AsiaCCS'14-Stock-Protecting Users Against XSS-based Password Manager Abuse

概述：(S) PM提供(半)自动化填充表单的功能，但都为直接将明文口令填充到DOM中，极被JS访问到，故可能遭受XSS攻击；(T) 本文旨在评估针对PM的自动化窃取凭证攻击的危害并给出一定的方案；(A) 首先给出不同的攻击模式，并给出对抗措施；评估不同PM可能遭受的漏洞，以及当前网站的表单的特性；(R) 给出一个客户端的对抗措施，并对安全性和功能特性进行评估

优点：①通过避免将口令存在DOM中避免了XSS攻击
②不仅给出攻击的理论分析，还通过实验验证了攻击的可行性
③给出了防护方案的设计与安全性分析。

问题：①该方法依赖于扩展的安全性，若数据库被攻击获取，可通过对照nonce找到口令值（如何保证存储值的完整性值得思考）
②对于提出的方案，应给出用户调研表明方案的可用性，否则又能以表明方案是易用的
③将nonce替换为口令的过程，攻击者若可监控流量(非HTTPS)则也可拿到对应的口令，应考虑攻击者恶意修改流量的情况如引发DoS攻击。

▲XSS攻击的三种类型

1) persistent XSS：攻击者的代码永久性地存储在应用的数据库中
2) reflected XSS：通过url等方式将数据以响应的形式发送给应用，以执行恶意代码
3) Dom-based XSS：基于用客户端代码的漏洞。

▲Same-Origin Policy：保证两个document对象只有在origin匹配才可以交互，即protocol, domain和port要保证相同。XSS攻击则是攻击者将HTML或脚本代码注入到页面中执行，即绕过同源策略

▲Attacks

1) 通过XSS攻击获取口令值：JS可以访问表单中各个字段的数据，攻击者可以通过XSS攻击注入脚本获取口令，要求的场景是用户已输入口令但未提交。
2) 利用PM进行自动化攻击：利用PM的自动填充表单的特性，用户无需参与
3) 特定的攻击模式(尝试向用户的界面中注入表单并导出浏览器自动填充的值) 4个角度。
① Matching requirements for the URL and form
PM在匹配到domain时即可填充，可能会根据字段名或类型，但可能不关心内容，
→此时攻击者可以通过注入表单自动获取浏览器填充的内容。
② Viewports (top frame, sub frame或popup window)。
若PM需匹配特定的URL，若不匹配正确的viewports，攻击者仍可通过注入表单(iframe或popup window)完成攻击。
③ User interaction (填充前用户是否必须与PM交互) ⇒ 完全自动化的XSS攻击不可行
攻击者可以通过Clickjacking Attack诱导用户点击按钮填充表单
④ Adherence to the autocomplete attribute.
W3C标准中，浏览器不能存储autocomplete=off的数据，若PM在存储时不根据此属性而只是填充时根据此属性，则仍可能受到攻击(例如输入一个无autocomplete的表单)
4) Network-based Attacks：攻击者位于网络中，可以向网站中添加内容。Gonzalez提出Lupin工具可以向网站添加iframe，但要求网站不使用HTTPS。

▲改进策略(现有的策略，有一定的缺点)

1) 转向替代的认证方法：Server-side Mitigation
① HTTP authentication：为用户提供一个专门的对话框，通过401 HTTP响应，服务器通过pull获取
② Client-side authentication：在deployment, usability和portability方面存在不足
2) 将口令表单转移到专门的sub domain，通过同源策略保护，但需多一层域部署开销变大。
3) 强制autocomplete=off，这将使得PM无法工作

2、End-user Protection
1) 直接关闭浏览器的PM功能 2) 使用需显式用户交互的第三方PM，交互应处于攻击页面的DOM外
→第三方PM的关注点更多的为安全性，而非易用性或透明性 ☆

▲攻击的实际效果。

■针对Password Manager (Chrome, Firefox, Opera, Safari, IE, 傲游浏览器Maxthon)
1) Filling only in the top frame. 除了IE，其余均在伪造的top frame上填充用户名口令
2) Explicit user interaction：只有IE需要用户点击下拉菜单中的表项，且该对话框在DOM之外
3) URL Matching：若PM采用精确的URL匹配，则攻击者只能通过pop-up window或iframe完成，但易被发现(iframe的不可见在X-Frame-Options的headers不可用，但本文研究发现仅有8.9%的网站用了这个header)，本文旨在确定在URL不完全匹配时是否能正确填充：①不同的协议，HTTP vs HTTPS ②不同的端口 ③不同的(子)域 ④改变的path
→Chrome：更改协议、端口或子域，将不再填充，但不同的path仍可填充；Firefox, Opera和Safari行为相似
IE：在4种情况变更时均不会填充。 Maxthon傲游：只需要二级域名匹配已即可填充。
4) Form matching (浏览器真正会存储的数据)：①移除表单的form和action ②移除所有字段的name ③修改所有字段的name ④将所有字段的type设置成text. ⑤录制单的表单，任意的name，无action和method，一个type为password，⑥将⑤中到autocomplete置为off，
→Chrome：不区分action、method、name，但不提供name时不填充凭证；在存储时无视autocomplete，故使用⑤中的表单即可完成攻击！
Firefox与Chrome不同的是，当name不在在时也会填充，且Firefox遵从autocomplete的规则，在⑤的情况下仍可以完成攻击；IE需用户进行交互，行为与Firefox相同；傲游：在无name或autocomplete为off时均会填充

■针对网站的口令字段：抽取了top 4000 Alexa sites网站的口令字段进行评估
1) 查找过程中，若无登录框或口令框，则检查页面中的hyperlink元素，
2) 记录登录表单对应的url, action, autocomplete和X-Frame-Option-headers
且通过get方法查看客户端对口令字段的修改
3) 采用user script模仿用户输入口令完成提交的过程。

⇒ ■结果：4000个domain共检测到2142个。
1) 只有293个domain未存储也未填充凭证(autocomplete=off)。
2) 为了对抗Clickjacking attack，需X-Frame-Options设置成DENY，有189个domain。
3) 325个口令字段，JS具有读访问的权限。 (117个网站通过HTTPS传输，9%以未加密方式传输)
4) 824个domain使用HTTPS，剩余的1289个domain可能受到network-based attacker的攻击

■评估：由于XSS攻击可以在HTTPS和HTTP执行，因此仅有的一道防线为autocomplete功能特性，除了IE，都会自动填充表单，为攻击者完成攻击节约了成本。

▲客户端的保护方法。

1）问题的根原因是PM的视点与实现不一致。

帮用户登录→凭证发送到登录页。R需传输至登录页后发送时为正确的吃值。

2）方案：PM只向表单中插入随机数，R有提交时才将吃替换回去。

①：需严格匹配吃字段的name属性，以及相应的POST值。

3）实现（Firefox的扩展）。明文存储在扩展的沙盒中。

①当凭证存储到PM时，Firefox会给出提示，包括用户名和吃，吃提交的origin，以及吃和用户字段的name。

②nonce与相应的吃条目均存储在数据库中，用于检索和替换。

③在GET请求中不允许替换nonce，只允许在POST中替换。

4）安全性评估

①吃在PM存储后，不会再传输到Web document上中

②执行了严格的匹配约束，只有相同的domain和吃参数名称填充。

5）功能评估。

从用户角度看无差别，但无法抵抗在传输前客户端外理吃值或通过GET传输的情况