# 密碼管理器 :攻擊與防禦

斯坦福大學的 David Silver、Suman Jana 和 Dan Boneh ;埃里克陳和
科林杰克遜 ,卡內基梅隆大學

https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/silver

這篇論文包含在第 23 屆 USENIX 安全研討會論
文集中。

2014 年 8 月 20-22 日 · 加利福尼亞州聖地亞哥

書號 978-1-931971-15-7

第 23 屆 USENIX 安全研討會論文
集的公開訪問由 USENIX 贊助

# 密碼管理器：攻擊與防禦

大衛·西爾弗·蘇曼賈娜·丹·博內
斯坦福大學

埃里克陳柯林杰克遜
卡內基·梅隆大學

抽象的

我們研究了流行密碼管理器的安全性及其自動填寫 Web 密碼的策略。
我們檢查瀏覽器內置密碼管理器、移動密碼管理器和第 3 方管理器。我們觀察到密碼管理器之間的自動填充策略存在顯著差異。一些自動填充策略可能會導致災難性的後果，即攻擊者的遠程網絡可以在不與用戶進行任何交互的情況下從用戶的密碼管理器中提取多個密碼。

我們對這些攻擊和增強密碼管理器安全性的技術進行了試驗。我們表明我們的改進可以被現有的經理採用。

## 1 簡介

隨著Web 服務的激增，普通用戶正在為大量站點設置身份驗證憑據。因此，想要在不同站點設置不同密碼的用戶被迫使用密碼管理器。許多密碼管理器可用：一些由瀏覽器供應商作為瀏覽器的一部分提供，一些由第三方提供，還有許多是基於網絡的，其中密碼被備份到雲並在用戶設備之間同步（例如 Apple 的 iCloud鑰匙鏈）。鑑於他們管理的數據的敏感性，研究他們的安全性是很自然的。

我們檢查的所有密碼管理器 (PM) 都不希望用戶在登錄頁面上手動輸入管理密碼。相反，當用戶訪問登錄頁面時，它們會自動填寫用戶名和密碼字段。

第三方密碼管理器使用瀏覽器擴展來支持自動填充。

在本文中，我們研究了四個平台上十個流行的密碼管理器的自動填充策略，並表明它們的自動填充策略都過於寬鬆：它們在不應該的情況下自動填充用戶的密碼，從而將用戶的密碼暴露給潛在的攻擊者。結果可能是災難性的：只要用戶連接到流氓 WiFi 網絡（例如咖啡店的流氓路由器），攻擊者就可以在用戶不知情或未同意的情況下從用戶的密碼管理器中提取許多密碼。基於雲的密碼同步進一步加劇了這個問題，因為攻擊者可能會提取從未在服務器上使用過的用戶密碼

被攻擊的設備。

我們的結果。我們研究密碼管理器的安全性並提出提高其安全性的方法。

- 我們首先調查了十位流行的密碼管理器如何決定何時自動填充密碼。
  不同的密碼管理器採用非常不同的自動填充策略，使用戶面臨不同的風險。

- 接下來，我們展示了自動填充策略中的許多極端情況可能會導致嚴重的攻擊，這些攻擊可以在用戶不知情的情況下遠程提取密碼，只需讓用戶連接到咖啡店的流氓路由器即可。

- 我們相信密碼管理器可以幫助加強憑證安全而不是損害它。

  在第 5 節中，我們提出了加強密碼管理器的方法，以便使用它們的用戶比手動輸入密碼的用戶更安全。

  我們在 Chrome 瀏覽器中實施了修改並報告了它們的有效性。

最後，我們討論了密碼管理器的相關工作。

一個例子。我們在論文中給出了許多密碼提取的例子，但作為熱身，我們在這裡舉一個例子。考慮通過 HTTP 提供登錄頁面但通過 HTTPS 提交用戶密碼的網站（旨在防止竊聽者讀取密碼但實際上使網站容易受到攻擊的設置）。

正如我們在第 4 節中展示的那樣，大約 17% 的 Alexa 500 強網站使用此設置。假設用戶 Alice 使用密碼管理器來保存這些站點的密碼。稍後，Alice 在咖啡店連接到流氓 WiFi 路由器。她的瀏覽器被定向到一個登陸頁面，要求她同意服務條款，這在免費 WiFi 熱點中很常見。 Alice 不知道的是，登錄頁面（如圖 1 所示）包含多個不可見的 iFrame，指向 Alice 保存密碼的網站的登錄頁面。當瀏覽器加載這些 iFrame 時，流氓路由器將 JavaScript 注入每個頁面並提取密碼管理器自動填充的密碼。

這種簡單的攻擊無需與用戶進行任何交互，就可以以每秒大約十個密碼的速度自動從密碼管理器中提取密碼。我們檢查的十個密碼管理器中有六個容易受到這種攻擊。從用戶的角度來看，她只是訪問了一個免費 WiFi 熱點的登錄頁面。沒有視覺指示表明正在進行密碼提取。

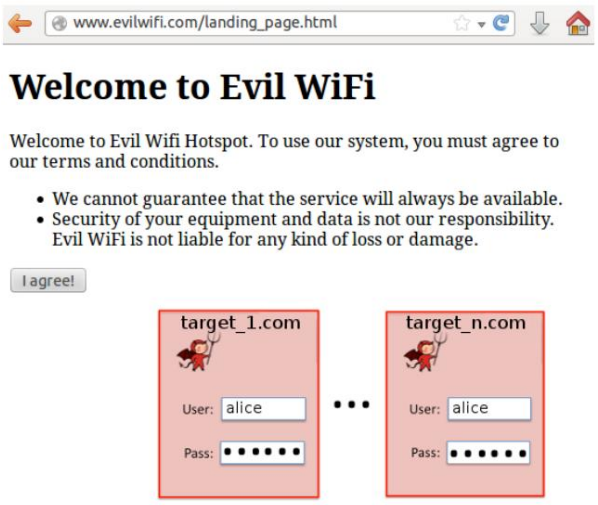


圖 1：流氓 WiFi 熱點的示例登錄頁面，其中包含指向目標站點的不可見 iFrame。請注意，iFrame 實際上對用戶是不可見的，此處顯示只是為了清楚起見。

2 密碼管理器：調查我們首先詳細調查廣泛部署的密碼管理器中實施的自動填充策略。

我們調查的密碼管理器包括：

- 桌面瀏覽器 PM： Google Chrome 34、Microsoft Internet Explorer 11、Mozilla Firefox 29 和 Apple Safari 7。
- 第 3 方 PM： 1Password [1]、LastPass [5]、Keeper [7]、Norton IdentitySafe [6] 和 KeeP ass [4]。除了 KeePass 之外，所有這些都提供了支持密碼字段自動填充的瀏覽器擴展。
- iOS PM：移動版Safari 的密碼管理器通過Apple 的iCloud Keychain 同步服務與桌面版Safari 同步。由於移動版 Safari 不支持擴展，因此 3rd Party PM 是獨立的應用程序，具有自己的內置 Web 瀏覽器。除了 Mobile Safari，我們調查了 Google Chrome、1Password 和 LastPass Tab 中的密碼管理器。
- Android PM：默認的 Android 瀏覽器和銘合金。

所有這些密碼管理器都提供“自動填充”功能，其中密碼管理器會自動填充用戶網絡瀏覽器中的用戶名和密碼字段。我們將自動填充策略分為兩大類：

- 自動自動填充：加載登錄頁面後立即填充用戶名和密碼字段，無需任何用戶交互。支持自動自動填充的密碼管理器包括 Chrome（所有平台）、Firefox、Safari、LastPass、Norton IdentitySafe 和 LastPass Tab。
- 手動自動填充：需要一些用戶交互才能自動填充。交互類型包括點擊或輸入用戶名字段、按鍵盤快捷鍵或按瀏覽器中的按鈕。始終需要手動交互的密碼管理器包括 1Password、Keeper 和 KeePass。

Internet Explorer 11 使用混合方法：它會在通過 HTTPS 加載的頁面上自動自動填充密碼，但需要用戶在通過 HTTP 加載的頁面上進行交互。

我們在第 4 節中表明，即使是這種保守行為仍然可以進行一些攻擊。

一些密碼管理器在特定情況下需要手動交互才能自動填充：

- 如果密碼字段位於 iFrame 中，則 Chrome 需要手動交互。
- Chrome 可以讀取存儲在 Mac OS X 系統範圍鑰匙鏈中的密碼，但在用戶至少手動選擇一次密碼之前不會自動自動填充它們。
- 第一次在 Mac OS X ac 上使用 Safari 或 Chrome 在系統鑰匙串中獲取密碼時，系統對話框會請求用戶的許可。如果用戶選擇“始終允許”，則不會再次顯示此對話框，並且密碼將在以後自動自動填充。如果密碼是使用 iCloud 鑰匙串從另一台設備同步的，則不會出現此對話框。
- LastPass 和 Norton IdentitySafe 提供非默認配置選項來禁用自動自動填充。在本文中，我們只討論這些密碼管理器的默認配置。

| 平台 | 密碼管理器 Mac OS X Chrome | 相同的<br>負載協議和動<br>作協議 | 不同的表單操作 | 不同的<br>表單操作完成提交時損壞 | 不同的<br>=“關閉” HTTPS | 汽車 | |
|---|---|---|---|---|---|---|---|
| 34.0.1847.137 自動 自動 | | | 無填充 | 手動的 | 汽車 | 汽車 | 無填充 |
| 10.9.3 | 火狐 29.0.1 Safari | | 無填充 | 沒有任何 | 汽車 | 無填充 | 汽車 |
| | 7.0.3 | 汽車 | 無填充 | 汽車 | 汽車 | 汽車 | 汽車 |
| 野生動物園分機。 | 1密碼4.4 | 手動的 | 手動的 | 手動的 | 手動的 | 手動的 | 手動的 |
| 野生動物園分機。 | LastPass 3.1.21 | 汽車 | 手動的 | 警告 | 汽車 | 汽車 | 汽車 |
| 野生動物園分機。 | Keeper 7.5.26 IE | 手動的 | 手動的 | 手動的 | 手動的 | 手動的 | 手動的 |
| Windows 8.1 | 11.0.9600.16531 Auto/Man No Fill KeePass 2.24 Manual Manual | | | 自動/手動 | 自動/手動 | 自動/手動 | |
| Pro IE 插件 | IdentitySafe 2014.7.0.43 Auto Auto Mobile Safari Auto | | | 手動的 | 手動的 | 手動的 | 手動的 |
| | | | | 汽車 | 汽車 | 汽車 | 汽車 |
| iOS 7.1.1 | | | 無填充 | 汽車 | 汽車 | 無填充 | 汽車 |
| | 1密碼 4.5.1 | 手動的 | 手動的 | 手動的 | 手動的 | 手動的 | 手動的 |
| | LastPass 選項卡 2.0.7 | 汽車 | 手動的 | 汽車 | 汽車 | 汽車 | 汽車 |
| | 銘 34.0.1847.18 | 汽車 | 無填充 | 無填充 | 汽車 | 無填充 | 汽車 |
| 安卓 4.3 Chrome 34.0.1847.114 自動 | | | 無填充 | 無填充 | 汽車 | 汽車 | 無填充 |
| | 安卓瀏覽器 | 汽車 | 無填充 | 汽車 | 汽車 | 無填充 | 汽車 |

表 1：Password Manager 自動填充行為（自動自動填充、手動自動填充或不填充），具體取決於協議 (http/https)、自動完成屬性、當前頁面上相對於協議使用的表單操作，以及當密碼已保存。手動自動填充是指在某些用戶交互後自動填充密碼，例如單擊或點擊其中一個表單字段。無填充意味著不會自動填充密碼。倒數第二列是指當密碼字段的自動完成屬性設置為“關閉”時的自動填充行為。最後一列是指通過錯誤的 HTTPS 連接加載的登錄頁面的自動填充行為。

## 2.1 自動填充策略

接下來，我們詢問當 PM 顯示的登錄頁面與保存密碼時的登錄頁面略有不同時會發生什麼。 PM 是否應該應用自動填充？不同的 PM 表現不同，我們調查了我們發現的不同政策。表 1 總結了我們的一些發現。

域和路徑。我們測試的所有密碼管理器都允許在與最初保存密碼的頁面相同域內的任何頁面上自動填充密碼。例如，最初保存在 https://www.example.com/aaa 上的密碼。 php 將在 https://www.example.com/ bbb.php 上填寫。這允許自動填充在多個頁面上顯示登錄表單的站點上運行，例如在所有頁面上可見的頁眉中。它還允許在移動登錄表單的站點重新設計後自動填充。

此功能意味著攻擊者可以在域中最不安全的頁面上攻擊密碼管理器（如第 4 節中所述）。這也意味著託管在同一域中的兩個站點（即 example.edu/~jdoe 和 example.edu/~jsmith)被密碼管理器視為一個站點。

協議：HTTP 與 HTTPS。假設密碼保存在通過一種協議加載的登錄頁面上（例如，HTTPS），但當前登錄頁面是通過不同的協議（比如 HTTP)加載的？ URL 的所有其他元素都相同，包括域和路徑。

密碼管理器是否應該在當前登錄頁面上自動填充密碼？

如果當前登錄頁面上的協議與保存密碼時的協議不同，Chrome、Safari、Firefox 和 Internet Explorer 都會拒絕自動填充。但是，在這種情況下，1Password、Keeper 和 LastPass 都允許在用戶交互後自動填充。請注意，LastPass 通常使用自動自動填充，因此在不同協議上降級為手動自動填充是作為一種有意識的安全措施實施的。 Norton IdentitySafe 不注意協議。它會自動填充在 HTTP 服務的頁面上保存在 HTTPS 下的密碼。正如我們稍後展示的那樣，任何形式的自動填充，無論是否手動，在協議更改時都是危險的。

修改後的表單操作。表單的 action 屬性指定表單內容在提交後將被發送到哪裡。

```
<form action= example.com method= post >
```

攻擊者竊取用戶密碼的一種方法是將登錄表單上的操作更改為

攻擊者的控制。因此，如果表單的操作與首次保存密碼時的操作不同，人們會期望密碼管理器不會自動填寫登錄表單。

我們考慮兩種不同的情況。首先，假設在加載登錄頁面時，表單的操作字段指向與首次保存密碼時不同的 URL。 Safari、Norton IdentitySafe 和 IE（在 HTTPS 頁面上）仍然會自動自動填充密碼字段。與用戶進行一些手動交互後，桌面版 Chrome 和 IE（在 HTTP 頁面上）會自動填充。 LastPass 在填寫表單之前要求用戶確認，該表單的操作指向與當前頁面不同的來源。

其次，假設在加載登錄頁面時，表單的操作字段指向正確的 URL。

但是，頁面上的 JavaScript 會修改表單操作字段，以便在提交表單時將數據發送到不同的 URL。我們測試的所有密碼管理器都允許在這種情況下提交自動填寫的表單，即使密碼被發送到錯誤的位置。我們在第 4 節中討論了這一點的含義，並在第 5 節中討論了緩解措施。

沒有自動填充功能的密碼管理器需要用戶在填寫表單之前進行交互，但沒有一個會向用戶表明表單的操作與首次保存憑據時的操作不匹配。

由於表單的操作通常對用戶不可見，因此用戶無法確定表單是否提交到了用戶想要的位置。

action 屬性對自動填充行為的影響
被捕獲在表 1 的第三和第四列中。

自動完成屬性網站可以使用自動完成屬性來建議禁用表單輸入的自動完成[3]：

<輸入自動完成=“關閉” ... >

我們發現 Firefox、Mobile Safari、默認的 An droid 瀏覽器和 Chrome 的 iOS 版本在應用於密碼輸入時尊重自動完成屬性。如果密碼字段的自動完成屬性設置為“關閉”，這些密碼管理器既不會填寫它，也不會提供保存輸入的新密碼。我們測試的所有其他密碼管理器無論如何都會填充密碼，忽略自動完成屬性的值。 LastPass 默認忽略該屬性，但提供了一個選項來尊重它。

一旦密碼管理器包含一個站點的密碼，自動完成屬性就不會影響它對本文提出的攻擊的脆弱性。如上所述在第 4 節中，在我們的設置中，攻擊者控製網絡並且可以修改登錄表單以打開密碼輸入的自動完成屬性，即使受害者網站將其關閉也是如此。

在支持的瀏覽器中，可以使用自動完成屬性來防止密碼被保存。

這可以簡單地防禦我們的攻擊，因為它們需要保存的密碼。但是，由於可用性問題，它通常不是合適的防禦措施。不保存或填寫密碼的密碼管理器不會受到用戶的歡迎。

損壞的 HTTPS 行為。假設密碼保存在通過有效 HTTPS 連接加載的登錄頁面上，但稍後訪問此登錄頁面時，生成的 HTTPS 會話被破壞，比如由於證書錯誤。用戶可以選擇忽略證書警告並訪問登錄頁面。在這種情況下，密碼管理器是否應該自動自動填充密碼？在這種情況下，桌面版和 Android 版 Chrome 拒絕自動填充密碼。 IE 從自動降級為手動自動填充。當用戶點擊 HTTPS 警告時，我們測試的所有其他密碼管理器都正常自動填充密碼。正如我們將看到的，這可能會導致重大攻擊。

修改密碼字段名稱。所有自動填充密碼管理器（LastPass 除外）都會自動填充密碼，即使登錄頁面上的密碼元素的名稱與首次保存密碼時出現的名稱不同。如第 5.2.1 節所述，在這種情況下自動填充可能會導致“自我滲漏”攻擊。 LastPass 需要手動交互，然後才能在名稱與保存密碼時不同的字段中自動填充密碼。

2.2 額外的 PM 特性

一些密碼管理器具有以下值得一提的安全功能：

iFrame 自動填充。 Norton IdentitySafe、Mobile Safari 和 LastPass Tab 不會自動填充 iFrame 中與其父頁面同源的表單。桌面 Chrome 需要手動交互才能自動填充 iFrame 中的表單，無論其來源如何。 iOS 版 Chrome 和 Android 瀏覽器永遠不會自動填充 iFrame。 Firefox、Safari 和 Chrome for Android 自動填充 iFrame 中的表單，無論來源如何。

Safari 和 Mobile Safari 只會在每個頂級頁面加載時自動填寫一個登錄表單。如果一個頁面及其所有 iFrame 具有多個登錄表單，則只有第一個會被自動填充。

我們在第 4 節中討論了這些政策對安全的影響。

能見度。 Norton IdentitySafe 不會自動自動填充不可見的表單，因為其 CSS 顯示屬性設置為“無”（直接設置或從父項繼承）。但是，它會自動填充不透明度為 0 的表單。因此，這種防禦不會增強安全性。

自動填充方法。 KeePass 在桌面密碼管理器中是獨一無二的，因為它不直接與瀏覽器集成。相反，它可以將一系列擊鍵“自動輸入”到任何活動的文本字段中。對於大多數登錄表單，這意味著它將鍵入用戶名、Tab 鍵、密碼，然後按 Enter 鍵來填充和提交表單。

自動填充並提交。 1Password、LastPass、Norton IdentitySafe 和 KeePass 提供“自動填寫和提交”功能的變體，其中密碼管理器不僅自動填寫登錄表單，而且自動提交。這使用戶無需與登錄表單的提交按鈕進行交互，從而使自動填充對用戶來說更加方便。

## 3 威脅模型

在下一節中，我們將介紹一些針對密碼管理器的攻擊，這些攻擊會從我們檢查過的所有管理器中提取密碼。首先，我們定義攻擊者的能力和目標。我們只考慮活躍的中間人網絡攻擊者，即我們假設對手可以插入和修改源自或發往用戶機器的任意網絡流量。

但是，與標準的中間人攻擊不同，我們不要求用戶在攻擊者在場的情況下登錄任何目標網站。相反，設置包括兩個階段：首先，用戶登錄到多個站點，攻擊者無法觀察或乾擾這些登錄。用戶的密碼管理器記錄了用於這些登錄的密碼。對於支持跨多台機器同步存儲密碼的密碼管理器（例如，Apple 的 iCloud KeyChain），用戶甚至可以在與最終受害者設備完全不同的設備上執行此步驟。

稍後，用戶連接到由攻擊者控制的惡意網絡，例如咖啡店中的流氓 WiFi 路由器。攻擊者可以注入、阻止和修改數據包，其目標是在用戶不執行任何操作的情況下提取存儲在用戶密碼管理器中的密碼。

我們稱這種類型的攻擊者為 tacker 的邪惡咖啡店。這些攻擊只需要暫時控製網絡路由器，而且更容易，因此更有可能在實踐中發生。我們表明，即使是中間攻擊者這樣的弱者也可以利用密碼管理器中的設計缺陷來遠程提取存儲的密碼，而無需用戶登錄任何網站。

攻擊者沒有在用戶機器上安裝任何軟件（惡意軟件）。我們僅假設存在在 Web 瀏覽器上下文中運行的密碼管理器。

## 4 遠程提取密碼

密碼管理器我們表明，邪惡的咖啡店攻擊者可以提取存儲在用戶密碼管理器中的密碼。在我們的許多攻擊中，用戶不需要與受害網站交互，也不知道正在提取密碼。我們在第 5 節討論防禦。

### 4.1 掃掠攻擊掃掠攻擊利

用自動密碼自動填充的優勢，一次竊取多個站點的憑據，而無需用戶訪問任何受害站點。對於由同步服務（例如 Apple 的 iCloud Keychain)支持的密碼管理器，即使用戶從未在該設備上訪問過網站，攻擊者也可以提取網站密碼。這些攻擊在支持自動自動填充的密碼管理器中起作用，突出了此功能的基本危險。

掃描攻擊包括三個步驟。首先，攻擊者讓用戶的瀏覽器在用戶不知情的情況下訪問目標站點的任意易受攻擊的網頁。

接下來，通過篡改網絡流量，攻擊者將 JavaScript 代碼注入易受攻擊的網頁，因為它是使用第 4.2 節中描述的方法之一通過網絡獲取的。最後，JavaScript 代碼使用第 4.3 節中的技術將密碼洩露給攻擊者。

在我們實施的掃描攻擊中，用戶連接到由攻擊者控制的 WiFi 熱點。當用戶啟動瀏覽器時，瀏覽器被重定向到標準熱點登陸頁面，要求用戶同意標準使用條款。這是公共熱點的常見行為。然而，用戶不知道的是，登錄頁麵包含實施攻擊的不可見元素。

iFrame 掃描攻擊。這裡無害的熱點登陸頁麵包含指向多個目標站點任意頁面的不可見 iFrame。當瀏覽器加載這些 iFrame 時，攻擊者使用他對路由器的控制，使用第 4.2 節中描述的方法將登錄表單和 JavaScript 注入每個 iFrame。正如我們將看到的，注入登錄表單和 JavaScript 不是

困難，可以通過幾種不同的方式完成。所需要的只是目標站點上的一些易受攻擊的頁面。
對於通過 HTTP 提供登錄頁面（但通過 HTTPS 提交密碼）的網站來說，這尤其容易，這是下一節中討論的常見設置。

當加載每個 iFrame 時，密碼管理器將自動使用用戶密碼填充相應的密碼字段。然後，每個 iFrame 中註入的 JavaScript 可以竊取和洩露這些憑據。

我們的實驗表明，這種方法可以在用戶不知情的情況下以每秒大約十個密碼的速度提取密碼。為了防止用戶在攻擊完成之前點擊登陸頁面，登陸頁麵包含一個 JavaScript 動畫進度條，強制用戶等待直到攻擊完成。

我們還發現，通過以層次結構排列 iFrame，而不是將一個 iFrame 添加到每個目標網站的頂級頁面，可以提高密碼提取過程的效率。將所有 iFrame 添加到頂級頁面會大大增加網絡流量和受害者瀏覽器使用的內存量。 iFrame 的分層排列可以避免此類訴訟。頂級 iFrame 包含大部分攻擊代碼，並動態生成子框架並將它們導航到目標頁面。此技術允許 iFrame 異步加載，從而確保網絡和內存使用在攻擊期間保持合理。

Chrome（所有平台）是唯一不容易受到基於 iFrame 的攻擊的自動自動填充密碼管理器，因為它們從不在 iFrame 中自動自動填充密碼。所有其他自動自動填充密碼管理器都容易受到這種攻擊。儘管第 2.2 節中描述的 Norton IdentitySafe、Safari、Mobile Safari 和 LastPass Tab 的自動填充策略將單次掃描中可竊取的密碼數量限制為 1，但它們仍然容易受到攻擊。

掃窗攻擊。掃描攻擊的一種變體使用窗口而不是不可見的 iFrame。如果攻擊者可以誘騙用戶禁用彈出窗口阻止程序（例如，通過要求在用戶訪問 WiFi 網絡之前打開一個窗口），登錄頁面可以在單獨的窗口中打開每個受害頁面。這比基於 iFrame 的方法更引人注目，但注入每個受害頁面的 JavaScript 可以偽裝這些窗口，以最大限度地減少被發現的機會。

偽裝窗口的技術包括最小化它們的大小、將它們移動到屏幕邊緣、隱藏頁面內容以便它們對用戶顯示為空白窗口，並在通過後立即關閉它們

詞已被盜。

幾乎所有自動自動填充密碼管理器，包括桌面 Chrome，都容易受到基於窗口的攻擊。只有 LastPass Tab 不易受攻擊，因為它根本不支持彈出窗口。因此，雖然 iFrame 使掃描攻擊更容易，但它們並不是必需的。

重定向掃描攻擊。重定向掃描攻擊無需任何 iFrame 或單獨的窗口即可提取密碼。在我們的實現中，一旦用戶連接到攻擊者控制的網絡並請求任意頁面（比如 a.com），網絡攻擊者就會響應 HTTP 重定向到目標站點上的某個易受攻擊的頁面（比如 b .com）。用戶的瀏覽器接收到重定向並發出對 b.com 頁面的請求。攻擊者允許加載頁面，但將登錄表單和 JavaScript 注入頁面，如第 4.2 節所述。注入的 JavaScript 會偽裝頁面（例如，通過隱藏其主體），以便用戶看不到正在訪問 b.com。

當用戶的瀏覽器從 b.com 加載頁面時，易受攻擊的密碼管理器將自動使用 b.com 的憑據自動填充登錄表單，然後注入的 JavaScript 可以洩露這些憑據。完成後，注入的 JavaScript 會將用戶的瀏覽器重定向到下一個受害站點（比如 c.com），並以相同的方式洩露用戶在 c.com 的密碼。當足夠多的密碼被洩露時，攻擊者會將用戶的瀏覽器重定向到用戶請求的原始頁面 (a.com)。

這種攻擊留下了密碼提取發生的小跡象。在攻擊過程中，用戶的地址欄會顯示被攻擊站點的地址，被攻擊站點將保留在用戶的歷史記錄中。但是，只要頁面主體本身經過偽裝，大多數用戶就不會注意到這些細小的視覺線索。

我們所有的自動自動填充密碼管理器
被測試的人容易受到這種攻擊。

概括。表 2 描述了哪些密碼管理器容易受到這些掃描攻擊。

通過密碼同步進行攻擊放大。大多數密碼管理器都提供在不同設備之間同步用戶密碼的服務。這些密碼同步服務可能會導致在設備從未訪問過受害站點的情況下從設備中提取密碼。

假設用戶的密碼管理器在他們的台式機和平板電腦之間同步，並且會自動自動填充從另一台設備同步的密碼而無需用戶輸入 -

| 平台 | 密碼管理器Mac OS X | iFrame 掃描窗口掃描 | 重定向掃描 | |
|---|---|---|---|---|
| 10.9.3 Chrome 34.0.1847.137 | | | + | + |
| | 火狐 29.0.1 | + | + | + |
| | 野生動物園 7.0.3 | 單身的 | + | + |
| 野生動物園分機。 | 1密碼4.4 | | | |
| 野生動物園分機。 | LastPass 3.1.21 | + | + | + |
| 野生動物園分機。 | Keeper 7.5.26 | | | |
| Windows 8.1 Pro Internet Explorer 11.0.9600.16531 KeePass 2.24 | | HTTPS | HTTPS | HTTPS |
| IE插件 | Norton IdentitySafe 2014.7.0.43 | 所以 | + | + |
| iOS 7.1.1 | Mobile Safari 1Password 4.5.1 | 單身，所以 | + | + |
| | LastPass Tab 2.0.7 Chrome | | | |
| | 34.0.1847.18 | 所以 | | + |
| | | | + | + |
| 安卓4.3 | 鎔 34.0.1847.114 | | + | + |
| | 安卓瀏覽器 | | + | + |

表 2：掃描攻擊的漏洞。 +表示不受限制的漏洞。 HTTPS表示僅在通過 HTTPS 提供的頁面上存在漏洞。單個表示單個站點在每次頂級頁面加載時易受攻擊。當包含 iFrame 的頁面與 iFrame 中的目標頁面同源時， SO表示存在漏洞。

互動。進一步假設站點 c.com 容易受到網絡攻擊，因此容易受到上述攻擊。用戶很小心，只在他們的桌面上訪問 c.com，它永遠不會離開用戶的安全家庭網絡。然而，當用戶在咖啡店將他們的平板電腦連接到攻擊者的 WiFi 網絡時，攻擊者可以對用戶的平板電腦發起掃蕩攻擊並提取用戶的 c.com 密碼，即使用戶從未訪問過 c.com藥片。

我們測試了 Apple 的 iCloud Keychain、Google Chrome Sync、Firefox Sync 和 LastPass Tab，發現它們都容易受到這種攻擊。一般來說，任何自動填充從另一台設備同步的密碼的密碼管理器都容易受到這種類型的攻擊放大。因此，任何密碼管理器的安全性僅與其同步的最弱密碼管理器的安全性一樣強。

### 4.2 注入技術

掃描攻擊依賴於攻擊者通過篡改網絡流量來修改受害站點上的頁面的能力。
當易受攻擊的頁面是登錄頁面本身時，攻擊最簡單。但是，任何與登錄頁面同源的頁面就足夠了，因為所有密碼管理器都將保存的密碼與域相關聯，而忽略登錄頁面的路徑。攻擊者可以將登錄表單註入到實際登錄頁面源中的任何頁面，並對該頁面發起密碼提取攻擊。

我們列出了一些可行的注入技術。

HTTP 登錄頁面。考慮一個通過 HTTP 提供其登錄頁面的網站，但通過 HTTP 提交登錄表單
HTTPS。雖然此設置可防止用戶密碼在提交表單時被竊聽，但咖啡店攻擊者可以輕鬆地將所需的 JavaScript 注入路由器的登錄表單，並發動上一節中討論的所有掃描攻擊。

顯然，通過 HTTP 提供登錄表單是一種不好的做法，因為它會使站點暴露於 SSLstrip 攻擊 [33]。通過 SSLstrip 提取密碼如何要求用戶在連接到攻擊者的網絡並訪問受害者頁面時主動輸入密碼。相比之下，上一節中的掃描攻擊無需任何用戶交互即可提取密碼。

為了測試此設置的普遍性 通過 HTTP 加載登錄頁面，但通過 HTTPS 提交登錄表單 我們通過手動訪問並檢查其登錄程序來調查 Alexa Top 500 網站（截至 2013 年 10 月）。在接受調查的 500 個網站中，408 個有登錄表單。這 408 個站點中有 71 個（佔 17.40%)使用 HTTP 加載登錄頁面，但使用 HTTPS 提交。這份 71 個站點的列表中有一些知名名稱，包括 ask.com、godaddy.com、reddit.com、huffingtonpost.com 和 att.com。

此外，123 個（或 30.15%)站點使用 HTTP 加載登錄頁面和提交登錄頁面。這種設置很容易受到竊聽，但易受攻擊的密碼管理器無需人工輸入密碼，從而增加了這種漏洞。

出於我們攻擊的目的，可以將這些站點視為具有通過 HTTP 提供的登錄表單的站點的特別易受攻擊的子類。

使用上一節中的掃描攻擊，可以輕鬆地從自動填充密碼管理器中提取所有這些易受攻擊網站的密碼。有人可能會爭辯說，所有這些站點都需要重新設計才能通過 HTTPS 加載和提交登錄頁面。然而，在此之前，需要加強密碼管理器以防止這些攻擊。我們在第 5 節討論防禦。

嵌入式設備 I.許多嵌入式設備默認通過 HTTP 提供登錄頁面，因為該通道被假定為受 WiFi 加密協議（如 WPA2)保護。事實上，Gourdin 等人。報告說大多數嵌入式 Web 界面仍然使用 HTTP [26]。同樣，公司網絡中的內部服務器也可以通過 HTTP 提供 Web 登錄頁面，因為只能通過虛擬專用網絡 (VPN) 訪問這些服務器。

掃描攻擊對這些設備非常有效：即使底層網絡連接不安全，密碼管理器也會自動填充密碼。通過將 JavaScript 注入 HTTP 登錄頁面，咖啡店攻擊者可以提取用戶之前與之交互過的嵌入式設備和公司服務器的密碼。

嵌入式設備二。一些家庭路由器通過 HTTPS 提供登錄頁面，但使用的是自簽名證書。攻擊者可以購買與路由器 [38] 具有相同通用名稱的有效證書，或者生成自己的自簽名證書。當用戶的機器連接到攻擊者的網絡時，攻擊者可以通過出示路由器網站的有效證書來欺騙用戶的家庭路由器。這允許攻擊者發動掃描攻擊並提取用戶的家庭路由器密碼。

損壞的 HTTPS。考慮一個通過 HTTPS 提供登錄頁面的公共站點。在第 2 節中，我們注意到許多自動填充密碼的密碼管理器會自動填充密碼，即使登錄頁面是通過斷開的 HTTPS 連接加載的，比如由於證書錯誤。

這可以在我們的重定向掃描攻擊中被利用：當瀏覽器被重定向到受害站點時，攻擊者使用該站點的自簽名證書提供修改後的登錄頁面。這個修改後的登錄頁麵包含一個登錄表單和在密碼管理器自動填充後洩露用戶密碼所需的 JavaScript。

這些自簽名證書會在瀏覽器中生成 HTTPS 警告，但如果重定向掃描攻擊作為登錄熱點過程的一部分發生，用戶就會有動機點擊生成的 HTTPS 警告消息。因此，攻擊者可以從密碼管理器中提取用戶密碼，即使對於那些登錄頁面通過 HTTPS 提供。

事實上，之前的幾項工作已經發現，用戶通常傾向於點擊 HTTPS 警告 [43、8]。用戶可能決定點擊警告並訪問該站點，但不輸入任何敏感信息。儘管如此，用戶的密碼管理器會自動填充密碼，導致攻擊者提取密碼，而不管用戶是否小心。我們測試的所有密碼管理器都會填寫密碼，即使用戶點擊了 SSL 警告，桌面版和 Android 版 Chrome 除外。

活動混合內容。任何包含通過 HTTP 獲取的活動內容（例如腳本）的 HTTPS 網頁也是一個潛在的向量。如果在用戶瀏覽器中啟用了呈現活動混合內容，則任何包含活動混合內容的 HTTPS 頁面都容易受到注入攻擊。默認情況下，Chrome、Firefox 和 IE 會阻止活動的混合內容，但會提供一個用戶選項來啟用它。 Safari、Mobile Safari 和 Android 標準瀏覽器允許獲取和執行活動的混合內容，而無需任何警告。幾種類型的活動混合內容，特別是那些由瀏覽器插件處理的內容，更難阻止。例如，如果未正確阻止，通過 HTTP 嵌入 Shockwave Flash (SWF) 文件可能會被網絡攻擊者用來注入任意腳本 [30]。

XSS 注入。頁面中的跨站點腳本漏洞允許攻擊者註入 JavaScript 以根據需要修改頁面 [24]。在賽門鐵克 [20] 的 2013 年互聯網安全威脅報告中，XSS 漏洞被列為最常見的 Web 漏洞之一。如果受害站點的任何頁面上存在 XSS 漏洞，即使該站點的登錄頁面是通過 HTTPS 提供的，掃描攻擊也會起作用。例如，攻擊者只需在鏈接到 XSS 頁面的惡意熱點登錄頁面上包含 iFrame 或重定向。該鏈接利用 XSS 漏洞將所需的登錄表單和 JavaScript 注入頁面。

此外，XSS 漏洞允許使用比我們的咖啡店攻擊者更弱的威脅模型。一個普通的網絡攻擊者可以誘使用戶訪問他的網站，然後通過 XSS漏洞發起攻擊。這種攻擊方式不需要訪問用戶的網絡，並且之前已由 RSnake [37] 和 Saltzman 等人提出。 [40]。

剩餘密碼。用戶的密碼管理器可能包含來自較舊、不太安全的站點版本的剩餘密碼。攻擊者可以欺騙舊站點以竊取剩餘的密碼。除非用戶主動刪除舊密碼，否則更新站點的安全性並不能保護域免受此類攻擊

攻擊。例如，如果用戶的密碼管理器包含 Facebook 在切換到 HTTPS 之前的密碼，則攻擊者可以欺騙 HTTP Facebook 登錄頁面來竊取密碼。

### 4.3 密碼洩露

在上一節中，我們提到了在密碼管理器自動填充用戶密碼後洩露用戶密碼的 JavaScript。一旦密碼管理器自動填寫登錄表單，攻擊者必須能夠訪問填寫的憑據並將它們發送到其控制下的服務器。我們簡要描述了實現此目的的兩種方法。

#### 4.3.1 方法#1：隱身

使用隱形滲漏，攻擊者等到登錄表單由密碼管理器自動填充用戶憑據，然後通過在不可見的 iFrame 中加載攻擊者控制的頁面並將憑據作為參數傳遞來竊取密碼。

以下簡單的 JavaScript 就是這樣做的，並且適用於我們測試過的所有密碼管理器：

```
功能測試密碼（）{
 var password =
          document.forms[0].password.value;如果（密
 碼 != “”）{ var temp = document.createElement
   （“div”）; temp.innerHTML +=

            <iFrame src=\      + attacker_addr +    ?
          password=    + password +    \
          style=\   display:none;\    />   ;
  document.body.appendChild(temp);清除間隔（間
  隔）； }} interval = setInterval(testPassword, 50);
```

#### 4.3.2 方法#2：行動

HTML 表單的“動作”是
表單的數據將被提交。攻擊者可以修改登錄表單的操作屬性，使其提交到攻擊者控制的站點，從而將用戶的憑據洩露給攻擊者。如果攻擊者將用戶的瀏覽器重定向回實際操作，用戶將不會注意到更改。

自動自動填充密碼管理器在頁面首次加載時填充密碼表單。然後，攻擊者可以使用注入的 JavaScript 來更改操作、提交表單並竊取密碼。如果登錄頁面在 iFrame 中加載或呈現為不可見，用戶甚至不會意識到已提交登錄表單。下面的簡單代碼就是這樣做的：

```
轉換器=函數（）{
  document.forms[0].action = attacker_addr;文
  檔.forms[0].submit(); } setTimeout（轉換器，1000）；
```

在 2.1 節中，我們展示了自動自動填充密碼的密碼管理器會在頁面加載時自動填充密碼，並且當提交的表單操作與首次保存密碼時的操作不同時，不會向用戶顯示警告。因此，所有具有自動填充功能的密碼管理器都容易受到這種滲漏方法的攻擊。

### 4.4 需要用戶交互的攻擊

到目前為止描述的所有攻擊都利用自動自動填充密碼管理器在用戶不與登錄表單交互時起作用。然而，無論登錄表單如何填寫，我們描述的滲出技術都有效。如果用戶的密碼管理器需要用戶輸入以填寫密碼，並且攻擊者可以在用戶沒有意識到的情況下誘騙用戶與登錄表單進行交互，則可以在填寫密碼表單後立即使用相同的滲漏技術來竊取密碼。

我們創建了一個簡單的“點擊劫持”攻擊 [29、39、31]。攻擊者向用戶呈現看似與目標站點無關的良性表單。覆蓋良性表單的是指向目標站點登錄頁面的不可見 iFrame。 iFrame 的定位使得當用戶與良性表單交互時，他們實際上與不可見的 iFrame 交互 - 在這種情況下，當用戶認為他們正在填寫良性網站上的表單時，他們實際上是在填寫密碼目標站點。一旦填滿，前面描述的任何滲透技術都可以用來竊取密碼。這種攻擊一次竊取一個站點的密碼，但可以重複竊取多個站點的密碼。

我們確認這種攻擊對 Chrome 和 Internet Explorer 11 都有效，因為至少在某些情況下兩者都需要手動交互才能進行填充。

## 5 加強密碼管理

在本節中，我們針對之前提出的攻擊提出了兩個互補的解決方案。在描述我們解決方案的細節之前，我們首先描述為什麼一些顯而易見的解決方案不起作用。例如，由於我們所有的攻擊都需要 JavaScript 注入，一個潛在的解決方案是防止密碼管理器在易受 JavaScript 注入攻擊的頁面上自動填充密碼。這個解決方案在實踐中很難實施，因為瀏覽器很難檢測到一些 JavaScript 注入向量（例如，XSS 錯誤）。另一種可能的解決方案是完全阻止 iFrame 中的自動填充。但是，此解決方案不能防止第 4 節中描述的窗口或重定向掃描攻擊。

此外，在 iFrame 中阻止自動填充會給在 iFrame 中包含登錄表單的良性網站用戶帶來不便。

5.1 強制用戶交互我們的最終目標是

確保使用密碼管理器比用戶在密碼字段中手動輸入密碼時具有更好的安全性。正如上一節的攻擊所證明的那樣，今天的密碼管理器肯定不是這種情況。我們從最簡單的防禦開始，使密碼管理器不比手動用戶輸入差。

我們最強大的攻擊利用了密碼字段的自動填充。一個明顯的防禦措施是在自動填寫表單之前總是需要一些用戶交互。這將防止在沒有任何用戶交互的情況下提取多個密碼的掃描攻擊。交互可以以鍵盤快捷鍵、單擊按鈕、從菜單中選擇條目或在用戶名字段中鍵入的形式出現。無論交互類型如何，都必須防止點擊劫持攻擊，如第 4.4 節所述。用戶交互應通過 JavaScript 無法與之交互的可信瀏覽器 UI 進行，以防止惡意 JavaScript 欺騙用戶交互並觸發自動填充。

此外，密碼管理器應該在填充發生之前顯示正在自動填充的域名，以便用戶知道正在自動填充哪個站點。這減少了用戶無意中填寫表格的機會。例如，如果一個站點的登錄頁麵包含一個不可見的 iFrame 指向另一個站點的登錄頁面，則用戶必須明確選擇他們想要填充的域。

在某些設置中，例如損壞的 HTTPS，密碼
經理應該簡單地拒絕自動填充密碼。

執行。始終強制用戶交互很容易在 Chrome1 中進行原型設計，因為 Chrome 已經準備好在某些情況下需要用戶輸入，例如當前頁面上的操作與保存密碼時的操作不同時。由於 UI 實現已經存在，我們只需始終觸發它即可。為此，我們在 PasswordFormFillData 對象的構造函數中將 wait_for_username 變量硬編碼為 true。請注意，這不能防止第 4.4 節中描述的點擊劫持攻擊，但可以擴展到這樣做。

最大限度地減少用戶的不便。由於總是在自動填充之前強制用戶交互可能會給用戶帶來不便，因此密碼管理器可以提供“自動填充並提交”功能，一旦觸發

1銘構建 231333

通過用戶交互將自動填寫登錄表單並提交。我們發現 1Password、LastPass、Norton IdentitySafe 和 KeePass 已經支持自動填充和提交的變體。

使用此功能，用戶的總體交互將與當前的手動自動填充密碼管理器類似。在密碼管理器自動填寫登錄表單後，用戶不會與提交按鈕交互，而是與密碼管理器交互以觸發自動填寫和提交。只要滿足本節前面所述的條件，使用此類功能將與手動輸入密碼一樣安全。

5.2 安全灌裝

我們的主要防禦措施稱為安全填充，旨在使密碼管理器的使用比手動輸入密碼更安全。僅僅要求用戶交互是不夠的。實際上，如果登錄頁面通過 HTTP 加載但通過 HTTPS 提交，則一旦登錄表單填寫了用戶密碼，瀏覽器或密碼管理器實現都不會提供安全性：JavaScript 可以直接從表單讀取密碼或更改表單的操作以便它提交到由 at tacker 託管的密碼竊取頁面。

安全填寫的目標是，即使攻擊者將惡意 JavaScript 注入登錄頁面，只要通過 HTTPS 提交表單，密碼管理器自動填寫的密碼將保持安全。這種防禦有點類似於 HttpOnly cookies [10]，但適用於自動填充的密碼：它們可以提交給 Web 服務器，但不能被 JavaScript 訪問。我們在本節末尾討論兼容性問題。

我們提議的防禦工作如下：

1. 與用戶名和密碼一起，密碼管理器存儲首次保存用戶名和密碼時出現在登錄表單中的操作。

2. 當密碼管理器自動填寫登錄表單時，JavaScript 無法讀取密碼字段。我們說自動填充現在“正在進行”。

3. 如果用戶名或密碼字段在自動填充過程中被修改（由用戶或 JavaScript），自動填充將中止。密碼從密碼字段中清除，密碼字段再次可由 JavaScript 讀取。

4. 一旦提交了一個正在自動填寫的表單，所有的 JavaScript 代碼都將是

run 已經運行，瀏覽器檢查表單的動作是否匹配它存儲的動作的域。

如果域不匹配，密碼字段將被刪除並且表單提交失敗。如果域確實匹配，則允許表單正常提交。

使 JavaScript 無法讀取密碼字段可防止秘密滲漏，因為惡意 JavaScript 無法讀取密碼字段，因此無法竊取密碼。在允許提交表單之前檢查操作可確保操作未更改為指向潛在的惡意站點。

保證密碼只填寫到提交到與最初保存密碼時相同的位置的表單中。為此，必須在 JavaScript（以及攻擊者)最後一次修改表單操作的機會之後執行檢查。

在表單的操作與存儲的內容不匹配的情況下，可能需要為用戶提供無論如何提交表單（和密碼)的選項。然而，瀏覽器應該允許用戶做出明智的決定，向用戶顯示新的和原始的操作，並解釋他們的密碼是如何洩露的。這會削弱安全性，因為用戶可能會選擇在不應該提交表單的時候提交表單，但是當網站進行重新設計並且登錄頁面發生變化時，它會提高兼容性。

執行。我們通過修改 PasswordAutofillAgent 類在 Chrome2 中實現了這種防禦的原型。

在裡面 FillUserNameAndPassword 方法，我們用虛擬值（一系列不可打印的字符)填充密碼字段，然後將真實密碼和表單的操作存儲在與表單關聯的 PasswordInfo 對像中。在 WillSendSubmitEvent 方法中，我們檢查密碼字段中是否仍然存在虛擬值；如果是，並且如果表單的操作與我們存儲的操作相匹配，我們將用真實密碼替換虛擬值並允許表單提交。雖然我們的實現只是一個原型，但它表明實現這種防禦相當簡單，至少在 Chrome 中是這樣。

儘管瀏覽器供應商需要在他們自己的密碼管理器中實現此功能，但他們可能會考慮為外部密碼管理器擴展提供一種機制來實現相同的功能。 API 可以允許密碼管理器擴展填寫表格並將其指定為自動填寫，以及指定表格上的預期操作。行為將與內部密碼相同

經理：密碼字段將變得無法被 JavaScript 讀取，並且瀏覽器會在提交表單之前檢查操作是否已更改。

#### 5.2.1 安全填充的局限性

安全填充方法將導致與現有站點的兼容性問題，這些站點的登錄過程依賴於使用 JavaScript 讀取密碼字段的能力。

基於 AJAX 的登錄。一些站點使用 AJAX 而不是標準表單提交來提交登錄表單。
當按下登錄表單的提交按鈕時，這些站點使用 JavaScript 讀取表單字段，然後構造並提交 XMLHttpRequest 對象。這種方法與我們的解決方案不兼容，因為 JavaScript 無法讀取填充的密碼字段，因此無法構造 XMLHttpRequest。此外，這不使用表單的操作字段，因此密碼管理器無法檢測到密碼何時被提交到與首次保存時不同的站點。

為了研究我們的提案對現有熱門網站的影響，我們在 2013 年 10 月 26 日的 Alexa Top 50 網站上尋找使用 AJAX 登錄的情況。這 50 個網站中有 10 個使用 AJAX 提交登錄。 10 個站點中有 8 個位於中國，列表中只有一個中文站點未使用 AJAX。其餘兩個站點位於俄羅斯和美國，這兩個國家的其他站點使用普通表單提交。這表明使用 AJAX 提交密碼在中國很流行，但在世界其他地方並不常見，總體而言，AJAX 被極少數流行網站使用。

我們提出了兩種解決方法，使我們的解決方案能夠與 AJAX 一起工作。首先，網站可以將登錄表單放在 iFrame 中，而不是使用 XMLHttpRequest。
iFrame 將使用標準表單提交來提交。使用這種方法，JavaScript 無需讀取表單字段，表單的操作也能正常運行。因此，它完全兼容我們的安全填充建議，但仍允許用戶異步登錄。

其次，對於必須使用 XMLHttpRequest 的站點，瀏覽器可以提供額外的 API，允許 JavaScript 在無法讀取密碼的情況下提交密碼。現有的 XMLHttpRequest API 使用 send() 方法發送數據。我們提出了一個額外的方法，sendPassword()。 sendPassword() 方法接受一個表單作為參數，並發送表單密碼字段的內容，而無需讓其他 JavaScript 可讀。為防止攻擊者使用 AJAX 洩露密碼，密碼管理器應在使用 send-

2鉻構建 231333

Password() ，目標 URL 與第一次發送填寫的密碼時的目標 URL 匹配。

儘管這些變通辦法需要對一些現有站點進行修改，但為了安全起見，這些努力是值得的。未進行必要修改的站點的唯一缺點是其用戶將無法使用某些密碼管理器。

防止自我滲透攻擊。陳等。 [17] 指出，在某些情況下，攻擊者可以使用他們所謂的“自我洩露”來提取數據。在我們的設置中，這轉化為以下潛在攻擊：如果受害站點上的任何頁面支持公共討論論壇，攻擊者可以使安全填充機制將密碼提交到論壇頁面並公開發布密碼。攻擊者稍後可以訪問公共論壇並檢索發布的密碼。由於攻擊者將登錄表單的操作更改為同一域中的另一個頁面，我們的安全填充機制將允許發送密碼。在此討論中，公共論壇可以替換為受害站點上的任何公共形式發布的數據 要使此攻擊生效，登錄頁面上的密碼字段名稱必須與登錄頁面上的文本字段名稱相同公共論壇頁面。攻擊者可以通過向瀏覽器發送具有所需名稱的登錄頁面來輕鬆實現此目的。

幸運的是，可以直接防禦這個問題：我們的安全填充機制應該只填充名稱與保存密碼時字段名稱匹配的密碼字段。此外，使用 JavaScript 動態更改名稱屬性會導致填充中止。這種防禦措施可以防止攻擊者使用任何具有名稱名稱的字段提交密碼，而不是站點本身為登錄頁面選擇的名稱名稱。這可以防止自我洩露攻擊，除了極不可能發生的事件，即受害站點上的公共論壇頁面有一個文本字段，其名稱恰好與登錄頁面上的密碼字段名稱相同。

用戶註冊頁面。我們的安全填寫提案的另一個限制是它無法提高手動輸入密碼的安全性。 HTML 不提供區分用戶註冊頁面上的密碼字段和登錄表單中的密碼字段的方法。註冊頁面經常使用 JavaScript 在提交之前評估密碼 例如，檢查密碼強度或驗證兩個密碼是否匹配。因此，註冊頁面上的 JavaScript 必須能夠訪問密碼。

這個問題有兩種解決方案。一種選擇是禁止 JavaScript 讀取任何密碼字段，並要求註冊頁面使用以編程方式使其表現得像密碼字段的常規文本字段。

在頁面上的每個擊鍵上，JavaScript 都會用星號替換該字符，就像在密碼字段中一樣。對於用戶而言，文本字段將充當密碼字段，但註冊頁面上的 JavaScript 將能夠訪問密碼。

或者，可以稍微擴展 HTML 以支持兩種類型的密碼字段，一種用於登錄，一種用於註冊。對於登錄，密碼字段不允許 JavaScript 訪問其內容以確保安全填寫。

用於註冊的 PasswordRegistration 字段允許 JavaScript 訪問其內容，但永遠不會自動填充已保存的密碼（單獨的密碼管理器功能，例如密碼生成器可以繼續工作）。

### 5.3 服務器端防禦

如果沒有密碼管理器的支持，站點如何保護自己？由於攻擊依賴於用戶密碼管理器在客戶端做出的決定，因此不可能進行完整的服務器端防禦。但是，可以使用一些現有的最佳實踐來大大減少攻擊區域：

1. 在登錄頁面和它提交的頁面上都使用 HTTPS。理想情況下，在網站的任何地方都使用 HTTPS 並啟用 HSTS（HTTP 嚴格傳輸安全)以防止頁面在 HTTP 下加載。

2、使用CSP（Content Security Policy)來阻止內聯腳本的執行，使得JavaScript直接注入登錄頁面無效。

3. 將登錄頁面託管在與站點其餘部分不同的子域中（即，login.site.com 而不是 site.com）。這限制了被認為與登錄頁面同源的頁面數量，減少了攻擊面。

這些防禦措施都不是我們描述的攻擊所特有的，而是使我們的攻擊更加困難的最佳實踐。即使有了這些防禦措施，攻擊仍然是可能的 例如，利用損壞的 HTTPS 的攻擊仍然是可行的。因此，密碼管理器實施我們描述的修復以全面抵禦攻擊仍然很重要。

## 6 相關工作

在現有密碼管理器中發現漏洞以及構建更強大的密碼身份驗證系統方面，已有多項先前的工作。我們在下面總結了它們。

密碼管理器中的漏洞： Belekno 等。 [11] 和 Gasti 等人。 [25] 調查了幾個密碼管理器，發現他們中的大多數以不安全的方式將密碼保存到設備存儲中。但是，這些攻擊的威脅模型與本文中描述的攻擊截然不同。它們要求攻擊者能夠物理訪問用戶的設備。相比之下，對於我們的攻擊，我們只考慮網絡攻擊者，這是一種比需要物理攻擊的威脅模型更弱的威脅模型

使用權。

除了自動填寫密碼外，一些密碼管理器還支持使用姓名、電話號碼等信息自動填寫表格。之前的工作 [21、35、27] 表明，攻擊者可以使用特製表格竊取自動填寫的信息。這是一種與登錄表單攻擊不同的攻擊類別，因為與登錄密碼不同，填寫到這些表單中的信息與任何特定來源無關。然而，為了完整起見，我們在附錄 A 中總結了關於針對常規表格自動填寫的攻擊的發現。

一些現有的作品 [23, 2] 已經演示了攻擊者如何使用注入的 JavaScript 在密碼管理器中竊取用戶存儲的密碼，以獲取易受 XSS 攻擊或通過 HTTP 獲取的登錄頁面。然而，與我們的攻擊不同，這些攻擊要求用戶在攻擊者在場的情況下自願訪問易受攻擊的網站。反向跨站點請求 (RCSR) [13] 漏洞利用以下事實執行網絡釣魚攻擊：多個密碼管理器將填寫密碼以登錄表單，即使該表單的操作與首次保存密碼時的操作不同。這些攻擊要求用戶單擊提交按鈕。相比之下，我們的攻擊是完全自動化的並且對用戶透明。

與我們在本文中提出的攻擊最密切相關的工作是 RSnake [37] 和 Saltzman 等人。 [40]。 RSnake [37] 推測，攻擊者可以利用表單自動填充工具，無需任何用戶輸入即可在易受 XSS 攻擊的站點中填寫表單，從而在用戶不通知的情況下提取可自動填充的信息。基礎的

想法是使用 XSS 攻擊注入 JavaScript 並洩露自動填充的信息。薩爾茲曼等人。 [40] 建議活躍的網絡攻擊者可以通過 XSS 攻擊或通過 HTTP 加載的頁面將 iFrame 注入易受腳本注入攻擊的網站登錄表單，讓密碼管理器填寫這些登錄表單，並在用戶不注意的情況下竊取這些密碼錯誤的。然而，這些作品都沒有測試攻擊。我們對漏洞能力進行了全面研究，並提出了幾種新的和不同的攻擊向量（混合內容、破壞的 SSL、嵌入式設備管理頁面等）和攻擊技術（例如重定向攻擊）。

Stock 等人也探索了使用 XSS 攻擊來竊取自動填充的密碼。 [42]。他們建議密碼管理器可以通過使用佔位符虛擬密碼自動填寫並在將登錄表單提交到遠程服務器之前將其替換為原始密碼來防止此類攻擊。在這項工作中，與 Stock 等不同，除了 XSS 攻擊之外，我們探索了幾種不同的用於竊取自動填充密碼的向量。我們還調查了幾種不同的第三方密碼管理器以及 Stock 等人分析的內置密碼管理器。

Blanchou 等人。 [12] 描述了密碼管理器瀏覽器擴展的幾個弱點，並實施了一個網絡釣魚攻擊，展示了自動填充的危險。他們沒有檢查任何內置的瀏覽器密碼管理器，也沒有考慮如何在一次攻擊中竊取來自多個站點的密碼。他們建議密碼管理器防止密碼的跨域提交（我們在本文中稱為動作洩露），但不考慮隱身洩露。

法爾等人。 [22] 展示了針對 Android 密碼管理器的攻擊。然而，他們的攻擊特定於 Android 操作系統，並且大多數依賴於惡意 Android 應用程序，而不是網絡攻擊者。

李等。 [32] 調查了第三方基於 Web 的密碼管理器和 Web 攻擊者特有的各種漏洞，然後討論了緩解策略。他們不討論瀏覽器或本機代碼密碼管理器，也不討論網絡攻擊者。

Chromium 和 Firefox 錯誤數據庫都有錯誤歸檔，以防止在 iFrame 中自動填充登錄信息 [18、16]。然而，防止在 iFrame 中自動填充密碼並不能防止第 4 節中描述的窗口掃描或重定向攻擊。在撰寫本文時，僅修復了 Chromium 錯誤。

另一個 Chromium 錯誤 [19] 試圖在用戶與登錄頁面交互後僅自動填充表單，但不一定是登錄表單。這尚未實現，但是，將交互範圍擴大到整個頁面將使攻擊者更容易發起點擊劫持攻擊。相比之下，只有在第 5 節中建議的用戶與登錄表單進行顯式交互後才進行自動填充，才能抵禦此類攻擊。

一個 Firefox 錯誤 [14] 討論了針對密碼管理器的中間人攻擊，類似於我們的重定向攻擊。另一個錯誤 [15] 表明填充的密碼詞不應該被 JavaScript 讀取。他們的方法類似於我們的安全填充，但仍然存在漏洞

能夠使用 action 屬性進行滲透。雖然這兩個 bug 都有好幾年了，但都沒有被採取行動
之上。

密碼管理器功能： Aris [9] 討論了自動完成屬性以及為什麼設置 autocomplete=off 除了糟糕的用戶體驗之外還會導致安全性差

恩斯。

安全密碼認證系統：另一項相關研究調查了設計安全密碼認證系統，該系統可以選擇強域特定密碼，而用戶干預最少 [36, 28]。這些工作背後的主要動機是最大限度地減少用戶通過網絡釣魚網站或社會工程錯誤洩露密碼所造成的損害。這些解決方案還可以防止攻擊者利用從低安全性網站竊取的重用密碼到高安全性網站。

這些工作都沒有專注於密碼的自動填充，因此無助於防止我們在本文中提出的攻擊。

還有一些研究工作構建了支持自動填充的密碼驗證系統 [45, 44]。但是，他們的主要目標是防止網絡釣魚攻擊。在本文中，我們關注現有的密碼管理器，因此不評估這些系統對我們的攻擊有多脆弱。

桑德勒等人。提出了“密碼亭”，這是一種新的安全瀏覽器控制機制，讓用戶安全地輸入密碼，這些密碼不是無法從作為主機頁面來源的一部分運行的 JavaScript 訪問的 [41]。

他們的解決方案類似於我們的安全填充防禦，但沒有考慮密碼管理器。安全填充利用密碼管理器提供密碼亭無法提供的保證，即自動填充的密碼被提交到保存它的同一來源。此外，他們的提案要求對所有用戶進行大幅度的 UI 更改，而我們的提案只需要非常小的 UI 更改，從自動到手動自動關閉。他們認為，顯著變化是一項功能，因為它使用戶更容易看到安全性，但與此同時，顯著變化將減少瀏覽器開發人員的採用，因為瀏覽器開發人員不願因更改而擾亂他們的用戶。最終，我們的兩個想法是兼容的，因為密碼亭可以擴展為與我們在本文中描述的密碼管理器一起使用。

本文的早期未發表版本僅包含結果的一個子集，作為技術報告出現在 [34] 中。

## 7 結論

在本文中，我們調查了各種密碼管理器，發現它們遵循非常不同且不一致的自動填充策略。我們展示了一個邪惡的咖啡店攻擊者如何利用這些策略在沒有任何用戶交互的情況下竊取用戶存儲的密碼。

我們還證明了密碼管理器可以通過簡單地執行以下兩個步驟來防止這些攻擊 - 在某些情況下永遠不會自動填充，例如在存在 HTTPS 證書驗證錯誤的情況下，並且要求用戶通過某種形式的受信任的瀏覽器 UI 進行交互，不受信任的 JavaScript 無法影響，之前自動填充任何密碼。最後，我們介紹了安全填充，一種使自動填充密碼管理器在某些情況下比手動輸入密碼更安全的防禦措施（例如，通過 HTTP 獲取登錄頁面但通過 HTTPS 提交）。我們希望這項工作能夠提高密碼管理器的安全性，並鼓勵開發人員採用我們的增強功能。

我們向密碼管理器供應商披露了我們的結果，促使對自動填充策略進行了多項更改。根據我們的發現，LastPass 將不再自動自動填充 iFrame 中的密碼字段，1Password 將不再提供在 HTTP 頁面上從 HTTPS 頁面填充密碼。

## 致謝

這項工作得到了
NSF、DARPA SAFER 計劃和 Suman Jana 的 Google 博士獎學金的支持。
本材料中表達的任何意見、調查結果和結論或建議均為作者的意見，不一定反映 NSF、DARPA 或 Google 的觀點。

## 參考

[1] 1password - 敏捷位。 https://agilebits.com/ 一個密碼。

[2] 使用 xss 濫用密碼管理器。 http://labs.neohapsis.com/2012/04/25/abusing password-managers-with-xss/。

[3] 自動完成屬性。 http://www.w3.org/TR/2011/WD-html5-20110525/common-input element-attributes.html#the-autocomplete 屬性。

[4] 保持密碼安全。 http://keypass.info。

[5] Lastpass 您必須記住的最後一個密碼。 https://lastpass.com。

[6] 諾頓身份安全：密碼管理器和在線身份安全。 https://identitysafe.norton.com。

[7] 安全密碼管理器 - 管理員密碼和數據保險庫 1 密碼。 https://keepersecurity.com。

[8] D. Akhawe 和 AP Felt。 Alice in warningland：瀏覽器安全警告有效性的大規模實地研究。在 USENIX 安全研討會上，2013 年。

[9] 阿里斯· 反對 autocomplete=off 的戰爭，2013 年。http://blog.0xbadc0de.be/archives/124。

[10] A.巴特。 http狀態管理機制。 RFC 2965，2011.

[11] A. Belenko 和 D. Sklyarov。智能手機上的安全密碼管理器和軍用級加密：哦，真的嗎？黑帽歐洲，2012 年。

[12] M. Blanchou 和 P. Youn。 密碼男孩年齡： 到處暴露密碼，2013 年。https://isecpartners.github.io/whitepapers/passwords/2013/11/05/Browser-Extension-Password-Managers.html。

[13] Bugzilla@Mozilla。錯誤 360493 - (cve-2006-6077) 跨站點表單 + 密碼管理器 = 安全失敗。 https://bugzilla.mozilla.org/show_bug。 cgi?id=360493。

[14] Bugzilla@Mozilla。錯誤 534541 - 來自登錄管理器的密碼可以被 mitm 攻擊者攔截（例如邪惡的 wifi 熱點或 dns 中毒）。 https://bugzilla。 mozilla.org/show_bug.cgi?id=534541。

[15] Bugzilla@Mozilla。錯誤 653132 - 自動填充的密碼字段不應具有可用於腳本）。 https://bugzilla.mozilla.org/show_bug.cgi?id=653132。

[16] Bugzilla@Mozilla。錯誤 786276 - 不要在與頂級頁面不同源的框架中自動填充密碼。 https://bugzilla.mozilla.org/show_bug。 cgi?id=786276。

[17] EY Chen、S. Gorbaty、A. Singhal 和 C. Jackson。自我滲透：瀏覽器強制信息流控制的危險。在 W2SP，2012 年。

[18] 鉻。 問題 163072： Chrome 應該僅在用戶操作後填寫保存的密碼。 https://code.google.com/p/chromium/issues/detail?id=163072。

[19] 鉻。問題 257156：不要在 iframed 內容的頁面加載時自動填充密碼。 https://code.google。 com/p/chromium/issues/detail?id=257156。

[20] S. Corp. 2013 年互聯網安全威脅報告，第 18 卷。http://www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_v18_2012_21291018.en-us.pdf。

[21] J. de Valk。為什麼你不應該使用自動完成。 https://yoast.com/autocomplete-security/。

[22] S. Fahl、M. Harbach、M. Oltrogge、T. Muders 和 M. Smith。嘿，你，離開我的剪貼板。在金融密碼學和數據安全中，第 144-161 頁。施普林格，2013 年。

[23] M.費爾克。密碼管理與 ie 和 firefox 有關，第一部分，2010 年。http://www.symantec。 com/connect/articles/password-management concerns-ie-and-firefox-part-one。

[24] S. Fogie、J. Grossman、R. Hansen、A. Rager 和 PD 佩特科夫。 Xss exploits：跨站腳本攻擊與防禦。 Syngress，2(3)，2007 年。

[25] P. Gasti 和 KB Rasmussen。關於密碼管理器數據庫格式的安全性。在 ESORICS 中。 2012.

[26] B. Gourdin、C. Soman、H. Bojinov 和 E. Bursztein。保護安全的嵌入式 Web 界面。在 USENIX 安全研討會上，2011 年。

[27] J.格羅斯曼。 我知道你的名字、工作地點和生活地點（safari v4 和 v5）。 http://jeremiahgrossman.blogspot.com/2010/07/i know-your-name-where-you-work-and.html。

[28] JA Halderman、B. Waters 和 EW Felten。一種安全管理密碼的便捷方法。在萬維網上，2005 年。

[29] R.漢森。點擊劫持。 http://ha.ckers.org/blog/20080915/clickjacking/。

[30] J. Hodges、C. Jackson 和 A. Barth。 Http 嚴格傳輸安全 (hsts)。 http://www.hjp.at/doc/rfc/rfc6797.html。

[31] L.-S。黃、A. Moshchuk、HJ Wang、S. Schechter 和 C. Jackson。點擊劫持：攻擊和防禦。在 USENIX 安全研討會上，2012 年。

[32] Z. Li、W. He、D. Akhawe 和 D. Song。皇帝的新密碼管理器：基於Web的密碼管理器的安全分析。在第 23 屆 USENIX 安全研討會 (USENIX Security 14)，2014 年 8 月。

[33] M.馬林斯派克。在實踐中擊敗 ssl 的新技巧。在黑帽 DC，2009 年。

[34] R. Gonzalez、E. Chen 和 C. Jackson。對現代密碼管理器的自動密碼提取攻擊。未發表，2013 年 9 月。arxiv.org/pdf/1309。 1416v1.pdf。

[35] RM 羅德里格斯。如何利用 chrome 自動填充功能獲取敏感信息。 http://blog.elevenpaths.com/2013/10/如何利用-of-chrome.html。

[36] B. Ross、C. Jackson、N. Miyake、D. Boneh 和 J. Mitchell。使用瀏覽器擴展程序進行更強大的密碼身份驗證。在 Usenix 安全研討會上，2005 年。

[37] 蛇。通過自動填寫表格竊取用戶信息。 http://ha.ckers.org/blog/20060821/stealing-user-information-via-automatic form-filling。

[38] 運行 SSL。 私有內部 IP 地址或本地 Intranet 服務器名稱的 SSL 證書。 http://runssl.com/members/knowledgebase/9/SSL Certificate For-Private-Internal-IP地址或本地-Intranet-Server-Name.html。

[39] G. Rydstedt、E. Bursztein、D. Boneh 和 C. Jackson。 Busting frame busting：一項關於熱門站點點擊劫持漏洞的研究。在 W2SP，2010 年。

[40] R. Saltzman 和 A. Sharabani。中間攻擊的活躍人。 OWASP 澳大利亞，2009 年。

[41] D. Sandler 和 DS Wallach。輸入類型=密碼必須死。 W2SP，第 102–113 頁，2008 年。

[42] B. Stock 和 M. Johns。保護用戶免受基於 XSS 的密碼管理器濫用。在 AsiaCCS，2014 年。

[43] J. Sunshine、S. Egelman、H. Almuhimedi、N. Atri 和 LF Cranor。狼來了：SSL 警告有效性的實證研究。在 USENIX 安全研討會上，2009 年。

[44] M. Wu、RC Miller 和 G. Little。 Web 錢包：通過揭示用戶意圖來防止網絡釣魚攻擊。在 SOUPS 中，2006 年。

[45] K.-P。 Yee 和 K. Sitaker。 Passpet：方便的密碼管理和網絡釣魚防護。在 SOUPS 中，2006 年。

表格的自動填寫我們在本文中研究

的幾個密碼管理器（Chrome、Safari、LastPass 和 1Password)也支持自動填寫表格，其中包含不同的信息，如姓名、電子郵件地址、電話號碼、信用卡號碼、到期日期等。儘管這與自動填充密碼沒有直接關係，但為了完整性，我們在本節中總結了我們的發現。

與登錄信息不同，表單的自動填充信息與任何來源無關。因此，來自任何域的表單都可以自動填充相同的信息。

為了使自動填充安全，我們研究的所有密碼管理器都需要用戶交互才能開始自動填充表單。然而，一些先前的工作已經註意到，惡意攻擊者可以創建特製的表單，這些表單只有某些無害的字段可見（例如姓名），而其他更敏感的字段（例如電話號碼)對用戶可見並且一旦用戶觸發自動填充，兩者不可見和可見字段被填充，因此攻擊者可以訪問 [21, 35]。

我們發現，雖然我們研究的所有自動填充密碼管理器在某種程度上都容易受到這種攻擊，但可以提取的敏感信息類型取決於所需用戶交互的性質觸發自動填充。與本節中的其餘部分不同，我們僅考慮 Web 攻擊者，因為自動填充信息不受任何來源的限制。

- Chrome 和 Safari： Chrome 和 Safari 都將可自動填寫的信息分為兩類 - 個人信息（例如，姓名、電子郵件地址、電話號碼、實際地址)和信用卡信息（例如，信用卡號、有效期）。要為每個類別觸發自動填充，用戶需要單擊每個類別中的一個字段並從可用的條目中選擇一個條目。因此，即使攻擊者讓用戶點擊個人信息類別中的可見字段，也不會自動填充隱藏的信用卡字段。這使得在用戶不注意的情況下，在這些密碼管理器中竊取信用信息變得更加困難。

- LastPass：與 Chrome 和 Safari 不同，為了觸發自動填充，LastPass 只需要用戶單擊頁面頂部顯示的按鈕。單擊此按鈕後，表單中的所有字段（隱藏的和可見的)都會被填充。這使得攻擊者很容易創建一個精心製作的表單，僅顯示姓名和電子郵件地址等字段，同時通過隱藏字段竊取信用卡或社會安全號碼等其他信息。

- 1Password：與LastPass 不同，1Password 要求用戶根據他們想要填寫的信息點擊不同的按鈕。因此，不可能通過隱藏所有信用卡從 1Pass word 竊取信用卡信息。但是，如果用戶想要填寫信用卡信息的合法頁面還包含一個帶有

來自第三方域的隱藏信用卡字段（例如，廣告），1Password 將在 iFrame 和主頁中填充信用卡信息，只需單擊一次，而不通知用戶。