USA14-Florencio-An Administrator's Guide to Internet Password Research

概述：(S)目前针对口令的研究很多，但很少有研究直接帮助增强web服务或者策略，各个作者在通过充足的证据和扎实的推理给出结论。(A)通过 literature survey 和 ground-up first-principles reasoning，标识出什么可奏效，什么不奏效以及什么未知。(R)将网站分为5类；指出在线和离线猜测抵抗次数的gap；并指出需根据 website scale 决定 engineering/user effort，抵抗离线猜测应server端部署

优点：①通过充足的证据给出了许多确定的结论，许多论述方法值得借鉴

②引用大量参考文献佐证，对参考学习

问题：①论述广度优先的 online guessing 时，未给出假设"4个月"的理由，或应说明只是例子

②文章重点考虑漫步猜测，实际定向猜测攻击具有更大的安全威胁，应加以讨论之中

③对口令策略的推断不够准确，应是更多为 online guessing resistance 设计的，而非离线的猜测

▲动机

1> 在以下的条件下如何保护/改保护的网站，很少有研究给出具体的结论

①只有有限的资源 ②对于真实威胁的理解不够深入 ③用户对某些举措强烈反对

2> 真正的问题不是缺少指导文件，而是这些指导性的文件缺乏支持性的依据

→ 近些年对泄漏的口令数据集的研究表明长期确信的内容是存在问题的

3> 口令不太可能消失，因此管理口令的挑战也不会消失

▲将帐户分类（无法要求用户采纳所有安全建议，使用户对所有帐户安全付出同样精力是不明智的）

↑促使

1> 不同类别对应需要不同级别的安全性，策略制定者应确定网站的类别以及从用户视角的网站的类别

2> 主要区分的问题是：用户会花费时间恢复丢失的帐户还是直接创建新的；以及用户是否会花费精力记住

3> 基于 potential consequences of account compromise 给出如下的分类：

① don't care：对用户无影响

② low-consequence：对用户无严重影响，不常使用的帐户

③ medium-consequence：有一定有限的影响

④ high-consequence：与就业金融健康相关且重要帐户，不易修复

⑤ ultra-sensitive：造成重大、影响生活且不可逆的危害（通过口令无法解决）

本文指的三种

口令策略 用户建议 实现细节 安全级别

对所有类型的帐户使用同样的术语 password，并为用户提供一系列休且同的口令建议，另外还有跨类别重用口令会降低安全性。建议为用户提供更细粒度的术语

▲猜测攻击和口令存储（由于之前数据集有限，对口令行为的实际研究很难开展，近期收集了5份数据集）

1> 口令的强度：ideal and actual

① 理想情况下，认为口令随机选取，则从C个字符的表中构成长度为L的口令，共有C^L个可能的口令

保守选择 L=16，C=10，每秒猜千万次也需一年的时间

② 实际用户选择的口令的构成规则：common words，proper nouns，predictable sequences

③ 适用于理想情况的估计方法不适用，例如 entropy，常见的口令比罕见口令的熵更高，口令破解工具也可证明口令熵与抗猜测能力关系很小

④ 其余的方法包括：

(i) 使用口令破解工具，包括 Hashcat 和基于 context-free grammar 的方法，模仿常见的用户策略

(ii) 可仿问真实口令分布X并按照可能性顺序猜测，产生了 partial guessing metrics，例如 α-guesswork，衡量攻击者达到α成功率时需对每个帐户平均发起的猜测数目，在真实的偏态分布下，此类猜测方法所需次数很少，而对均匀分布，仍需对应比例的猜测数（偏态分布下猜测数小很多达到指定成功率的猜测数更小）

⑤ 本文探索口令的 guessing resistance 属性

2> Online and offline guessing

攻击者可执行的猜测数取决于：(i) 攻击的位置 (ii) 口令存储的方式

攻击位置：client，network，web server（online），public facing part and backend（offline）。猜测攻击者

① offline attack 需满足的三个条件：(i) 绕开防御获取口令文件 (ii) 过程未被检测到 (iii) 哈加盐哈希

② 口令文件的存储方式（若非加盐哈希，则离线猜测要么是不必要的，要么是不可能的）

存储

(i) plaintext - 可直接获取口令明文 (ii) hashed but unsalted - 可采用 rainbow table attack

(iii) salted and hashed - 有必要执行离线猜测 (iv) reversibly encrypted：无解密key不可猜测有key可直接得到明文，即离线猜测无key时不可行，有key时没必要

③ 加盐可以抵抗彩虹表等预计算的攻击

3> 口令需抵抗的猜测数

① Online guessing (breadth-first) → 4个月，每秒发送1次猜测 $\approx 10^7$

攻击若采用广度优先，则会每s向每个帐户进行猜测，假设用户每天尝试k次，并有5%几率设则单个攻击者每秒对每个帐户产生的流量 $86000/k$ 倍，错误数为 1.73×10^6 倍。

显然，10^7 作为上界不可行，每个帐户抵抗 10^4 猜测数更实际。

② Online guessing (depth-first)

(i) 大多数帐户不足以成为 targeted effort (ii) 10^7 意味着 server 无 lockout 和 rate limiting

(iii) 通过真实数据集观察，从初始到 10^6 次猜测，成功率下降5个数量级

(iv) 即黑名单将使得 10^7 不可行（面向1个帐户）

⇒ 10^7 不可行，指出 10^6 是一个更合适的上界，与历史研究结果相似。

→ 不会随着硬件的改善而改变

③ Offline guessing

在不考虑 iterated hashing 的情况下，GPU 可产生4个月可产生 10^{17} 猜测

硬件的提高将提升抗猜测数的上界，作者指出下界至少为 10^{14}

（以及猜测算法的增强）

④ Online-offline gap

抵抗在线和离线猜测的上界和下界之间相差3到7个数量级，

在 10^3 以下，在线猜测的风险下降很快，且到 10^6 时可抵抗在线猜测，但直到 10^{14} 才可还抵抗离线猜测（10^{20} 可抵抗离线猜测），在 10^6~10^{14} 之间，增长口令可抵抗的猜测数对安全性无提高（指面向猜测攻击）

→目前,许多网站不能希望于用户口令可抵抗离线猜测。但即使网站采用了口令策略并给出反馈,当前用户的口令通常无法抵抗在线猜测攻击.

4> Storage and stretching of passwords

①使用salt可提供额外的保护.

②由于猜测是资源密集的,可以通过iteration hash来缓解(key stretching)
不应对用户产生较大的延迟,且iteration hash可减小 online-offline chasm.

③ key stretching的实例通过adaptive key derivation functions实现,例如 bcrypt、PBKDF2、scrypt.

④ keyed hashing:可以通过MAC做进一步保护,即使key丢失,敌手仍需离线猜测.

5> 许多网站未采用加盐哈希存储:原因包括 confusion,不理解好处,软件或协议的默认设置等.

▲本文的部分观点过于确定,存在不妥

①认为online-offline之间gap过大,大于10^6而小于10^{14}的抵抗猜测数未提供保护,但文件泄漏且及时检测到时,攻击者只有少部分时间可执行攻击.故仍可提供保护作用.

②认为当口令不是哈希加盐存储时使用强的口令策略无用,实际上强的口令策略可以抵抗在线猜测攻击.

③认为抵抗在线猜测较简单,实际上在线猜测很大程度依赖认信息,仅限制次数效果不佳

6> 可抵抗离线猜测的方法.

①保护口令文件: 使得计算哈希需额外的随机元素,攻击者无法以超出正常认证的速度猜测

②检测泄漏: honeywords,与用户真实口令不可区分的加盐哈希值,若敌手使用猜测到的honeyword认证,则会警告管理员口令文件已泄漏

▲口令策略和系统防御.

1> 构造和长度策略

①无法通过泄漏的数据集来看口令策略是否真的奏效.

②口令构造策略在抵抗离线猜测方面不够好,即使口令策略使得半数的用户跨越了online-offline的分界点,而且许多管理员会认为只要半数口令被敌手获取就认为100%破解.

③口令策略可以增加对在线猜测的抵抗,但可能使用户不满,且似乎对用户的努力回报较少

2> Blacklists and proactive checking

①使用黑名单简单的屏蔽常见的口令可以帮助缓解在线猜测,但想达到抵抗离线猜测不实际.

②使用黑名单的一个优点在于只会使部分处于危险中的用户不便,而口令构造策略则会影响所有用户

③静态黑名单的一个问题是会有新的口令变流行.

④另一种方法是设置一定的threshold使得网站中的口令数不会超过threshold这个值.

3> Expiration policies (password aging)

①强制口令定期更新是NIST推荐的建议,在企业和大学中常见,但在普通的网站中少见.
目的是减少攻击者攻击口令的时间,并减少攻击者利用帐户的时间

附带的两个好处是使得用户选择不同的口令选择策略,以及减少口令重用

②但收益有限 (i)离线猜测只是一种攻击的方式,若口令通过keylogging-malware获取,则更改口令无用. (ii)即使是离线猜测,过期策略效果不大,新口令可能由旧口令简单修改得到.

→可在high consequence使用该机制

4>.Rate-limiting and lockout policies.

①限制在线猜测的一种方法是implement lockout policy.在失败几次后锁定帐户一段时间或用特定方式unlock
另一种方法是在失败后增加再次尝试的delay.(两种方法在low和medium场景中很少见)

②主要问题
(i).resulting usability burden (ii) the denial of service vulnerability
可能被攻击者利用使得部分用户无法访问服务

③限制在线猜测并同时避免故意的服务封锁的一种方式是将帐户与常用的设备绑定,若从其它设备登录则需口令和CAPTCHA.(潜在的缺点是CAPTCHA的可用性)

5> Password meter effectiveness

①许多大网站使用password meter,目的是评估口令强度,促使用户使用更强的口令

②通常通过JS实现,但这严重限制了强度评估算法实现的复杂
许多评估方法无法体现口令的抗猜测能力,甚至存在不一致的现象

③若可提供与他人口令强度的对比,比单纯给出强度值的效果更好

④看到meter的用户更可能选择强口令,但与meter类型无关.

6> Backup questions & reset mechanisms

①在大多数网站中均提供另一种通信渠道用于重置口令,例如通过邮件发送重置链接或临时口令

②另一种方法是通过备份认证问题,一方面防止攻击者获取电脑访问权重置口令,另一方面对于高价值网站,这些问题应进一步认证用户,中低价值应注意用户使用这些问题耗费的精力.

③当email,phone等不可用时,备份问题常用,但答案的猜测空间很小,或容易被找到.

7> Phishing :(钓鱼攻击是一种社会工程攻击)

①通常可以分为两种: (i)通过垃圾邮件完成分散式的钓鱼攻击 (ii).针对个人或组织的spear phishing

②利用用户难以区分合法网站和欺骗网站

③分散式钓鱼攻击,由浏览器提供的保护已取得了较大进步;spear phishing仍是一个巨大的问题

8> Re-using email address as username

许多网站鼓励用户使用邮箱作为用户名,可能鼓励用户重用邮箱口令甚至被用于钓鱼攻击
另外也可能泄漏用户的隐私信息

▲服务端和客户端的防御.

1> 服务器需要工程负担,客户端需用户负担. 2> 需根据server和client重要价值的不同采取措施