Web サーバからの不当な要求によって生じる情報の発信の範囲を Web サービスに対応する仮想環境内の情報に限定することができる.

現在、 Web サービスごとに隔離されたブラウジング環境の実装は、現在はあらかじめドメインの設定を必要とする. 将来的にはユーザが Web サービスとコンテナの管理を容易にできるようにする. 具体的には、新たな Web サービスに対して新しいコンテナの生成や不要なコンテナの削除などをユーザが対話的に行えるようにする.

参 考 文 献

1) Docker - Build, Ship, and Run Any App, Anywhere: https://www.docker.com/, Accessed: 2017-10-18.
2) Firefox: https://www.mozilla.org/ja/firefox, Accessed: 2017-10-18.
3) WebExtensions - Mozilla — MDN: https://developer.mozilla.org/ja/Add-ons/WebExtensions, Accessed: 2017-10-18.
4) WebRequest - Mozilla — MDN: https://developer.mozilla.org/en-US/Add-ons/WebExtensions/API/webRequest, Accessed: 2017-10-18.
5) J. Rutkowska, R. Wojtczuk, "Qubes OS Architecture", Version 0.3. Jan. 2010, http://qubes-os.org/.
6) B. Liang, W. You, L. Liu, W. Shi and M. Heiderich, "Scriptless Timing Attacks on Web Browser Privacy," 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2014, pp. 112-123.
7) Hyungsub Kim, Sangho Lee, and Jong Kim, "Inferring browser activity and status through remote monitoring of storage usage," 32nd Annual Conference on Computer Security Applications (ACSAC '16), 2016, pp. 410-421.