WWW'05-J.Alex Halderman-A Convenient Method for Securely Managing Passwords

(S)即使选用了较强的口令，口令重用行为仍然会带来较大的安全威胁。(T)旨在为每一个网站帐户提供唯一的口令值。(A)设计hash-based方案：Password Multiplier，通过two-level的哈希函数，可以抵抗前人设计可能遭受的离线字典攻击(通过first-level的高迭代次数，使攻击猜测成本增加)，用户可以缓存该中间值(防止反复运算)，对方案的构造和安全性进行了详尽的分析。(R)与三种方案进行了可用性和安全性对比，表明Password Multiplier安全性和可用性较好，并给出了实现。

优点：①通过two-level的方式抵抗离线暴力攻击。②给出了不需要修改mpw的修改网站口令的方式。

问题：①在4.1可用性分析对PwdHash的分析有误，PwdHash也需要输入空格或特殊按键来唤醒

②没有讨论如何满足不同网站的口令策略。

③应提供与实际的浏览器的PM的可用性、安全的比较。

1. 本文设计浏览器集成方案的原因

1> 大多数用户口令将用于网站，可减少用户负担，能增透明性和便利性，增加系统的采用率。

2> 可以为用户提供spoofing和phishing等attack的防御方法。

2. 本文方案设计(专注于使猜测更耗费时间进而增加暴力破解的代价)

▲增加计算哈希值的总时间 → 采用key stretching，例：$f^3(x)=f(f(f(x)))$

当攻击者在每个猜测上执行3次时，用户也需要进行同样的操作，因此需要在可用性和安全性上达到一个平衡。

1>构造(使用two-level的迭代哈希算法)

First-level在用户使用机器(首次)运算，时间相对较长，结果被缓存。

Second-level用于计算网站口令，包含网站或帐户名称。

$V=f^{k_1}$(username:master_password) ↑仍加盐值，作为salt，防止彩虹表攻击！▲ V可以保存作为中间值

site-password$=f^{k_2}$(site-name:master_password:V) > 保证唯一

▲要求k_1比k_2大很多，使得第1步耗时很长(数100s)，但第2步时间短，通过缓存中间值平衡了安全性和可用性。

2> Security Analysis (4种具有不同信息的敌手)

①No Information (已知用户名和目标网站)：敌手只能执行在线猜测且即使成功了也只能登录1个网站的口令值，且在线猜测的行为很容易被检测到。

②Stolen Site Password (已知至少1个网站的口令)：敌手需要执行$p(k_1+k_2)>pk_1$次哈希运算来确定正确的master password值。(在实际最常见的攻击场景) 离线

③Stolen Cache (敌手获取到了V值)：可以在已知V的情况下猜测master password，但是由于$k_1>>k_2$，故该猜测不会特别有效，且当用户发现V值泄漏(多为系统被破坏)时，可以及时修改口令。离线

④Stolen Cache and site Password (已知V和至少一个网站的口令)，敌手只需要执行迭代了k_2次的口令哈希运算(但仍比传统哈希耗时长)。→应避免网站口令存在本地，防止在V泄 离线 被攻破得

3>Iterated Hashing

本文方案的安全性基于敌手必须进行$n\cdot k$次哈希才能搜索完大小为k的口令集，将哈希函数认为是一个ROM

▲Kelsey的研究表明敌手可以花费2^m计算2^m次迭代哈希以普通生日攻击$\frac{1}{2}$倍的速度找到哈希碰撞，但在实际中，敌手无法均摊在n个口令的k次迭代，速度可能仅为$\frac{1}{n}$倍

3. 比较(从可用性和安全性与Password Safe, LPWA和PwdHash比较.

1>Usability (创建、查找口令，更改口令，迁移口令)

①Typical Usage：PwdHash在用户视角几乎与原始方案相同，本文方案与Password Safe需要用户与特定应用程序交互，LPWA需输入特定字符提示系统填充口令。

LPWA和PwdHash只保护网站生成的口令，本文可保护用户选择的口令，Password Safe还可保护PIN码等。

②Password Changes and Migration:

由于口令是生成的而非存储，故只能计算修改网站口令(特别是因为安全事件或定期更新来修改口令时，可能需要一大批口令) → PwdHash，本文方案，LPWA

Password Safe由于存储口令，故容易完成更改和迁移口令。

③Transportability (可以随时随地访问)，

一方面Password Safe只能在口令存储的位置访问网站，故不具有可移植性。

另一方面，hash-based方法一般提供可移植性，但LPWA还有另外两者开源，可移植性更好。

2> Security (比较后3种离线攻击的抵抗能力)：

▲定量分析：认为敌手愿意花费1年时间，计算让敌手花费至少半时间的口令熵的bit.

通过对比②③④攻击场景下敌手需要的猜测时间(一年可执行的猜测数)，计算抵抗攻击所需口令熵的bit数，表明本文方案更安全，即所需bit更少(②19.2bits ③19.2bit, ④25.9 bit 且本文方案可通过调整k_1, k_2保证安全性。

4. 实现

1>测试性能后，选择$k_1=10^8$, $k_2=10^5$

2>激活该应用程序的方法：双击口令字段或focus后Alt+P

3>Password Multiplier支持HTML form password inputs和standard HTTP authentication prompts，可以与浏览器内部的PM正确交互

4>生成口令的熵大约为47bit

5>口令更改

在name后附加用户记忆的一串数字，例如日期(特别是周期性更改)