

Google Chrome のパスワードマネージャの脆弱性

市原 隆行 †

寺本 健悟 ‡

齊藤 泰一 §

† 東京電機大学

120-8551 東京都足立区 千住旭町 5
cludzerothree@gmail.com

‡ 東京電機大学大学院

120-8551 東京都足立区 千住旭町 5
15kmc11@ms.dendai.ac.jp

§ 東京電機大学

120-8551 東京都足立区 千住旭町 5
taiichi@c.dendai.ac.jp

あらまし Google Chrome のパスワードマネージャは、保存されたユーザ ID とパスワードを補完するオートコンプリート機能を持つ。本稿は Google Chrome のパスワードマネージャの脆弱性を示す。WhiteScid と Andrea Giammarchi は、Firefox と Internet Explorer のパスワードマネージャの脆弱性を報告しているが、それらを利用する攻撃は XSS 脆弱性を必要とする。一方、我々の示す Google Chrome の脆弱性を利用する攻撃は、HTML インジェクション脆弱性のみを必要するため、JavaScript が動作しない環境であっても実行可能である。

Vulnerability of password manager in Google Chrome

Takayuki Ichihara †

Kengo Teramoto ‡

Taiichi Saito §

† Tokyo Denki University

5 Senju, Asahi-cho, Adachi-ku, Tokyo 120-8551, JAPAN
cludzerothree@gmail.com

‡ Graduate School of Tokyo Denki University

5 Senju, Asahi-cho, Adachi-ku, Tokyo 120-8551, JAPAN
15kmc11@ms.dendai.ac.jp

§ Tokyo Denki University

5 Senju, Asahi-cho, Adachi-ku, Tokyo 120-8551, JAPAN
taiichi@c.dendai.ac.jp

Abstract This paper presents a password manager's vulnerability in Google Chrome. Even when the hostname in a page's URL is the same as in the login page but the action attribute is not the same, if the username textbox is clicked and the username is selected from the suggested list of options, the password manager fills in the corresponding password. This

allows attackers to steal the username-password pair saved in the password manager if the login page or other pages in the same domain are vulnerable to HTML injection.

1 はじめに

Web ブラウザのパスワードマネージャは、ユーザが入力したパスワードを記憶し、パスワードの入力を要求された際に自動入力する機能を持つ。一方で、パスワードを自動で入力するため、パスワードを盗み出しが可能になる場合がある。本稿では、発見した Google Chrome のパスワードマネージャの脆弱性について報告する。

2 準備

2.1 HTML の入力フォーム

Web サイトの開発者は、ユーザにユーザ名やパスワードを入力・送信させるために HTML の form 要素と input 要素を用いる(図 1)。form 要素の action 属性は入力された情報の送信先を示す。input 要素をフォームの入力に用いる場合は、form 要素の開始タグと終了タグの間にそれを追加する。

input 要素は入力に用いる UI(User Interface)を生成する(図 2)。その種類は type 属性によって決定される。type="text" の場合、テキストボックスを生成する。type="password" の場合も、テキストボックスを生成する。しかし、テキストボックスに入力された文字は●と表示され、画面上では確認できなくなる(図 3)。type="submit" の場合、ボタンを生成する。このボタンが押されると、form 要素内の input 要素に入力された内容が送信される。

```
<form action="login.php" method="post">
  <input type="text" name="username"
    id="username"><br>
  <input type="password" name="userpass"
    id="userpass"><br>
  <input type="submit" value="SEND">
</form>
```

図 1. 入力フォームの HTML のスクリプト

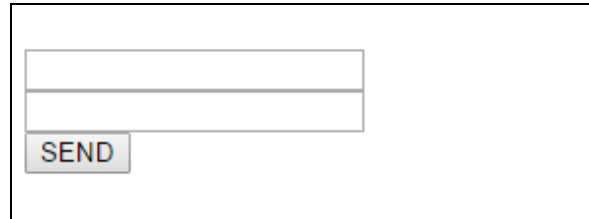


図 2. HTML の入力フォーム




図 3. HTML の入力フォームの例

2.2 パスワードマネージャ

パスワードマネージャは、ユーザがパスワードを入力した際、ブラウザに以下の情報を保存する。

- ・Web サイトのホスト名
- ・ユーザ ID
- ・パスワード

再度、同じ Web ページを訪れた際に、ユーザ ID とパスワードを自動で入力する。

パスワードマネージャの動作について簡単に説明する。まず、パスワードの保存について説明する。

- ①ブラウザが、ユーザ ID とパスワードの入力フォームを含む Web サイトを表示する。
- ②ユーザが、ユーザ ID とパスワードを入力する。
- ③入力フォームに入力された内容を送信する。
- ④ユーザ ID とパスワードの保存を行うかどうかを選択するダイアログが表示される(図 4)。

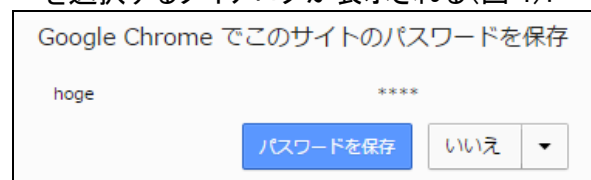


図 4. パスワードマネージャのダイアログ

⑤図 4 のダイアログで保存することを選択する。

次に、パスワードの自動入力について説明する。上のパスワードの保存をすでに行っているとする。

- ① 再度、ユーザ ID とパスワードの入力フォームを含む Web サイトにアクセスする。
- ② 入力フォームのテキストボックスに、ユーザ ID とパスワードが自動で入力される。

2.3 従来研究

WhiteAcid[1] は Web ブラウザ Mozilla Firefox のパスワードマネージャの脆弱性を報告した。攻撃対象サイトは以下のような性質を持つ必要がある。

- ・ユーザ ID とパスワードの入力フォームを持つ Web ページと XSS 脆弱性を持つ Web ページが同一のホストに存在

- ・XSS 攻撃によりログアウト処理が可能

攻撃者は JavaScript で書かれたエクスプロイトを用意する。これを XSS 脆弱性のあるページに挿入し、ユーザのブラウザ上で実行する。まず、エクスプロイトはログアウト処理を行う。次に iframe 内にログインページを取得する。すると、Firefox のパスワードマネージャが、ログインページの入力フォームにユーザ ID とパスワードを自動で入力する。その後、エクスプロイトは、パスワードを盗み出す。

Andrea Giammarchi [2] は、Internet Explorer の脆弱性を報告した。

攻撃対象サイトは以下のような性質を持つ必要がある。

- ・ユーザ ID とパスワードの入力フォームを持つ Web ページと XSS 脆弱性を持つ Web ページが同一のホストに存在

- ・XSS 攻撃によりログアウト処理が可能

攻撃者は JavaScript で書かれたエクスプロイトを用意する。これを XSS 脆弱性のあるページに挿入し、ユーザのブラウザ上で実行する。まず、エクスプロイトはログアウト処理を行う。次に、iframe 内にログインページを取得する。ログインページを取得した後、テキストボックスに対し、下キーを押すイベントを生成する事で入力するユーザ名の候補を選択し、Enter キーを押すイベントを生成する事でユーザ名をテキス

トボックスに入力する。するとパスワードマネージャにより、パスワードが自動で入力されるため、エクスプロイトはパスワードを盗み出す。

3 報告

本稿では発見した Google Chrome のパスワードマネージャの脆弱性を報告する。

3.1 Google Chrome のパスワードマネージャ

Google Chrome のパスワードマネージャは、`type="password"`である input 要素と、その直前の `type="text"`である input 要素の情報を保存する。この `type="text"`である input 要素が生成したテキストボックスを、以降、ユーザ ID 入力欄と呼ぶ。

Google Chrome のパスワードマネージャは、以下の条件が成立する場合、ユーザ ID とパスワードを自動入力する。パスワードマネージャはすでにパスワードの保存を行っているとする。

- I. 入力フォームにおいては、パスワードを保存したログインページと同様に、同一の form 要素の内容の中に、`type="text"`である input 要素と `type="password"`である input 要素が連続して存在
- II. 入力フォームを含む Web ページのホストと、パスワードを保存したログインページがあるホストが同一
- III. 入力フォームの action 属性の送信先のホストと、パスワードを保存した入力フォームの action 属性の送信先のホストが同一

3.2 発見した脆弱性

条件Ⅲを満たさない場合、入力フォームを含むページを表示した時点ではパスワードは自動入力されない。しかし、ユーザ ID 入力欄をクリックすることによりユーザ ID の入力候補が表示される。その候補のなかで、パスワードマネージ

に登録されているユーザ ID と一致するものを選択すると、パスワードマネージャによりパスワードが自動入力される。

3.3 攻撃シナリオ

発見した脆弱性を利用した攻撃が成功する条件は以下のとおりである。

- i. ユーザはログインの求められる Web ページへアクセスをする。このページの URL は `http://web.site/login.html` とする。また、`login.html` の HTML スクリプトは図 1 を含む。
- ii. ユーザはログインを行うために、ユーザ ID とパスワードを入力する。その後、ユーザは入力された内容を送信する。
- iii. この時、ユーザはログイン ID とパスワードをパスワードマネージャに保存する。
- iv. `http://web.site/login.html` と同一のホストである Web ページ `http://web.site/vuln.html` は HTML インジェクション脆弱性を持つ。

ユーザは Web ページ `vuln.html` にアクセスする。攻撃者は、HTML インジェクションを利用して `vuln.html` に偽の入力フォームを埋め込む。その入力フォームの HTML スクリプトは図 5 である。

```
<form
  action="http://attacker.site/collect.password.php"
  method="post">
  <input type="text" name="username"
    id="username"><br>
  <input type="password" name="userpass"
    id="userpass"><br>
  <input type="submit" value="SEND">
</form>
```

図 5. 偽入力フォームの HTML スクリプト

ユーザのブラウザには偽の入力フォームが表示される。図 1 と図 5 の違いは、`action` 属性で指定された URL のみである。図 5 のブラウザ表示は図 2 と同じである。また、URL のホスト名も一致している。したがって、ユーザは、フィッシングが行われていることに気が付きにくい。偽の入力フォームは、3.1 の条件 I を満たすよう

に作成される。入力フォームが生成される Web ページは、パスワードを保存した Web ページと同一のホストであるため、条件 II を満たす。しかし、偽の入力フォームの `action` 属性は攻撃者が用意したパスワード収集サイトである。パスワード収集サイトのホストは、パスワードを保存した入力フォームの `action` 属性のホストとは異なる。よって偽の入力フォームは条件 III を満たさない。

ユーザがユーザ ID 入力欄をクリックすることでユーザ ID の入力候補が表示される。その候補のなかで、パスワードマネージャに登録されているユーザ ID と一致するものを選択すると、パスワードマネージャによりパスワードが自動で入力される。その後、送信ボタンをクリックする事でパスワードがパスワード収集サイトへ送信されてしまう。

3.3 別の攻撃シナリオ

発見した脆弱性を利用した攻撃が成功する条件は以下のとおりである。

- i. ユーザはログインの求められる Web ページへアクセスをする。このページの URL は `http://web.site/login.html` とする。また、`login.html` の HTML スクリプトは図 1 を含む。
- ii. ユーザはログインを行うために、ユーザ ID とパスワードを入力する。その後、ユーザは入力された内容を送信する。
- iii. この時、ユーザはログイン ID とパスワードをパスワードマネージャに保存する。
- iv. `http://web.site/login.html` と同一のホストである Web ページ `http://web.site/vuln.html` は HTML インジェクション脆弱性を持つ。

ユーザのブラウザでは、HTML インジェクションにより、偽の入力フォームが表示される。偽の入力フォームは、3.1 の条件 I を満たすように作成される。ただし、`type="password"` である `input` 要素により生成されるテキストボックスは、CSS により画面上には表示されないように作成される(図 6)。

```

<form action="http://attacker.site/collect.password.php"
method="post">
<input type="text" name="username"
id="username"><br>
<div
style="text-indent:-300000000px;overflow:hidden;">
password<input type="password" name="userpass"
id="userpass">
</div>
<input type="submit" value="SEND">
</form>

```

図 6. CSS スクリプトを使用した偽入力フォームの HTML スクリプト

また、入力フォームが生成される Web ページは、パスワードを保存した Web ページと同一のホストであるため、条件Ⅱを満たす。しかし、偽の入力フォームの action 属性は攻撃者が用意したパスワード収集サイトである。パスワード収集サイトのホストは、パスワードを保存した入力フォームの action 属性のホストとは異なる。よって偽の入力フォームは条件Ⅲを満たさない。

ユーザがユーザ ID 入力欄をクリックすることでユーザ ID の入力候補が表示される。その候補のなかで、パスワードマネージャに登録されているユーザ ID と一致するものを選択すると、パスワードマネージャによりパスワードが自動で入力される。しかし、画面上には type="password" である input 要素は表示されないため、パスワードが入力されたことは確認できない(図 7)。

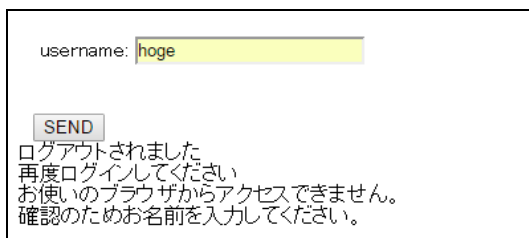


図 7. CSS スクリプトを使用した偽入力フォーム

その後、送信ボタンをクリックする事でパスワードがパスワード収集サイトへ送信されてしまう。

4 まとめ

本稿で報告した脆弱性を過去に発見された脆弱性と比較する。

脆弱性[1][2]は、入力フォームを持つ Web ページと同じホストに XSS 脆弱性を持つ Web ページを必要とする。本稿で報告された脆弱性は入力フォームの Web ページと同じホストに HTML 脆弱性を持つ Web ページを必要とする。よって本稿で報告された脆弱性の方が適用される範囲が広い。

また、[1][2]の攻撃法は JavaScript を用いているため、ブラウザで JavaScript を動作しないように設定することにより攻撃を防ぐことができる。一方で、本稿で報告した攻撃法は JavaScript を用いないため、この方法で攻撃を防ぐことはできない。

[1][2]は、いずれも XSS 脆弱性を持つ Web ページが入力フォームを持つページでなければ、iframe を用いる必要がある。よって、X-Frame-Options レスポンスヘッダに値 DENY を指定することで攻撃を防ぐことができる。

また、[1][2]の攻撃法はログアウトする必要があるが、本稿で報告された脆弱性は正規の入力フォームを含む Web ページを表示する必要はないため、ログアウトする必要は無い。一方で、過去に報告された脆弱性では JavaScript を用いているため、自動で攻撃が行われるが、本稿で報告された脆弱性はユーザの操作を必要とする。

なお、今回発見した脆弱性は Google Chrome に報告した[3]。

参考文献

[1] slideshare Breaking Browsers: Hacking Auto-Complete , <http://www.slideshare.net/jeremiahgrossman/breaking-browsers-hacking-autocomplete-blackhat-usa-2010> , Jul 29, 2010.

[2] Andrea Giammarchi, " Security Basis, and an Internet Explorer data stealer",

<http://webreflection.blogspot.jp/2008/09/security-basis-and-internet-explorer.html>, Sep 4, 2008.

[3]<https://code.google.com/p/chromium/issues/detail?id=456502>