Busting Frame Busting 機能を備えた Click Jacking 攻撃と”same-origin” reflected XSS 攻撃

田邉杏奈[1] 寺本健悟[2] 齊藤泰一[1]

概要：Web アプリケーション上における攻撃手法の一つとして Click Jacking 攻撃がある．Click Jacking 攻撃とは，ユーザを視覚的にだまし，正常に見える Web ページ上のコンテンツをクリックしようとするユーザに，別の Web ページのコンテンツをクリックさせる攻撃のことである．Click Jacking 攻撃には iframe を用いる場合が多い．iframe を使用した Click Jacking 攻撃を防ぐための技術として，Frame Busting がある．Frame Busting の中には，JavaScript を使用する場合もある．
そして，Frame Busting を回避する技術として，Busting Frame Busting がある．それは， Web ブラウザに搭載されている XSS Filter 機能を悪用し，Frame Buster を行うための JavaScript を，無力化させる．本稿では，いくつかのブラウザにおいて，XSS Filter 機能を利用した Busting Frame Busting が可能であるかを調査した．

その結果から，IE の XSS Filter では，”same-origin” reflected XSS 攻撃に対応できないのではないかと推測された．“same-origin” reflected XSS 攻撃に対していくつかのブラウザの XSS Filter が動作するかについて検証を行った．その結果，IE の XSS Filter は動作せず，”same-origin” reflected XSS 攻撃が成功した．

Click Jacking Attacks using Busting Frame Busting vs “same-origin” reflected XSS Attacks

ANNA TANABE[1] KENGO TERAMOTO[2] TAIICHI SAITO[1]

1. はじめに

Web アプリケーション上における攻撃手法の一つとして Click Jacking 攻撃がある．そして，Click Jacking 攻撃を防ぐ方法として，JavaScript を利用した Frame Busting という技術がある．しかし，JavaScript を Frame Busting を回避するための技術として Busting Frame Busting という技術が存在する．Busting Frame Busting を行うために，XSS フィルタを利用している[1].

本稿では，3 つのブラウザ IE(Internet Explorer)，Chrome(GoogleChrome)，Edge(Microsoft Edge)が持つ XSS フィルタ機能を用いて，Busting Frame Busting が可能であるか調査を行った．

2. 準備

2.1 Reflected XSS(Cross-site Scripting)攻撃

XSS 脆弱性によって，悪意のある JavaScript スクリプトを既存のページに不正に挿入する攻撃が可能となる．XSS 攻撃には Reflected （反射型）と Stored（持続型）の 2 種類が存在する．ここでは，Reflected XSS 攻撃について説明する．

例えば，サーバ B の target.php に Reflected XSS 脆弱性があったとする．攻撃者はサーバ A の trap.html に悪意のあるリンクを作成する．そして，ユーザは trap.html にアクセスしリンクをクリックしてしまうとする．リンクの URL のクエリ文字列には悪意のある JavaScript スクリプトが含まれている．ユーザがリンクをクリックすることでその JavaScript スクリプトを含むリクエストがターゲットサーバであるサーバ B の target.php に送信され，それが反射され悪意のある JavaScript スクリプトをユーザは受け取ってしまう(図 1).

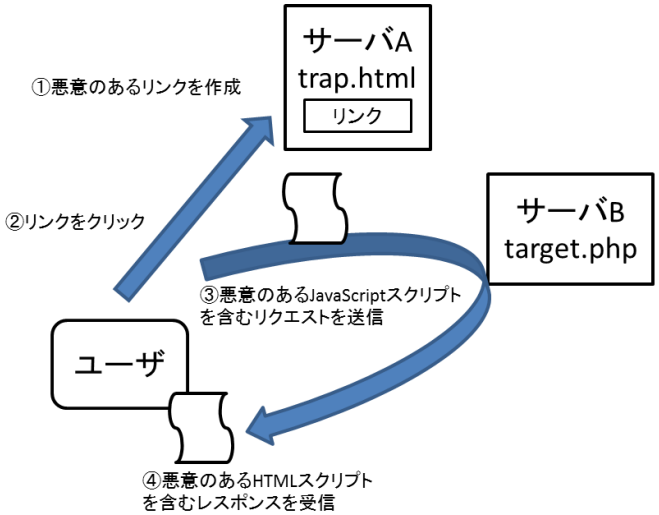

図 1. Reflected XSS 攻撃

1 東京電機大学
2 東京電機大学大学院

2.2 “same-origin” Reflected XSS 攻撃

攻撃者の用意した悪意のあるリンクとターゲットサーバが同じホストである場合の Reflected XSS 攻撃を，本稿では“same-origin” Reflected XSS 攻撃と呼ぶ．

例えば，サーバ A の target.php に Reflected XSS 脆弱性があったとする．この時，攻撃者が，サーバ A の review.php においてレビュー欄に悪意のあるリンクを投稿する．そして，ユーザは review.php にアクセスしリンクをクリックしてしまう．リンクの URL のクエリ文字列には悪意のある JavaScript スクリプトが含まれている．ユーザがリンクをクリックすることでその JavaScript スクリプトを含むリクエストが同じホストであるサーバ A の target.php に送信され，それが反射され悪意のある JavaScript スクリプトをユーザは受け取ってしまう(図 2)．

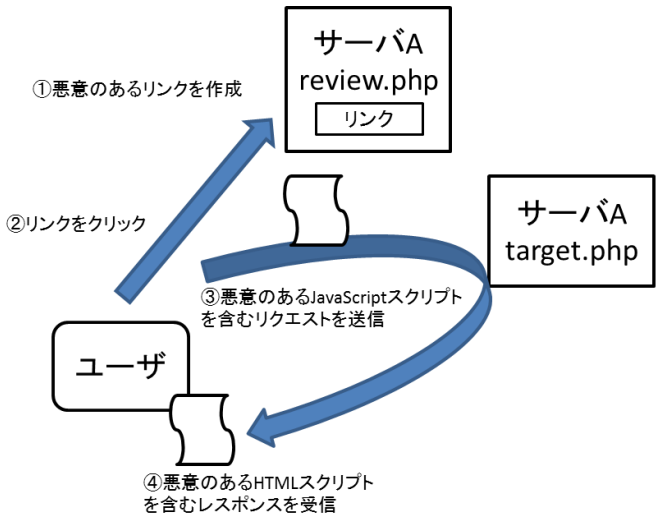

図 2. “same-origin” Reflected XSS 攻撃

2.3 Stored HTML インジェクション攻撃

攻撃者はターゲットサーバに悪意のある HTML スクリプトを永久的に格納する．ユーザはサーバにリクエストを送り，格納された悪意のある HTML スクリプトをサーバから受け取ってしまう．

例えば，サーバ A の target.php に Stored HTML インジェクション脆弱性があったとする．この時，攻撃者はサーバ A の target.php に悪意のある HTML スクリプトを送信する．この時，サーバ A の target.php は悪意のある HTML スクリプトを受け取り格納してしまう．悪意のある HTML スクリプトが格納された後，ユーザがサーバ A の target.php にアクセスすることで悪意のある HTML スクリプトが格納されたレスポンスを受け取ってしまう(図 3)．

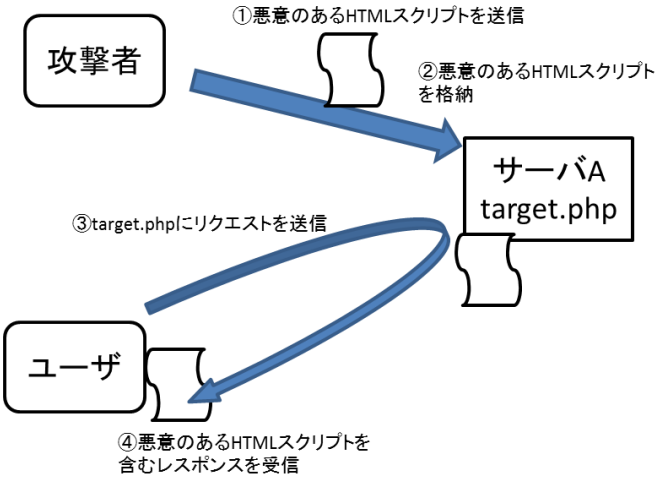

図 3 .stored HTML インジェクション攻撃

2.4 Click Jacking 攻撃

Click Jacking 攻撃とは，Web アプリケーション上における攻撃手法の一つである．Click Jacking 攻撃は，ユーザを視覚的にだまし，正常に見える Web ページ上のコンテンツをクリックしようとするユーザに，別の Web ページのコンテンツをクリックさせる攻撃のことである．

例えば，ユーザは正常に見える Web ページ A(図 3)にアクセスし，ENTER ボタンをクリックしようとする．

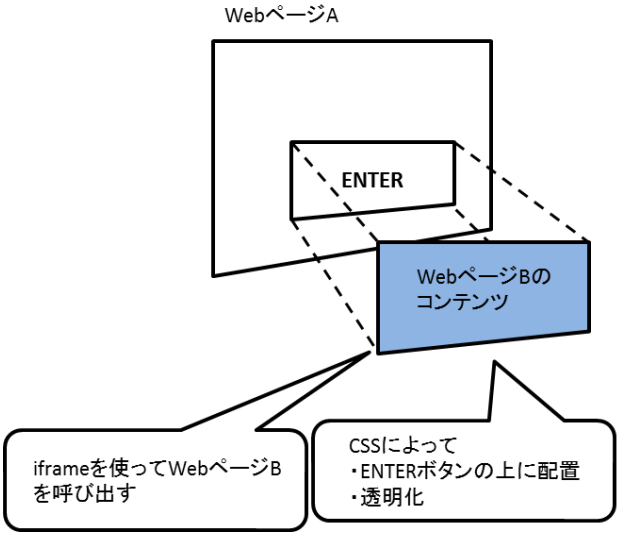

図 4.Click Jacking 攻撃

しかし，実際にユーザがクリックするのは，透明化された Web ページ B のコンテンツである．

本稿における Click Jacking 攻撃では，Web ページ B を呼び出すために，iframe を使うこととする．また，Web ページ B を呼び出した iframe を透明化し，ENTER ボタンの上に重ねるために CSS(Cascading Style Sheets)を使用する(図 4)．

2.5 Frame Busting

iframe を使用した Click Jacking 攻撃を防ぐための技術として，Frame Busting がある．Frame Busting を行うことにより，iframe 内で呼び出されることを防ぐことができる．これにより Click Jacking 攻撃を防ぐことが可能となる(図 4).

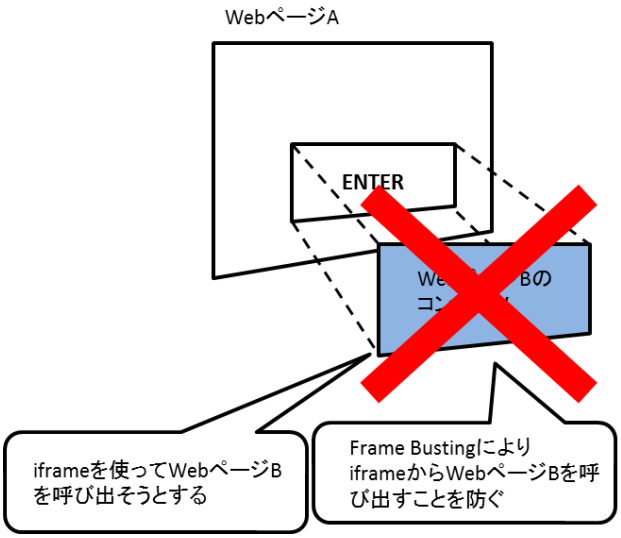

図 4. Frame Busting

Frame Busting の中には, JavaScript を使用する場合がある．JavaScript を利用し，TOP ページと子フレームの location が一致しているかを判定する．例としては以下のコードである(図 5)．本稿では，Frame Busting コードとして図 5 を想定する．

```
<script>
 if(top.location.hostname!=self.location.hostname){
  top.location=self.location;
 }
</script>
```

図 5. Frame Busting コード

Web ページ B にこのような Frame Busting コードが含まれていたとする．そして Web ページ A から iframe を使い Web ページ B を呼びだそうとする(図 6)．この時，Web ページ B に含まれる Frame Busting コードにより，Web ページ A(top.location)のホスト名と Web ページ B(self.location)のホスト名を比較する．その結果，Web ページ A のホスト名と Web ページ B のホスト名が異なる場合，top.location を Web ページ B(self.location)に上書きする．これにより，Web ページ B が iframe 内で呼び出されることを防ぐことができる．

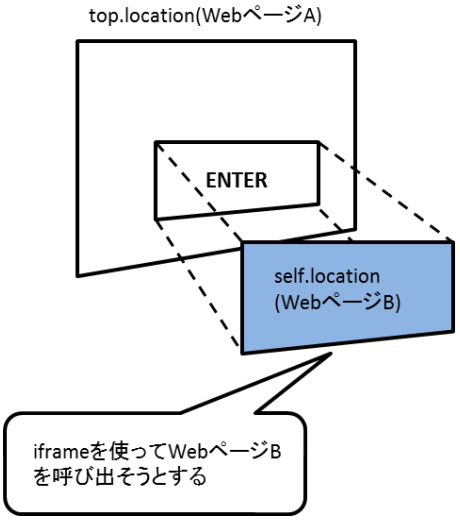

図 6. Frame Busting の例

Gustav Rydstedt ら[1]は Alexa-TOP500sites において，Frame Busting の実装が行われているかについて調査している．調査した多くのサイトでは，JavaScript を使用して， Frame Busting を行っていると報告している．

2.6 Busting Frame Busting

Gustav Rydstedt ら[1]はさらに，JavaScript を使用した Frame Busting を回避する技術である Busting Frame Busting について提案している．

2.6.1 XSS Filter を用いた Busting Frame Busting

IE，Chrome，Edge には，XSS 攻撃から Web ページを守るための XSS フィルタが搭載されている[3]．JavaScript を利用した Frame Busting を回避する場合，これらの Web ブラウザに搭載されている XSS フィルタ機能を悪用し，Frame Busting を行うための JavaScript を，無力化させる．

例

hostA の framebusting.php の HTML は以下の通りである(図 7)． framebusting.php には Frame Busting を行うためのコード(図 5)が含まれているとする．

```
<html>
 <head>
  <script>
   if(top.location.hostname!=self.location.hostname){
    top.location=self.location;
   }
  </script>
 <head>
</html>
```

図 7. framebusting.php

この時，攻撃者は attacker/frame.php から iframe を使用し hostA の framebusting.php にリクエストを送信させる(図 8, 図 10).

```
<iframe src=" hostA/framebusting.php?name=
<script>%0d%0a%20%20%20if(top.location.hostname!=self.location.hostname){%0d%0a%20%20%20%20top.location=self.location;%0d%0a%20%20%20}%0d%0a%20%20</script>" >
```

図 8. Busting Frame Busting コード

パーセントエンコーディングを元に戻すと以下のようになる(図 9).

```
<iframe src=" hostA/framebusting.php?name=
  <script>
   if(top.location.hostname!=self.location.hostname){
    top.location=self.location;
   }
  </script>
```

図 9. Busting Frame Busting コード(変換)

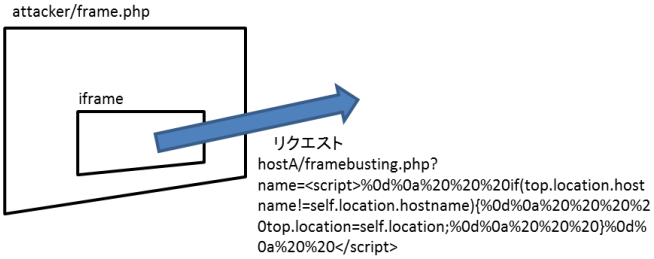

図 10. Busting Frame Busting

これにより，IE，Edge，Chrome では XSS フィルタが動作する．その結果，framebusting.php(図 7)に含まれている Frame Busting コード(図 5)がエスケープされる. このため，Busting Frame Busting が成功し，Click Jacking 攻撃が可能となる．

3. Busting Frame Busting 機能を備えた Click Jacking 攻撃の検証

本章では，Chrome，IE， Edge において”same-origin” Busting Frame Busting が可能であるかを検証した.

3.1 実験環境

IE のバージョン:11.212.10586.0
Chrome のバージョン:50.0.2661.94 m
Edge のバージョン:25.10586.0.0

今回の実験では，hostA の frame.php の iframe から hostA の framebusting.php(図 7)を呼び出す．この時， Busting Frame Busting を行う．本稿では，この実験における Busting Frame Busting を”same-origin” Busting Frame Busting と呼ぶ.
この攻撃が成功するために，hostA の frame.php には Stored HTML インジェクション脆弱性がある必要がある．hostA の frame.php に Stored HTML インジェクション脆弱性がある時，攻撃者は，hostA の frame.php に図 8 のような iframe を埋め込む.
ユーザが, hostA の frame.php にアクセスすることで, iframe から“same-origin” Busting Frame Busting を行うためのリクエストが送信される(図 11).

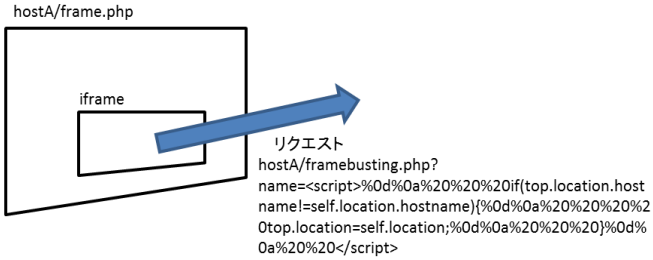

図 11. ”same-origin” Busting Frame Busting 攻撃の検証

“same-origin” Busting Frame Busting 機能を備えた Click Jacking 攻撃が各ブラウザで成功するかを検証する.

3.2 実験結果

結果を表 1 に示す. “same-origin”の場合， IE および Edge の XSS フィルタは動作せず，Frame Busting が動作するため，Busting Frame Busting は成功しなかった．“same-origin”の場合， Chrome の XSS フィルタは動作するため，Frame Busting が動作せず，Busting Frame Busting は成功した． この結果から，”same-origin”の場合，IE および Edge の XSS フィルタは動作しないことが考えられる.

表 1. 実験結果 1

ブラウザ	“same-origin” Busting Frame Busting 攻撃
IE	失敗
Chrome	成功
Edge	失敗

4. “same-origin” Reflected XSS 攻撃の検証

3 章の実験結果から，”same-origin” XSS 攻撃に対して，IE および Edge の XSS フィルタは働かないのではないかと推測する．本章では，Chrome，IE， Edge で”same-origin” Reflected XSS 攻撃が可能であるかを検証した.

4.1 実験環境

各ブラウザのバージョンは，3 章と同じである.
今回，hostA の xssvul.php に XSS 脆弱性があるとする.

xssvul.php のコードは以下の通りである(図 12).

```
<html>
 <head>
  <?php
   $name=$_GET["name"];
   echo($name);
  ?>
 </head>…
</html>
```

図 12. xssvul.php

攻撃者は，hostA の sample.php に以下のようなリクエストを送信させるリンクを挿入する(図 13).

```
hostA/xssvul.php?name=<script>XSS 攻撃コード </script>
```

図 13. XSS 用リンク

ユーザは，hostA の sample.php にアクセスし，リンクをクリックする．このリクエストに対して以下のようなレスポンスを受け取る(図 14).

```
<html>
 <head>
  <script>
   XSS 攻撃コード
  </script>
 </head>…
</html>
```

図 14. 攻撃コードの埋め込まれた xssvul.php

XSS 攻撃コードによってブラウザ上で JavaScript が動作する．”same-origin” Reflected XSS 攻撃(図 15)が各ブラウザで成功するかを検証する．

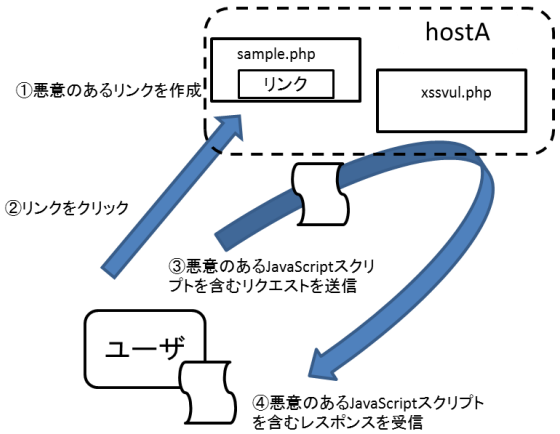

図 15. ”same-origin” Reflected XSS 攻撃の検証

4.2 実験結果

結果を表 2 に示す．“same-origin” reflected XSS 攻撃に対して，IE および Edge の XSS フィルタは動作せず，攻撃が成功した．一方，同じ“same-origin” reflected XSS 攻撃に対して，Chrome の XSS フィルタは動作したため，攻撃が成功しなかった．

表 3. 実験結果 2

ブラウザ	“same-origin” XSS 攻撃
IE	成功
Chrome	失敗
Edge	成功

5. まとめ

“same-origin” Busting Frame Busting 攻撃および“same-origin” XSS 攻撃結果を表 2 に示す．IE および Edge の XSS Filter は”same-origin” reflected XSS 攻撃を防ぎきれていないということが言える．”same-origin” reflected XSS 攻撃を防ぐという点では，IE の XSS Filter は劣っておる．しかし，他の XSS Filter は制限が強すぎるため，誤検知を悪用され，攻撃に利用されるケースが多い．

表 4. 実験結果まとめ

ブラウザ	“same-origin” Busting Frame Busting 攻撃	“same-origin” XSS 攻撃
IE	失敗	成功
Chrome	成功	失敗
Edge	失敗	成功

参考文献

[1] Gustav Rydstedt, Elie Bursztein, Dan Boneh. Collin Jackson : Busting Frame Busting: a Study of Clickjacking Vulnerabilities on Popular Sites ,Web2.0 Security and Privacy 2010(W2SP 2010)(2010)

[2] ipa 独立行政法人情報処理推進機構技術本部セキュリティセンター: 知らぬ間にプライバシー情報を非公開設定を公開設定に変更されてしまうなどの「クリックジャッキング」に関するレポート,(2013)

[3] Bhawna Mewara, Sheetal Bairwa, Jyoti Gajrani, : Browser’s Defenses Against Reflected Cross-Site Scripting Attacks, Signal Propagation and Computer Technology (ICSPCT) (2014)