**超越凭证填充：使用神经网络的密码相似性模型**

**摘要：**

**使用从一个网站泄漏的密码来破坏其他网站上关联账户的情况越来越常见，攻击者之所以能够进行这种有针对性的攻击，是因为用户在不同的网站上重复使用相同或者相似的密码。我们将针对性猜测攻击中的核心技术挑战重新定义为密码相似性建模任务。我们展示了如何使用泄漏的14亿邮箱账户和密码来学习密码相似性模型。**

**在使用我们训练以后的密码相似性模型的情况下，我们对此展开了目前为止最具破坏性的针对性攻击。模拟表明，即使使用了最新的对策，只要攻击者知道他们的其他密码之一，我们的攻击就会在不到一千个猜测中入侵超过16％的用户的账户。我们通过大型大学认证服务的案例研究表明，这种攻击在实践中也是有效的。我们将继续通过个性化密码强度计（PPSMs）提出针对这种定向攻击的首次防御。这种密​​码强度值可以在用户选择容易受到攻击的密码时向用户发出警告，包括利用用户先前被盗用并且容易被攻击的密码。我们设计并构建了一个PPSM，可以将其压缩到小于3 MB使其易于部署，以便针对所有已知的猜测攻击来准确估计密码的强度。**

**Ⅰ.介绍**

尽管反复呼吁用完全不同的身份验证机制替换密码[1]-[4]，但今天密码仍然广泛使用，并有很大可能性在将来继续使用，尽管它们易于猜测[5]，难以记住[6]和难以正确键入[7]。因为密码难以记住和键入，所以人们才会在不同网站重复使用相似的密码：将近40％的用户重复使用相同或相似的密码[8]。

密码重用和不断增加的密码泄漏使有针对性的猜测攻击成为越来越严重的威胁。有针对性攻击的最普遍形式为凭据填充，其中攻击者只是尝试使用在泄漏中发现的与该用户相关联的密码来登录用户的账户。到2017年底，互联网上已经分发了超过50亿个泄漏的账户[9],[10]；机器人驱动的凭证填充攻击占据全球一些最大网站的登录流量的90％[11]；这些攻击是账户接管的最大来源[11]。

网站运营商有时会借助第三方服务（例如HIBP[9]）帮助重置用户密码（如果发现其用户名或密码遭到破坏）。NIST现在正积极推行这种保护措施，即用户可以选择一些较小的被破坏密码作为其密码，但它只能防止凭据填充[12]。少数学术著作研究了凭证填充的一般性，根据处理规则[13]或概率性上下文无关文法（PCFG）[14]选择泄漏的密码变体。研究表明，即使在用户重置密码后，这种有针对性的攻击仍可能会造成破坏，因为用户仍具有倾向于选择相似密码的趋势[15]。我们使用凭据调整来应对提交泄露密码变体的攻击。

在这项工作中，我们将从人们选择的密码之间所具有的相似性角度研究凭证填充攻击。我们探索使用现代机器学习技术对相似性进行建模的数据驱动方法。这样就产生了一种新的有针对性的密码猜测攻击，其性能优于以前的所有攻击，以及一种新型的密码强度计的设计，该强度计在强度估算中包括了针对目标攻击的脆弱性。

简而言之，我们通过学习估计模型来处理相似性，其中是一个站点泄漏的密码，表示用户在另一网站选择的口令。然后，我们将会估计这组条件概率分布（每个对应一个）作为学习任务，在这里我们使用包含14亿泄漏邮箱账户和密码的集合。我们探索了各种启发式方法来识别数据集中单个用户使用的密码，最后将获取大量有关密码相似性的数据。

我们首先使用该数据集来学习一种紧凑的生成模型，该模型使用序列到序列（seq2seq）算法[16]捕获所有的。这种方法在自然语言处理文献中广泛用于语言翻译和其他任务。在这里，我们将输入的“源”密码视为，并且模型学习如何生成新密码，以反映数据中看到的相似模式。但是，以这种方式使用seq2seq会导致结果不优于以前的攻击。因此我们采用了不同的方法，训练模型用来预测将转换为w所需要做的修改。虽然看似等效，但事实证明这种方法更为有效。直观地，它使模型更好地集中于学习数据中发现的常见转换。我们将生成的算法称为密码路径（pass2path），该路径表示转换的顺序。

使用pass2path模型，我们建立了一个凭证调整攻击，通过模拟可以证明，如果他们另一个账户的密码出现了漏洞，该攻击可以在不到一千个猜测中破坏超过48％的用户账户。用于凭证调整攻击的基准算法可以猜测泄漏的密码，大约有40％是由于密码重用造成的。因此需要关注的是在凭据填充对策的情况下，我们的攻击效果如何。我们针对该情况执行（分离）模拟，该模拟表明我们的攻击可能会破坏16％的用户账户。这是以前最佳针对性攻击的1.2倍，是最佳的非针对性攻击的3倍。

模拟可能无法准确地代表真实的效果，因此，我们通过与康奈尔大学（Cornell University）IT安全办公室（ITSO）合作，评估了对真实系统的凭据调整攻击。ITSO部署了凭证填充对策以及其他最新的防御措施。但是，基于pass2path的凭据调整攻击成功地在1000次尝试中猜中了泄露的15665个活跃的Cornell用户账户中超过8.4％的密码。我们在这里进行的实验不仅证实了在实践中进行凭据调整攻击的威胁性，而且还帮助我们领先于攻击者一步，并确定了成千上万个存在危险的Cornell账户并进行特殊监控。然而，强迫这些用户选择新密码并不一定会阻止攻击，因为他们最终可能会选择以前的密码的变体。

因此，我们引入了个性化密码强度计（PPSM）。这些参数会考虑用户的其他（泄露的）密码来估算密码的强度（不可猜测性）。我们使用基于神经网络的词嵌入技术[17],[18],创建了一个称为vec-ppsm的PPSM，它代表了另一种建模密码相似性的方法，该方法比pass2path更适合用作强度表。我们的PPSM可以在面对有针对性的猜测攻击时识别不安全的密码，并且可以与现有的密码强度计结合使用，以针对所有已知攻击提供准确的密码强度估计。同时在文章主体中我们讨论了vec-ppsm的各种部署设置。

总而言之，我们的贡献包括以下内容：

• 我们将针对性猜测攻击中的核心技术挑战重新定义为密码相似性建模任务。这种观点使我们能够适应最先进的机器学习工具，并将其应用在公开可用的数十亿泄漏凭证。我们设计了一个模型pass2path，它可以准确地生成给定泄露密码的用户可能选择的转换。

• 使用pass2path，我们构建了迄今为止最有效的针对性密码猜测攻击。 仅在1000次猜测中，它就可以保护16％的用户账户免受凭据填充的侵害。

• 我们在实践中首次衡量了有针对性的攻击，结果显示，尽管康奈尔大学(Cornell University)采取了应对凭证填充的对策，但仍有1316个正在使用的账户可能被我们的证书篡改攻击攻破。

• 我们介绍了个性化密码强度计(PPSM)的概念。我们使用词嵌入技术来构建PPSM，并展示如何将其用于帮助防止凭证篡改攻击。

**Ⅱ.背景**

**密码模型。**人们选择的密码之前已经使用自然语言处理(NLP)工具进行了分析。早期的例子包括使用马尔可夫模型来帮助改进基于字典的破解工具[19],[20]。随后提出了许多使用密码泄漏来学习密码的语言模型。此外，还提出了许多利用密码泄漏来学习密码语言模型的数据驱动方法，其中Weir等使用了概率上下文无关语法(PCFGs)[21]，后来被Kom0anduri等[22]改进，在“通过有限的样本对对手进行建模以评估密码强度”中估计人为选择的密码的分布情况。Ma等[23]使用一些精心选择的参数对基于Markov模型的技术进行了改进，表明在用于生成大量密码时优于基于PCFG的模型。在2016年，Melicher等[24]使用递归神经网络（RNNs）和Hitaj等[25]提出使用深度生成对抗网络（GAN）对密码进行建模。

**密码猜测攻击。**密码模型的一个主要应用是训练暴力猜测攻击。此类攻击分为两大类：离线和在线。当攻击者获得某些用户密码的加密哈希值并尝试通过猜测和检查数十亿（甚至数万亿）的密码来尝试恢复用户密码时，就会发生脱机攻击。攻击者面临的主要挑战是生成密码猜测值，...的有序列表，其真实用户密码很可能会提前出现。该列表中密码的索引称为密码的猜测等级(β)。

当攻击者使用登录界面或其他API针对某个账户提交猜测的密码时，就会发生在线攻击。由于现代身份验证系统会在几次失败尝试后锁定账户（例如10次），因此就攻击者可能做出的猜测而言，在线攻击比离线攻击的限制更大。但是，主要挑战是相同的。给定若干猜测或查询的q，攻击的成功概率就是我们所说的q成功率，记为λq。对于本研究，我们将重点放在在线设置上，将查询预算限制为1,000或更少。

大多数密码猜测文献都集中于非目标攻击，这些攻击以与被攻击账户无关的方式生成密码猜测序列。相反，针对性攻击则尝试利用有关被攻击账户的额外信息，凭据填充攻击将一个账户的泄露密码提交给另一个网站上的关联账户。这些问题引起了越来越多的关注，这在很大程度上是由于大量的密码泄漏：用户账户很可能与从另一个来源泄漏的至少一个账户相关联。

Das等[13]利用此类辅助信息进行了针对性攻击的第一项学术工作。他们表明，大约43％的用户在不同的网站上重复使用相同的密码。他们还手动开发了一种基于规则的算法，以通过用户其他密码之一的信息来猜测用户的密码。我们将这种凭证填充的一般化称为凭证调整，因为攻击者还会对泄露的密码进行修改。后来，Wang等[14]构建了个性化的PCFG模型，以基于个人信息（包括泄露的密码）来指导凭证调整。对于与在线猜测有关的少量查询预算，这些有针对性的攻击要胜过无目标的攻击。但是，这些现有技术不适合更好地利用现在可用的大量泄漏数据。我们将转向更现代的机器学习技术。

**密码强度计。**密码模型还用于开发强度值，它们最常被用作“助推”以帮助引导用户选择更强的密码。最初使用各种统计方法（例如Shannon熵）进行密码强度估计,这种方法有很多缺陷，请参见[28]，[29]。最近可以通过在某种密码模型下计算密码的猜测等级来估算密码强度。给定密码模型可以使用Dell和Filippone引入的蒙特卡洛技术有效地估计猜测等级。

**Ⅲ.预备知识**

用户为不同的账户选择相似和相关的密码，因此可以利用用户的一个密码的信息来更有效地猜测他们的其他密码。虽然可能有许多潜在因素会影响用户对密码的选择，例如其个人统计信息，网站内容的敏感性以及网站的密码策略，但先前的研究表明[14]，用户先前的密码是用户选择密码的最主要因素。因此要了解密码之间的相似性，我们将只关注用户的密码，而不考虑选择密码的用户和选择密码的网站。如果用户经常选择两个密码，我们认为这两个密码是“相似的”。

在形式上，令Σ表示密码中允许的字符集（例如，所有ASCII字符），表示密码的最大允许长度（例如50）。令p表示用户为账户选择密码w∑∗的可能性。我们用W表示该分布的支持度。我们将两个密码和之间的相似性建模为条件概率，即用户选择密码W的情况下用户选择密码W的概率。我们可以扩展相似性的定义，以考虑用户过去的多个密码，，...，并计算用户选择的概率。在这种情况下，我们可以将密码的条件概率分布建模为

先前的研究隐式地尝试使用人工策划的规则或使用概率上下文无关文法（PCFG）来理解人为选择的密码的相似性。近年来，神经网络已被证明对于许多自然语言任务非常有效，例如理解单词的相似性或将自然语言文本从一种语言翻译为另一种语言。我们采用基于神经网络的NLP工具来对密码相似性进行建模。使用这些工具，我们可以建立更有效的攻击和针对目标攻击的有效防御。

**密码相似模型的应用。**如果攻击者可以从其他网站访问用户的密码，则可以使用良好的密码相似度模型对用户进行有针对性的攻击。这种模型对于创建针对最新目标攻击的防御措施也很有用。客户端应用程序可以通过查看与用户其他各种密码之间的相似性来警告/阻止用户选择密码，从而在面对有针对性的攻击时减少危险。密码相似性的另一种应用可以是纠正密码输入错误[7]，因为输入错误通常包含相似的密码。

尽管密码相似性的所有这些应用都需要学习条件概率分布，但是它们需要的是与训练模型不同的接口。例如，要构建有针对性的攻击，必须能够有效地枚举条件概率分布以生成猜测。但是，在使用密码强度计的情况下，我们不需要有效的枚举功能。因此，我们针对两种不同的密码相似性模型。

第一个模型是生成模型，使用之前提出的用于语言翻译的序列到序列样式模型（seq2seq）构建[16]。给定密码，这个模型可以用于按条件概率降序的列举类似的密码。

我们训练的第二个模型基于词嵌入技术，通常被用来理解词之间的相似性[17]。此模型对于获得一对密码之间的相似性评分（代表条件概率）很有用，但是对于仅给定一个输入密码的情况无法枚举相似密码。尽管生成模型也可以用于获得相似性评分，但嵌入模型足以构建强度计。正如我们在第VII节中所示，嵌入模型比生成模型更容易训练，并且更有效地计算密码之间的相似性评分。

**密码泄露数据集。** 我们用于学习密码相似性的数据集是随时间推移泄露的各种密码集合。 该数据集最早是由4iQ在Dark Web[31]中发现的，该数据集包括14亿个邮箱账户密码对，11亿个唯一电子邮件和4.63亿个唯一密码。数据集的管理者（未知）删除了重复的电子邮箱账户密码对。

尽管我们不知道此数据集的确切泄漏情况，但该文件夹包含名为“ imported.log”的文件，该文件指示泄漏主要是在2017年12月5日之前发生的。列出的泄漏来源主要包括Linkedin，Myspace，Badoo，Yahoo，Twitter，Zoosk，Neopet等。尽管没有官方方法可以确认泄漏的真实性，但一些研究人员已验证了一部分密码是合法的。（令人担忧的是，泄露的数据集中出现了两位作者的密码）

|  |  |  |
| --- | --- | --- |
| 属性 | 值 | 百分比 |
| 长度 | 3-5  6-8  9-12  13-50 | 2  48  40  10 |
| 构成 | 仅含小写 | 80 |
| 仅含大写 | 3 |
| 仅含字符 | 38 |
| 仅含数字 | 8 |
| 仅含特殊字符 | <0.1 |
| 仅含字母和数字 | 55 |
| 至少包含一个字母、一个数字和一个特殊字符 | 5 |

图1：清洗后数据中密码长度和组成的分布。

**数据集清洗**。数据集中的几个密码是完整的哈希值。为了清理数据集，我们删除了任何包含20个或更多字符的子字符串(只包含十六进制字符)。这种方法删除了150万个密码。我们还删除了包含非ASCII字符的密码以及长度超过30个字符或少于4个字符的密码。总体而言，我们删除了260万个密码（占0.6％），将有效密码的数量减少到4.604亿。我们还发现4,528个用户与数千个密码相关联。这些不太可能是真实用户的密码，因此我们删除了这些账户。

清洗后的数据集中最受欢迎的密码（123456）被所有用户的0.9％使用。因此，密码分配的最小熵为6.68bit。 q-成功率定义为攻击者对每个账户可以进行q次猜测的预期成功概率，它的上限是q个最可能密码的概率之和。对于我们的数据集= 0.11。这些值与先前工作报告的密码分配一致[14],[32]。图1显示了清理后的数据集中有关密码的组成和长度的统计信息。超过88％的密码长度在6和12之间，而80％的密码仅包含小写字母。

**合并账户。**泄漏数据集包含电子邮件密码对形式的账户凭据，并删除了重复的密码对。我们要合并账户以查找属于单个用户的账户集,这将为我们提供与用户对应的密码列表。我们探索了三种启发式方法来合并账户，如下所述

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| 用户数量（百万）  密码数量（百万） |  | 146  183 | 195  210 | 174  190 |
| 每位用户的密码 | 2  3  ≥4 | 77.0  15.5  7.5 | 57.1  19.1  23.8 | 74.9  16.3  8.8 |
| 密码重用率 |  | 0.0 | 30.3 | 39.7 |
| 编辑距离 | 1  2  3  ≥4 | 9.4  5.2  3.3  82.1 | 6.8  3.9  2.4  86.9 | 9.1  5.9  3.2  81.8 |

图2：三种账户加入技术下的数据集比较.电子邮件地址（），用户名（）和电子邮件地址和用户名的组合（）。 我们仅考虑具有至少两个泄露密码的用户。最后一组行给出了在指定的编辑距离内来自同一用户的不同密码的分数。除前两行外，所有值均为百分比（％）。

* **基于电子邮件（）。**在第一种（也是最明显的）策略中，我们识别用户并根据电子邮件地址加入账户。我们可以说，这种策略将仅合并属于同一用户的账户，因为在大多数情况下，电子邮件地址属于唯一用户。但是，由于从数据集中删除了重复的电子邮件密码，因此我们无法观察到用户使用此方法重复使用密码。对于用户拥有多封电子邮件的情况，此策略无法捕获。
* **基于用户名（）。**因此，我们考虑了另一种方法，特别是使用电子邮件地址的用户名字段-域名和'@'符号（也称为local-part[33]）之前的字符串来进一步加入可能属于同一用户的账户。如果用户名相同，我们将合并两个电子邮件。在此过程中，我们发现用户重用了30％的密码（请参见图2），这略低于先前的报告（40％）[8]。同样，如图2所示，每个用户密码数量的分布以及属于一个用户的密码对之间的编辑距离的分布与基于电子邮件的加入后的情况发生了巨大变化。我们预计，这是由于属于不同用户的账户的错误合并造成的。
* **混合方法（）。**为了减少错误的合并，我们最终考虑了两步方法。我们首先根据电子邮件地址加入账户。然后，如果两封电子邮件的用户名部分相等，则将其视为“已连接”，并且与这些电子邮件关联的两个密码至少具有一个共同的密码。然后，所有连接的电子邮件都被视为属于一个用户。因此，属于用户的两封电子邮件可能没有直接的公用密码，但是它们可能与另一封电子邮件共享公用密码。这种启发式方法导致密码重用率达到40％，同时使编辑距离的分布和每个用户的密码数量与仅基于电子邮件的加入所观察到的非常相似。

另一个可能的更宽松的试探法是在多个账户之间匹配用户名。例如，攻击者可能有理由得出结论，“ Alice.Chang@service1.com”和“ AliceChang@service2.com”是同一个人拥有的账户，我们没有详细探讨这种启发式方法。

合并账户后，我们仅考虑数据集中有至少两个密码泄漏的用户，因为训练和测试我们的目标攻击以及个性化强度计需要用户至少输入两个密码-一个密码用作目标账户密码，另一个作为密码泄露。

由于基于用户名的合并技术不准确，因此我们不再讨论。中密码重用率（40％）和子字符串排列率（18.2％）（参见图2）[13],[14]，这与先前的研究一致。虽然我们没有确凿的事实证明混合方法得出的结果是正确的，但我们相信，鉴于我们在数据集中拥有的信息，这是用户选择的密码分布的最佳近似值。

我们将清理后的基于电子邮件的数据集分为两部分：（80％）和（20％）。类似地，混合数据集分为（80％）和（20％）。除非另有说明，否则对于所有训练和验证（训练期间），我们均使用。这是因为和中用户的相似（唯一）密码的分布几乎相同。由于我们在训练期间仅考虑用户的相似密码，因此我们不会单独使用进行训练。为了测试，我们使用来自和的随机样本。

**Ⅳ.密码相似性的生成模型**

在本节中，我们描述如何构建一个生成模型，该模型估计输入密码的条件概率分布，其中。密码可以看作是字符序列。 因此，我们可以对字符序列的条件分布进行建模，如下所示：

 (1)

这种密码相似度的公式与统计机器翻译(SMT)问题非常接近，或者更一般地说，是学习序列到序列的翻译。Sutskever等提供了一个非常有效的通用框架来训练序列到序列（seq2seq）模型，而无需明确指定序列代表什么。他们的seq2seq模型使用基于编码器-解码器的体系结构。编码器功能将输入序列映射到某个超参数化d维的实值向量上。该向量简洁地“总结”了输入序列的细节。解码器获取向量v并输出相关输出序列空间中tokens的条件概率分布。

学习密码相似性的一个建议是将seq2seq方法直接应用于字符序列的密码。我们将此模型称为“密码转密码”或“ pass2pass”，但是与以前的工作相比，该技术并未提高性能。在附录A中，我们提供了有关如何训练此模型的详细信息。下面我们描述使用基于编码器-解码器的体系结构对密码相似性进行建模的另一种（更有效的）方法。

**密码到路径模型。**在pass2pass中，我们尝试学习完整密码的条件概率。由于效果不佳，我们决定学习用户可能对其先前密码进行的修改。网站的密码策略可能会影响其中某些修改的选择。但是，我们的相似性模型可以轻松扩展为考虑网站密码策略，为简单起见，我们暂时将忽略密码策略的影响，并考虑所有的密码都是相似的。

我们将对密码的修改视为如下定义的一系列转换。单位转换在指定密码中要应用的编辑。因此，用三元组（）表示，其中e代表要应用的编辑，是字符或空字符串，而是密码中的编辑位置。我们考虑三种类型的编辑：替换（sub），插入（ins）和删除（del）。对于插入和替换编辑，表示要插入或替换的字符；如果表示删除，始终为空字符串⊥。例如，在字符串'password1'上应用转换（sub'！',8）意味着用密码'！'替换密码中的第8个（最后一个）字符，那么字符串将变为'password！'。

给定一对具有编辑距离t的密码（，），我们可以找到一系列变换，当以累积方式应用于密码时，它将产生密码。这样的变换就是我们所说的路径。为了计算两个密码之间的路径，我们选择一个最短的路径，在该密码关系中，更倾向使用删除而不是插入和替换来打破联系。路径中的变换按编辑位置排序。（有关更多详细信息，请参见附录C。）例如，从"cats"到"kates"（编辑距离为2）的路径为：{(sub，'k'，0)，(ins，'e'，3)}。

在pass2path中，我们在给定另一个密码的情况下定义密码的条件概率，如下所示。



其中t是两个密码和之间的最小编辑距离，并且。

我们使用基于编码器-解码器的模型，其中解码器函数的输出是T变换中的概率分布。 这样，我们可以将上面的等式重写为



其中是编码器的输出，是解码器在输入和上的输出，并且是特殊的路径开始符号。包含来自的信息，因此替换了最终方程中的。 我们将学习此概率模型的任务设置为有监督的学习任务，其训练目标是找到使各个用户选择的密码提示之间正确编辑路径的对数概率最大化的参数。设D为此类密码广播的集合，则训练目标为



pass2path的模型架构类似于用于seq2seq的基于编码器和解码器的架构，该架构使用两个递归神经网络（RNN）实例化[16]。编码器和解码器的RNN一起训练，附录B中给出了模型体系结构的详细信息。下面，我们将介绍pass2path的一些进一步的训练详细信息。训练后,给出的模型可用于生成类似的密码泄露中的密码。关于如何生成q个最可能的密码的详细信息将在第五节中给出。

**训练pass2path。**我们使用基于电子邮件数据集（）创建的数据来训练密码模型。对于数据集中的每个用户，我们计算所有密码对包括密码对的重新排序，从而产生8.23亿个密码对。

我们将密码表示为美式键盘上的按键序列。例如，“ PASS-WORD！”表示为“<c>password<s>1”，其中<c>和<s>表示键盘上的大写锁定和Shift键。Chatterjee等[7]表明密码的键序表示法可以有效地改善密码错字校正，我们在这里使用它是因为它比标准编辑距离更好地捕获了与大写有关的变换。

对于训练集中的每对密码，我们使用基于动态编程的算法生成了它们之间的最小路径。该算法是精算算法的扩展，用于计算字符串之间的最小编辑距离[34]。给定一对密码和，我们首先将密码转换为密码序列，然后找到可以将转换为的转换路径。我们在附录D中详细描述了我们的算法。

少量密码对的手动样本显示，其中很大一部分是完全不同的，没有任何明显的语义或句法相似性。因此，我们决定在训练之前根据路径长度（也等于密码之间的键序编辑距离）过滤密码。给定一个截止值δ，我们只考虑路径长度最大为δ的密码对。我们从δ= 2开始，完成三个时期的训练，然后通过添加更多对δ= 3、δ= 4的对来逐步转移学习网络。我们发现这种方式收敛速度更快，并且具有更高的准确性。总体而言，该模型是针对1.44亿个密码对进行训练的，有关训练pass2path的更多详细信息，请参见附录C。

pass2path模型有240万个参数，占用磁盘上60兆字节的存储空间。在Nvidia GTX 1080 GPU和Intel Core i9处理器上训练批量大小为256的模型大约花了两天时间。然而，这项训练只需要不到2 GB的物理内存。

**Ⅴ.使用path2path进行定向攻击**

如第二节所述，学习密码相似度的一个主要应用是了解目标猜测攻击的危险，在此情况下,攻击者根据用户的其他密码生成猜测密码。在本节中，我们将描述如何从我们训练有素的pass2path模型中获得数千个猜测，以建立有效的针对性攻击。我们将通过仿真证明我们的攻击优于所有先前的猜测攻击。

**生成相似密码。**为了将密码相似性模型用于有针对性的攻击，我们需要能够在给定泄露密码的情况下，以似然度递减的顺序生成一系列密码。即。 这里的q是一些猜测，我们将在下面对一个参数进行具体化。给定我们的pass2path模型下生成非常简单。首先，将输入密码转换为固定的维向量，并将其与特殊的序列开始符号一起馈送到解码器。解码器在一组变换T上输出概率分布。在每次迭代中选择最可能的输出，并将其用作下一次调用解码器的输入，直到到达序列结束符号。然后将转换的输出序列应用于输入密码以生成新密码。

但是此过程仅输出最可能的密码。为了产生多个输出，我们使用了广度优先光束搜索技术。波束搜索算法使用一组大小为q的波束，该波束在每次解码迭代时都存储到目前为止生成的q个最可能的路径（以及网络状态和概率）。如果路径以结束序列转换结束，我们将其称为完整路径，否则将其称为不完整路径。利用解码器在向量的输入和序列开始符号上输出的q个最可能的变换来初始化波束。接下来，对于当前存储在波束中的每个不完整路径，在路径的最后一个变换上调用解码器，并通过将变换附加到路径来计算新路径。光束中仅保留q个最可能的新建路径，以进行下一次迭代。重复此步骤，直到达到预定义的最大迭代计数或完成光束中的所有路径为止。

光束搜索是一种贪婪算法，不能保证提供q个最可能的路径。但是，它是一种广泛使用的启发式方法，用于在有限的内存和时间的情况下找到前q个猜测。要查找输入密码的q条路径，波束搜索将最多对解码过程进行q·t次调用，其中t是表示模型中允许的输出路径的最大长度的参数。

由于可能存在多个路径，当应用于密码时会输出相同的密码w，因此波束宽度为q的波束搜索可能不会生成q个唯一的密码。 因此，我们生成q'≥q个密码，然后输出前q个唯一密码。 在我们的实验中，我们发现取q’= 2·q足以为超过99.9％的密码找到q个唯一密码。

**评估有针对性的猜测攻击。**我们基于有多少用户的账户会因为另一个泄露的密码而被攻破的情况对有针对性的攻击进行了评估。我们主要关注在线攻击设置，在这种情况下如果尝试登录失败的次数过多，则应阻止账户登录（即如果没有带外身份验证，则将不允许登录）。我们称查询预算q为尝试次数，也是攻击者阻止账户之前可供攻击者使用的最大猜测数。为了评估攻击，我们将使用q{10,100,1000}的猜测预算。这些是身份验证服务使用的典型值。

我们使用两个测试数据集：（仅使用电子邮件来识别用户），以及（使用混合方法生成）（请参见第III节）。第一个模拟攻击是已部署凭据填充对策的服务，例如，如果用户先前的密码存在泄漏，则强制用户选择新密码。由于不允许在两个账户之间重复使用密码，因此以下将其称为“无重复”设置。第二个模拟攻击没有部署这种对策的服务，因此我们将其称为“重复”设置。

这两个数据集都包含数百万个用户。我们评估的某些目标攻击在计算上非常昂贵。我们需要选择一个较小但有代表性的测试数据样本。我们计算了用户针对不同测试集大小使用相同和相似密码的比率的差异。对于大小≥105的测试集，我们发现方差足够低（<0.5％）。因此，我们为每个数据集随机抽取105个随机用户进行评估。对于每个选定的用户，我们从与该用户关联的多组密码中随机选择两个密码而不进行替换-其中一个（随机选择）被视为泄露密码，另一个被视为目标密码w。

我们将攻击算法与现有的两种有针对性的猜测攻击进行了比较。Das等[13]创建了一个手动管理的转换列表以生成类似的密码。 Wang等[14]根据有关用户的信息提供了多种攻击，包括他们的人口统计信息，其他密码以及它们的组合。我们将集中讨论其中的TarGuess-II攻击[14]，该攻击仅在知道先前密码的情况下进行。Wang等慷慨地提供了Das等的实现、算法及其TarGuess-II算法。后者需要从数据集中进行训练。有关详细信息，请参见附录E。

我们还针对仅基于训练集中密码的经验分布的两种攻击进行了比较。第一个是无目标攻击，它简单地（对于任何泄露的密码）猜测训练数据集中用户的q个最常用的密码。我们发现，在少量猜测（例如q≤104）的情况下，该非目标经验模型的性能优于最新的非目标猜测攻击[24]。

第二种是有针对性的经验攻击，对于给定的泄露密码w，攻击者输出同时使用密码w的用户最常用的q个密码。尽管这种有针对性的经验攻击从概念上讲是简单明了的，但它需要大量的有效访问内存才能实现。因此，我们通过计算在测试数据中泄露的密码的经验分布来模拟这种攻击的有效性。

我们将泄漏的密码用作所有设置中所有定向攻击的第一个猜测。无目标的经验攻击会在所有设置中为所有账户使用固定的q个猜测列表。

对于更高的查询预算值q，某些攻击无法对某些泄露的密码产生q猜测。在这些情况下，我们只是中止了攻击而没有用完剩余的查询预算。在实践中，人们可能会尝试以某种临时方式扩展猜测的数量，例如，通过添加非目标猜测。放眼未来，这样的修饰无法充分改善攻击以赶上pass2path的速度。

在图3中，我们展示了两种设置下不同攻击的功效。我们依次讨论每个设置的结果。

**无重复设置时的攻击效率。**首先，我们讨论结果（左表），其中目标密码与泄露的密码不同。值得注意的是，对于查询预算q = 10，基于经验分布的目标攻击的性能要优于所有先前的目标攻击。但是，它缺乏可概括性，因此无法在较高的查询预算下发挥作用。平均而言，每个密码在训练数据集中只有几个关联的密码，这种攻击只能猜测在训练数据集中观察到的密码。

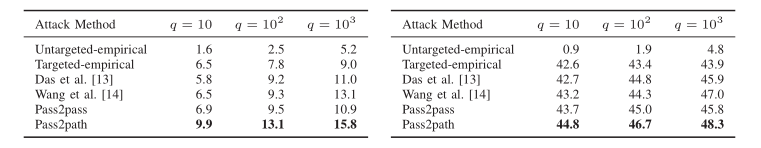


图3：在由无重复设置（左）和有重复设置（右）生成的两个测试集中，q猜测中各种攻击所猜测的密码百分比。 在后一种情况下，猜测性能的显着提高是因为40％的目标密码与泄露的密码相同。

Das等攻击不需要训练数据，是我们测试的所有目标攻击中执行速度最快的。它的表现相对较好，在不到一千次的猜测中破解了11％的用户账户。但是，对于许多密码而言，这种有针对性的攻击无法产生1,000次猜测（因为它用尽了处理规则以应用于泄漏的密码）。Wang等[14]算法是在我们工作之前最先进的针对性猜测攻击。它可以在不到1,000个目标猜测中破解13％的用户账户。但是，猜测的生成过程非常缓慢，需要花费三天多的时间才能在具有Core i9 CPU和128 GB内存的机器的一个线程上仅对我们的一个测试集中的所有密码进行猜测。尽管在在线猜测攻击中，计算复杂度并不是特别重要（与试图破解哈希的离线猜测攻击不同），我们之所以提到它是因为它在我们的仿真中证明是一项重大的工程挑战。

最后，pass2path在所有攻击中表现最佳，在100个猜测中破解了约13％的密码（比Wang等破解的密码高40％），在1,000个猜测中破解了15.8％的密码（增长了20％）。 pass2path算法的计算速度也相对较慢，需要四个小时的计算才能评估测试集。这仍然比Wang等的速度要快得多。

**重复设置下的攻击效率。**现在我们讨论来自的测试集的结果。在结果中用户重用了大约40％的密码，这使其很容易成为凭据填充的目标。因此，在这种情况下，对于每次攻击，我们将泄漏的密码用作第一个猜测。根据攻击技术得出剩余的q-1个猜测。

无目标的经验性攻击表现不佳，但是不足为奇，因为它没有利用泄露的密码。在这种情况下，其他攻击的基准功效非常高，因为仅凭凭证填充即可破解40％的账户。我们的攻击pass2path再次优于所有以前的算法，不过由于较高的基线功效，在此情况下，改进程度较小。例如，相对于最佳的先发攻击（Wang等），在1,000个猜测中的改善仅几个百分点。就是说，绝对而言，pass2path使用1,000个猜测来破坏出现在泄漏中的近一半用户账户。

如果没有凭证填充防御，如果用户之前的某个密码被泄露，那么与非定向攻击相比，该用户的账户在1000次猜测中受到攻击的可能性将增加10倍。

**使用多个泄露的密码进行攻击。**到目前为止，我们解释的针对性攻击假定仅访问一个泄露的密码。但是在某些情况下，攻击者可以获得目标账户的多个泄露的密码。从理论上讲，可以训练一种类似于pass2path的模型，但是它会使用一系列密码作为输入而不是仅输入一个。

但是，我们决定采用一种更简单的临时方法。我们为每个输入密码独立生成猜测的排序列表，然后以循环方式从每个列表中选择一个来合并列表，直到猜测预算用尽。为了测试这种攻击策略，我们从不重复的数据集中随机选择了105个用户，这些用户至少泄漏了三个或更多密码。对于每个用户，我们随机选择一个密码作为目标，其余密码作为泄漏的密码。我们的启发式攻击方法可能会在103次猜测中危及23％的用户账户-与仅使用一个泄露的密码相比，提高了47％。未来的工作可能会探索更高级的模型，这些模型将更仔细地利用多个泄露的密码。

**攻击所有账户。**在此实验中，考虑到攻击者知道他们的密码之一，我们考虑破解用户的所有账户。在这种情况下，攻击者将为每个账户获得q个查询。为了测试攻击的有效性，我们从的105个用户中抽取了两个以上的泄露密码，并选择其中一个作为泄露密码，其余作为目标密码。对于每个账户，我们使用pass2path定向攻击为泄漏的密码生成103个猜测，并检查猜测列表中是否有目标密码。我们发现，在被攻击者丢失密码的用户中，有18%的用户至少有另外一个容易受到针对性攻击的账户，尽管这些账户使用的密码与被泄露的密码不同。

**目标和非目标攻击之间的交叉。**与非目标猜测攻击相比，针对少量猜测（q≤103）的目标攻击非常有效。但是我们观察到，随着q的增加，针对目标的定制攻击的价值将降低。对于不同的猜测次数，我们在图4中绘制了pass2path（有针对性）和非目标经验攻击的效果。为了生成该图，我们从抽取了105个随机用户，并为每个用户随机抽取了两个密码。因此，我们将定向攻击的优势与性能最好的非定向攻击进行比较，而忽略了凭据填充的优势。

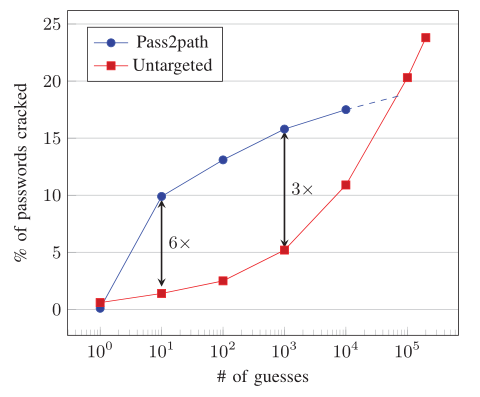


图4：针对较大的猜测预算，针对性攻击和非针对性攻击的相对优势，以及交叉点，即非针对性攻击比针对性攻击更有效。 由于计算限制，我们没有使用pass2path计算超过104个猜测的点，因此虚线反映了观察到的趋势。

可以看出，在q = 10的猜测预算中，针对pass2path的攻击可能比未针对性的攻击（占测试账户的1.6％）所攻击的账户（占测试账户的10％）多六倍。图3左表的第一列也显示了这一点。但是，这种相对优势随着查询预算的增加而降低，并且如果攻击者可以做出很多假设（例如q≥105），则无目标攻击会变得更加有利。在脱机攻​​击环境中，攻击者窃取了Web服务的密码哈希数据库，并试图通过进行数十亿次猜测来破解密码哈希，因此针对性攻击的使用将受到限制。

**讨论。**与先前的工作一致，我们使用了模拟来评估定向猜测攻击的功效。在实践中，攻击者会遇到一些其他复杂问题，例如有关密码组成的网站特定规则。pass2path模型的一个很大的优点是，它可以调整为轻松生成与网站密码策略匹配的密码。如我们在第六节中所示，只能在满足策略的数据集的子集上对使用传递学习的pass2path模型进行重新训练。成功的密码猜测可能不足以单独访问采用双因素身份验证机制的现代服务。近年来，双因素身份验证的使用有所增加，但仍不广泛。一些双因素身份验证系统具有漏洞[36]，[37]，可以与我们的密码猜测攻击一起利用。最后，对于测试模拟，我们使用各种启发式方法加入了账户，但是无法确定正确匹配的用户名数量。测试数据集还包含泄漏中存在的密码，因此通常可能偏向于弱密码。我们想要验证对实际账户的攻击效率，这促使我们执行下一部分第六节中讨论的真实破解实验。

**Ⅵ. 在实践中有针对性的攻击效能**

在上一部分中，通过将各种目标攻击的性能与保留测试数据集进行了比较，对各种目标攻击进行了评估。在这里，我们转向评估针对真实账户的针对性攻击的有效性，从而准确模拟攻击在真实情况下的进行方式。为此，我们与康奈尔大学（ITSO）的IT安全办公室合作。我们测试了Cornell用户账户中有多少部分容易受到在线猜测攻击。尽管我们已经对真实用户账户进行了非目标攻击（例如[5]中的分析），但据我们所知，这是对真实用户账户的目标攻击的首次评估。

在破解后汇总的数据中，我们发现了具有有效康奈尔账户的19868封电子邮件。在ITSO自2009年以来维护的密码更改日志中，我们确认至少有15776个账户肯定具有用户选择的密码。除非另有说明，以下所有实验结果都是针对这15776个账户的。我们针对这些账户进行了三种在线猜测攻击实验：无目标经验，Wang等人的方法和pass2path。

Cornell使用L8C3密码策略，也就是说，密码必须至少具有来自至少三个不同字符类的8个字符：大写字母，小写字母，数字和符号。我们使用转移学习对训练数据重新训练pass2path，训练数据的目标密码符合康奈尔大学的密码组成要求。我们还采用了非针对性的经验攻击，考虑最流行的密码满足康奈尔密码组合的要求。然而，没有简单的方法来调整Wang等人攻击算法生成的猜测。有关实验设置的更多详细信息，请参见附录F。

**结果。**实验结果总结在图5中。非针对性的经验攻击表现很差：它只能破解0.1％的目标账户。王算法攻击效果要好一些，最多可以破解这些账户的2.6％，但是如上所述，其性能受到康奈尔（Cornell）密码要求的自定义难度的负面影响。

Pass2path表现最好，在不到1,000个猜测中破解了8.4％的账户。其中，只有22个（0.1％）账户使用与泄漏的相同的密码进行了破解。这是因为ITSO使用第三方服务来帮助防止凭据填充攻击。

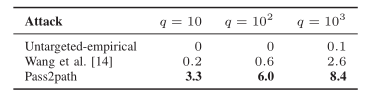


图5：在突破数据集中发现的15776个康奈尔活跃账户中，在三种攻击方法的猜测次数范围内被破坏的比例。

回想一下，我们使用来自漏洞的保留数据进行的模拟表明成功率为16％。尽管尚不清楚是什么原因解释了这一差距，但我们认为这是由于与这些违规行为相比，康奈尔大学密码分配的差异。换句话说，针对性攻击对于这些公共数据泄露略有过分适应，并且在评估实际系统中的漏洞时，发生率会有所不同。

但尽管如此，这个实验表明了账户在目标攻击下的脆弱性，其中1374个活动账户易受至少一种远程猜测攻击的攻击。我们已通知ITSO有关这些易受攻击的账户，我们正在与ITSO合作以保护它们。

**Ⅶ.防御有针对性攻击**

前一节强调了即使使用最先进的凭据填充对策，也存在定向猜测攻击的危险。我们可以保护账户免受这些攻击吗？一种方法是让站点操作员像对ITSO一样模拟有针对性的攻击，并在站点未正确登录尝试的阈值范围内为易受攻击的人员重置密码。但是即使这样做，用户也可能会选择自己易受攻击的密码变体。因此，我们还想提供一种在面对标准和有针对性的猜测攻击时衡量密码强度的方法。

**密码强度计。**密码强度计（PSMs）向用户提供有关其密码强度的实时反馈。从过去情况看，密码强度是使用Shannon熵或启发式变量来衡量的[27]，但这些度量方法却完全不准确[28]，[38]。最先进的强度计[24] [26]通过在最著名的猜测攻击下估算其猜测等级（β）来推断密码的强度。密码的猜测等级是攻击在到达密码之前进行的猜测次数。但是，到目前为止，考虑的猜测攻击是用户不可知的，因此相对于目标猜测攻击而言是不准确的，正如我们现在解释的那样。

考虑以下示例情况。用户将在某个网站上以密码“atbaub183417a”注册用户名“alice@gmail.com”。 zxcvbn强度计[26]当前在Internet上使用，被认为是同类产品中最好的PSM，它提示密码的猜测等级为1012，这意味着它是一个非常强大的选择。但是，如果alice@gmail.com存在一个密码为“atbaub183417123”的泄漏，我们的目标攻击在不到五次的猜测中就猜对了。稍后，我们将更广泛地量化现有PSM频繁高估pass2path容易破解的密码强度的频率。

**个性化密码强度计。**为了解决上述差距，我们提出了个性化密码强度计（PPSM）。 PPSM可以用于在密码选择过程中向用户提供有关其密码的反馈，这可以作为一种推荐，也可以作为严格要求密码具有必要强度的要求。PPSM将目标（潜在）密码w和用户的关联密码集合P作为输入，并在包括目标攻击在内的最著名攻击下返回猜测等级。将来，我们可能会扩展PPSM，以考虑到其他特定于用户和上下文的信息，例如用户名和站点域名。

一种估计密码w的猜测等级的方法是在攻击下返回猜测等级，例如使用pass2path的攻击。但是，使用基于神经网络的模型来生成猜测，这样不仅计算量大，而且占用大量带宽（如果需要通过网络发送给客户端）。 Melicher 等[24]使用各种灵活的优化来减少RNN模型，使其更有效。我们有可能使这些技术适应基于RNN的编码器-解码器体系结构。相反，我们将探索一种根本不同的方法，它将更加高效。

传统上，密码强度计会提供一个易于计算和易于理解的分数（大约反映猜测等级）。例如，zxcvbn [26]在{0，...，5}中给出一个分数，而nn-pwmeter [39]在[0,100]之间给出一个分数。同时我们观察到，对于大多数用途，PPSM仅需要输出强度得分，而不必输出猜测等级。因此，我们的PPSM的基础将是一个二进制分类器C，如果使用目标猜测攻击很可能在给定另一个密码的情况下很容易地猜出目标密码w，则将两个密码w和作为输入，并输出0，否则输出1。之所以要建立二进制分类器，是因为易受目标攻击的密码是可以在很少的猜测中猜到的密码。我们使用1,000作为“很少”的度量标准，但是我们的框架可以很容易地与其他值一起使用。

为了构建这样的分类器，我们将使用基于词嵌入技术的密码相似性度量。这一切的好处是，我们无需使用（通常更昂贵的）生成密码模型，而使用基于嵌入的相似性模型，该模型可以快速输出两个密码之间的相似性得分，但不能从一个泄漏的密码中提供枚举相似密码的有效方法。

展望未来，我们将向您展示如何结合多种强度估算，特别是传统的非目标强度计和我们的相似性得分，构建我们的PPSM。这将产生一个强度计，它可以在有针对性和无针对性的攻击下准确地测量目标密码w的强度。

在本节的其余部分，我们将讨论如何使用基于单词嵌入技术的密码相似性度量来构建分类器C。

**嵌入密码相似度。**单词之间的相似性已在NLP中探索了数十年。最近，基于神经网络的词嵌入技术已被证明非常有效[17]，[18]，[40]。根据单词嵌入模型，我们将密码嵌入定义为将密码映射到中的d维向量的函数。维数d是一个参数，通常选择相对较小，例如100或200。对嵌入进行训练，以使相似密码的矢量具有较小的距离（对于距离的某种度量）。相似性将取决于上下文。对于我们的个性化强度计，如果两个密码经常由同一用户选择，则应将其视为相似。嵌入提供一种方法来定义评分函数s：W×W→[-1,1]，用于测量两个密码的相似性：应用嵌入，然后计算所得向量之间的距离。

我们使用[18]中描述的FastText模型构建密码嵌入。FastText模型通过将大量文本语料拆分为一组上下文（单词的短序列）来学习相似性。一起出现在上下文中的单词被认为是相似的。通过将同一用户选择的密码视为上下文中的密码，我们将其应用于密码。 FastText考虑n-gram的单词，因此可以产生一个嵌入来处理训练集之外的单词。这对于我们的应用很重要。

就我们的目的而言，密码表示为的n-gram的并集。令表示密码w的n元语法集。每个单词的开头和结尾都通过添加两个特殊符号“<”和“>”来明确表示。（否则不在Σ中）。例如，密码w = qwerty，= 4， = 5，将具有以下n-gram。

= {<qwerty>，<qwe，qwer，wert，erty，rty>，<qwer，qwert，werty，erty>}。 （请注意，完整密码始终包含在中。）然后，分数功能为：

 (2)

在这里,是元素gV的向量嵌入，而V是在训练数据中看到的w的所有的并集。 我们将密码wW表示为的嵌入，如果wV，则将密码wW嵌入，那么=，否则。如果密码w或它的任何n-gram不存在于V中，则w的嵌入被设置为中的一个随机向量。

**训练密码嵌入。**为了训练我们的密码嵌入FastText模型，我们使用了带有负采样的skip-gram方法。我们将每个密码表示为一系列按键，就像在第四节中训练pass2path一样。该模型需要选择各种超参数。

我们将向量的维数设置为d=100。与通常建议的d=300相比，这将导致训练更快，并且使用嵌入生成的分类器的性能更好（请参见下文）。 我们把子采样设为10-3。子采样通过随机忽略某些频繁使用的密码来平滑频繁使用和很少使用的密码之间的更新频率。我们还只考虑在训练数据集中出现至少10次或以上的密码。最后，我们将n-gram的最小大小设为m min = 1，以确保我们可以为训练中看不到的所有密码构建一个嵌入。我们将设置为4。

**使用相似性评分对密码进行分类。**我们想要使用密码相似性函数s来构建一个二进制分类器C，该分类器C使用一对密码并输出一个二进制分数。为此，我们确定阈值：对于相似度分数大于的所有密码对，我们给它们分配分数0，否则将其分配为1。我们将阈值表示为的分类器。如果目标密码w可以由到目前为止已知的三种目标攻击（Das等人，Wang等人，和pass2path）给定在103个猜测之内猜出，则我们将密码对称为脆弱对。我们想要选择a，以便它正确地识别易受攻击的密码对（通过在它们上输出0），同时最大化其输出1的密码对的数量。后一个竞争目标是归因于密码注册期间分类器的可用性，即过分地将密码对标记为易受攻击，而实际上却并非如此，这会受到阻碍。

相对于一组密码对，的召回率是相似性低于阈值的易受攻击的密码对的一部分。精度是相似性实际上高于阈值的密码对的分数。

我们通过以下方式计算阈值。我们从中随机选择105个用户。对于每个用户，我们从与之关联的密码集中随机选择两个密码而无需替换。密码之一（任意选择）被视为目标，另一个密码被视为泄漏。对于每对（，），我们将它们标记为易受攻击或未使用如上所述的三种有针对性的猜测攻击。 这构成了我们的基本准则。 现在我们计算每对之间的相似度分数s（，）。对于一系列阈值α∈[0,1]，我们计算的精度和召回率， 结果精度和召回率如图6所示。从图中可以看出，在精度和召回率之间存在一个权衡。为了确保召回99％（能够检测到99％的易受攻击的密码对），我们选择a = 0.5的阈值。 的精度为60％。

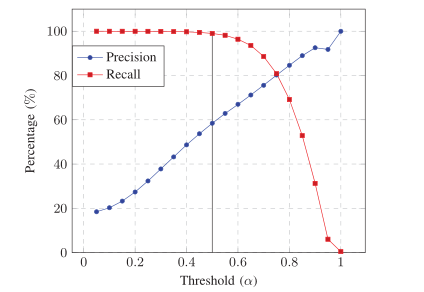


图6：在根据的105个密码对的随机样本计算出的阈值的不同值下的PPSM分类器的精度和召回率。

**压缩嵌入模型。**我们的密码嵌入模型的基础是一个查找表，其中的键是常见密码及其n-gram的列表，而值是d维实值向量。因此，需要空间来存储嵌入内容。对于我们性能最佳的型号，该容量超过1.5GB。在这里，我们探索两种减少模型大小的技术，同时能在识别用于目标猜测的弱密码时保持良好的准确性。

首先，我们观察到，即使删除了所有针对w∈V的密码嵌入值=，模型的质量也几乎保持不变。相反，可以通过来估计这些值。从模型中删除单词词汇将大小从1.5 GB减小到仅195 MB，而强度估计的准确性没有任何明显的变化。

接下来，我们使用乘积量化（PQ）技术[41]进一步压缩向量，这种技术已经被证明对压缩神经网络模型有效[24]。 PQ使用参数η来确定压缩率­——η的值越小，模型大小越小，但是压缩后输入向量的重构精度越差，n元语法向量的重构误差进而影响得分函数和分类器。

我们为η的不同值构造分类器，并根据从的许多随机用户中选择的105个密码对的样本来计算其精度和调用率。 结果记录在图7中。我们可以看到，即使将模型压缩到3 MB（η= 5）之后，对召回的影响也很小。精度从59％降至41％，我们认为可以接受。

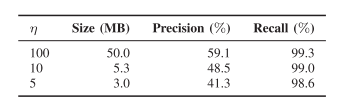


图7：对于不同的η值，使用乘积量化（PQ）压缩基础密码嵌入模型时，对分类器的精度和召回率的影响。

**Ⅷ.PPSM评估**

我们构建了称为vec-ppsm的PPSM，其中包含两个组件–一个负责估算针对目标攻击的强度，另一个负责估算针对非目标攻击的强度。对于前者，我们使用第七节中的分类器，而对于后者，由于其准确性和性能，我们将使用zxcvbn。在给定一组（泄露的）密码P的情况下，Vec-ppsm估计密码w的强度在0（最不安全）至4（安全）的范围内。

回想可以在仅给一个密码的情况下进行分类。要在vec-ppsm中使用它，当给定密码集P中可以有多个密码时，我们可以使用zxcvbn也会使用的最小强度方法。也就是说，给定每个∈P我们计算w的强度得分，并输出最小值。如果P为空，则输出1。

此后，按顺序为了估计针对非目标攻击的强度，vec-ppsm与常规的非目标强度计（例如zxcvbn）结合使用：如果w的目标强度得分在给定P时为0，则vec-ppsm输出0，否则输出分数输出zxcvbn。

**其他比较方法。**我们将vec-ppsm与两种最先进的强度计：zxcvbn [26]和nn-pwmeter进行了比较，后者是在[24]，[39]中提出的基于神经网络的强度计。这些强度计的默认行为与用户的其他密码无关。但是，zxcvbn接受一个可选参数来添加特定于站点的密码黑名单。我们使用此选项来模拟目标强度计版本的zxcvbn，我们将其称为tar-zx。它应用zxcvbn，将可选参数设置为（泄漏的）密码P的集合。未经修改的zxcvbn的原始用法在下文中称为untar-zx。

untar-zx和tar-zx都给出了可以在不到一千次猜测中被猜测的密码的分数0。 nnpwmeter返回一个百分比值，其中0表示弱密码，而100表示​​非常强。因此，我们将其分为5个部分，并为得分小于20的密码分配了0分。

理论上，以前的目标攻击[13]，[14]可用于构建个性化强度计，但Das等人将其作为参考。作为目标攻击，效果不佳（见图3），Wang等人的猜测生成过程太慢，无法实时生成许多猜测（例如103）。因此，两者都不立即适合用于构造强度计。

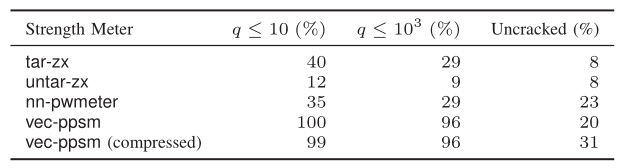


图8：比较被考虑的强度表分配为零强度（使用的“不安全”）的易受攻击的密码的百分比。我们使用untar-zx作为vec-ppsm中的非目标强度估算组件。vec-ppsm的最后一行使用压缩的嵌入模型。最右边的一栏给出了未针对任何目标攻击在103个猜测中破解的密码的分数。

**评估vec-ppsm。**我们从测试数据集DE用户中随机抽取了105个样本，对于每个用户，我们随机抽取了两个密码，而不替换为目标密码和用户的其他密码。然后，我们尝试使用来自第五节的pass2path目标攻击来破解目标密码。我们还计算所有考虑中的强度表下的目标密码强度。

在图8中显示了易受攻击的密码的百分比——­可以通过pass2path猜到的密码少于10和1000的猜测——被各种强度计分配了强度0（不安全使用）。毫不奇怪，所有现有的强度计的性能都很差：它们为70-90％的易受攻击密码分配的分数为1或更高（这意味着密码可以安全地抵抗在线猜测攻击）。但是这些密码在我们的有针对性的攻击中猜测不到1000次，因此使用起来很危险。

当我们仅关注q = 10次尝试中可以猜测的密码时，情况可能会更加令人担忧。先前的强度计认为其中60％是安全的。三种强度计中性能最好的强度计是tar-zx，它通过对输入密码应用一组规则来构造黑名单，从目标密码中删除所有出现在黑名单中的字符串，然后计算其强度。即使这样，tar-zx也只能在不到10个猜测的情况下检测到40％极易受到目标攻击的密码。

最后，veci-ppsm可以检测出所有密码中96%的密码，这些密码可以在1000次猜测中被猜中。除了误报率增加外，vecp - sm的压缩版本执行情况类似。（请参见图8的最后一列。）

我们还调查了vec-ppsm是否标记了发现的康奈尔账户密码容易受到第六节所述的三种在线猜测攻击之一的攻击。考虑到相关的泄露密码，我们发现vec-ppsm会将得分0（标志不安全）分配给99.1％的脆弱密码。其余的0.9％实际上是易受非目标性经验攻击而不是目标性攻击的密码。从理论上讲，vec-ppsm底层的无目标攻击强度计应已将这些密码标记为弱密码，但未考虑康奈尔密码策略。这可以通过修改无目标攻击强度计来解决。

**部署vec-ppsm。** PPSM将在几种不同的部署方案中帮助提高安全性，我们现在将讨论这些方案。

立即部署PPSM的最简单的地方可能是在密码更改工作流程中，在该流程中，用户提供了旧密码和新密码。用户的旧密码可以用作“泄露的”密码，因此PPSM可以确定新密码是否足够强，即使先前的密码已泄漏。加上导致用户更改密码的违规通知，这可以完全防止凭据调整攻击。PPSM可以作为JavaScript有效负载发送，并在客户端执行强度检查，从而确保不需要将候选密码发送到远程服务器。

我们注意到，在这种情况下，嵌入模型会发送到客户的计算机上，我们必须考虑这可能带来什么风险。例如，攻击者可能试图发现用于训练模型的泄漏数据集中存在的一组密码。但是我们的压缩嵌入模型不包含有关单个密码的任何信息，也不包含在培训数据中与之关联的账户。相反，它包含大小为1到4的n-gram。它也不包含有关它们在训练数据中的流行程度的任何信息。它向攻击者揭示了一些有关密码相似性的信息，但与其他强度计（例如nn-pwmeter）不同，它没有提供足以针对目标猜测攻击的生成模型。

第二种部署方案可以在登录期间使用vec-ppsm。我们假设该服务可以访问泄露的密码数据（可能通过第三方服务）。每次用户成功登录，给定与该账户关联的密码泄露，该服务都会根据vec-ppsm检查输入的密码是否不安全。如果是这样，则会采取必要的步骤来警告用户或以其他方式保护账户。这些都可以在服务器端完成。

使用身份验证服务进行初始密码注册时，可能会使用PPSM。但是PPSM需要访问用户的其他（泄露的）密码，才能准确地估计所选密码的强度。如果无法访问用户的其他密码（无论是否泄露），则vec-ppsm将默认为非目标强度估算值。在典型的Web注册中，我们希望将PPSM作为JavaScript有效负载发送到客户端，但是随后它也需要将泄漏的密码发送到客户端，这存在安全风险。相反，可以在服务器端执行PPSM检查，但这需要向服务器显示候选密码。

最后，可以与密码管理器一起在客户端设备上使用vec-ppsm。密码管理器可以代表客户端使用第三方泄漏检查服务（例如[9]，[42]）来检查客户端的任何密码是否泄漏。然后，在给定那些泄露的密码（或所有其他密码）后，可以使用vec-ppsm评估用户其他密码的强度，类似于它们已经提供了针对性攻击强度的反馈[43]。当然，现代的密码管理器提供了选择随机密码的选项，这种情况避免了使用vec-ppsm（或任何强度计）。但是，许多用户仍然使用他们自己选择的密码，并将其存储在密码管理器中，在这里vec-ppsm将提供好处。

我们已经证明vec-ppsm可以警告用户选择易受攻击的相似密码。但是，我们尚未解决有关如何提供建设性反馈并帮助指导他们创建强密码的用户界面问题。例如，如果密码太类似于泄露的密码而被拒绝，用户可能会感到困惑。如何最好地告知他们这仍然是一个悬而未决的问题。

**概念验证的实现。**我们在Python 3.6中实现了vecppsm。为了压缩嵌入模型，我们使用了Facebook的Faiss库[44]提供的产品量化功能。我们通过随机采样100个密码对并计算相似性得分，在Core i9处理器的单线程上测试了强度计。我们记录从磁盘加载模型的时间，以及计算每对相似度得分所需的平均时间。加载和解压缩η= 5（磁盘上的大小3.3 MB）的模型的平均（跨10次运行）时间为0.2秒。加载模型后，平均需要0.3毫秒来计算一对密码的相似度得分，其中99个百分位数在0.1毫秒内。

**Ⅸ.结论**

在这项工作中，我们解决了人为选择的密码在建模上的相似性，并展示了它如何构建针对破坏性的针对性猜测攻击和它的新防御措施。我们探索了两种学习密码相似性的方法：一种基于序列到序列（seq2seq）样式学习的生成模型，该模型先前用于语言翻译，另一种基于词嵌入技术的判别模型。

生成模型使我们能够构建新的针对性攻击，在这种攻击中，对手利用用户的其他密码来针对用户账户进行特定的猜测。我们证明，在不到一千个猜测的情况下，我们表现的最好攻击方法可以破坏康奈尔大学的8.4％的活动用户账户，因为该账户先前的密码已泄露。此攻击的性能比以前的最佳攻击高出3.2倍。

尽管定向攻击已经是一种广泛的威胁，但针对它们的防御措施却很少。我们唯一知道的是停止凭据填充，但是并不能阻止我们的凭据调整攻击。因此，我们提出了个性化密码强度计（PPSM），可用于警告用户不要选择在各种攻击（包括目标攻击）下容易猜到的密码。我们使用词嵌入技术构建了一个名为vec-ppsm的PPSM原型，并展示了如何将其用于缓解攻击。

**鸣谢**

感谢Tyler Kell，Dan Villanti和Jerry Shipman帮助我们进行了Cornell ITSO的实验。我们也感谢匿名评论者的深刻见解。这项工作得到了NSF赠款CNS-1514163和CNS-1564102以及美国陆军研究办公室（ARO）赠款W911NF-16-1-0145的支持。

**附录**

1. **Pass2Pass模型**

学习密码相似性的一个建议是将seq2seq方法直接应用于字符序列的密码。我们将此模型称为“密码转密码”或“ pass2pass”。编码功能将输入密码映射到实值向量∈。 解码器函数采用向量∈和字符∈{‘<’, ‘>’}，并输出{''，''}中的字符和另一个向量v的概率分布 ∈Rd，它被馈送到解码器的下一个迭代。 每个密码都包含一个特殊的序列开头符号 =“”和一个序列结尾符号“”。 因此，在该模型中，我们可以如下重写方程式（1），其中是输入和上解码器的输出。

我们使用默认的神经网络架构和seq2seq[16]中使用的超参数来训练pass2pass的几种变体。我们通过测试训练过的模型的有效性来对验证集进行有针对性的猜测攻击，这与我们稍后报告的最终测试集不同。（有关如何使用基于seq2seq的模型生成目标猜测的详细信息，请参见第V节。）最初，我们尝试使用中的密码对来训练pass2pass。然后我们将注意力集中在同一用户的在两个编辑距离之内的密码对。这加快了训练速度，似乎有助于模型集中精力于易于学习的相似性。我们还尝试了编辑距离3，但是效果比编辑距离2差。最后，我们性能最好的pass2pass模型的效果仍然不佳。基于性能最佳的pass2pass模型的有针对性的攻击只能在1,000次猜测中猜测11％的用户密码，而[14]中的最新方法则可以猜测13.1％。（请参见图3）

我们对这种性能不佳的判断是，密码具有更大的支持范围（我们拥有大约2亿个不同的密码），并且不像自然语言那样遵循任何预定义的规则（保存由密码策略设置的规则）。编辑距离的限制有助于学习，但是错过了许多重要的相似点，理想情况下可以捕获攻击。我们需要一种不同的方法。

1. **pass2path的模型架构**

Pass2path使用两个递归神经网络（RNN）—一个用于编码器功能，另一个用于解码器—一起训练，类似于seq2seq学习[16]。 RNN旨在识别序列数据中具有不同序列长度的模式。但是，vanilla RNN会遇到消失和爆炸的梯度问题。 RNN的一种变体，称为长短期记忆（LSTM）[45]，可有效避免消失和爆炸的梯度问题[46]。最初用于深度神经网络[47]的图像识别的残差单元将一层的输入“短路”到输出，从而绕过内部计算。（请参见图9（b）。）我们发现在有剩余单元的训练时间比没有使用剩余单元的训练时间明显缩短的情况下，pass2path具有更高的精度。我们在TensorFlow [48]中使用库提供的构件实现了pass2path。模型中的每个LSTM单元都有三个隐藏层，每个层具有128个隐藏单元。

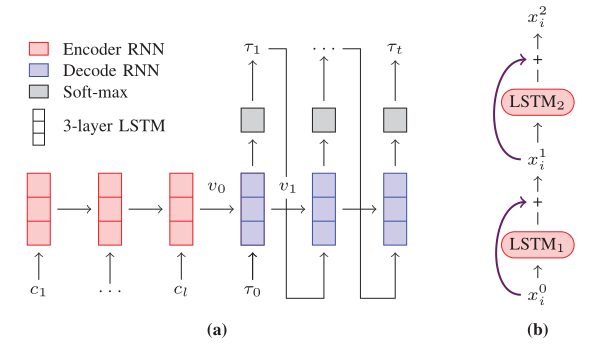


图9：（a）用于pass2path学习的编码器-解码器架构图。（b）具有残余连接的2层LSTM单元。 这里是输入密码的字符，是过渡，是神经网络的内部状态。

图9（a）给出了pass2path的神经网络架构图。编码器按顺序处理密码中的每个字符。首先将字符表示为维度|Σ|的单热点向量，并将其嵌入到维度200的实值向量中。然后将嵌入的字符馈送到具有三个隐藏层（每个维度为128）的LSTM单元。 LSTM输出两个向量，第一个向量被编码器忽略，第二个向量（称为状态）与密码的下一个字符一起被馈送到下一个LSTM单元。将其应用于整个输入字符序列后得到编码器的输出

然后将矢量与特殊的序列开始符号馈送到解码器。 解码器的架构与编码器相同，只是我们考虑了LSTM层的第一个输出，该输出被投影到大小为|T|的向量上。 将softmax函数应用于投影向量，以将其转换为T上的概率分布。 除非输出是特殊的“序列末尾”符号，否则最可能转换被视为输出，并用作解码器下一次迭代的输入。然后可以将转换输出的序列应用于输入密码以获得另一个密码。

1. **训练pass2path模型**

我们使用基于编码器-解码器的神经网络架构训练了pass2path。在这里，我们提供了训练方法的详细信息，尤其是训练之前如何初始化网络以及超参数

我们使用[16]中提出的初始化技术：使用来自[-，]的统一随机值来初始化嵌入层，而使用[-r，r]中的统一值来初始化网络的其余部分，其中r =，其中是神经网络第j层的输入维度。为了进行训练，我们使用了随机梯度下降法（SGD），并使用了Adam的优化程序[49]来最小化网络的预测输出和预期输出之间的交叉熵损失[50]。 最小化交叉熵损失（使用softmax）可确保在给定输入的情况下学习输出的条件概率。

在训练的初始阶段，我们使用teacher-forcing来训练模型，方法是将预期的输出转换作为解码器的输入，而不是预测的字符作为输入。随着训练的进行，我们开始提供实际的预测角色作为输入。我们没有使用注意力机制[51]（seq2seq语言翻译模型中使用的一种常用技术），因为与语言翻译中的句子相比，密码的大小相对较小。

我们需要为我们的架构选择一些超级参数。 除以下内容外，我们使用[16]中建议的内容。 以下是为更好的性能而设置的不同值。

（1）学习率（learning rate）。学习速率参数控制损耗梯度对模型参数变化的影响。我们使用了固定的0.0003学习率进行培训。

（2）丢弃率（dropout rate）。丢弃率控制着训练过程中神经网络单元（神经元）的随机去除，这对防止过拟合很有用[52]。我们尝试了0.3和0.4的丢弃率，后者的效果最好。

（3）图层（layers）。 每个RNN单元由多个隐藏层组成。对于语言模型，典型的隐藏层数为n∈{3,4} [46]。我们发现具有三个隐藏层的pass2path的性能优于四个层。每层包含128个隐藏单元。

（4）迭代（epochs）。epoch的数量确定训练过程在训练数据集上迭代多少次。我们发现三个epoch就足够了，继续增加训练时间但是收益却很小。

**D．从密码对生成路径**

对于每个训练输入对，我们首先使用动态编程（DP）方法计算最小编辑距离，然后回溯DP解决方案以查找导致计算出的编辑距离的实际过渡。我们根据公式计算距离矩阵。

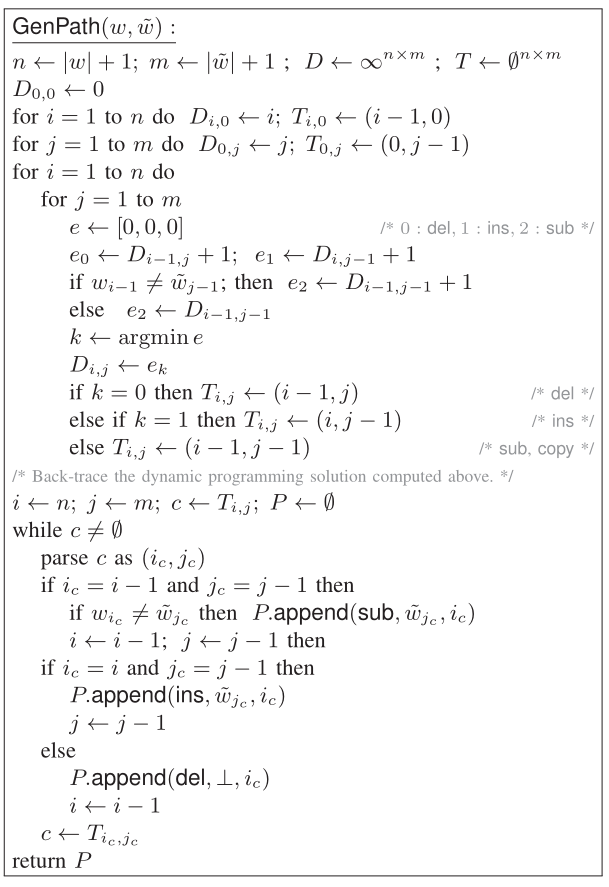


图10：从一对密码生成转换序列的GenPath算法。

图10中给出了用于生成路径的伪代码。

**E. 为我们的数据集重新训练Wang等人的方法**

Wang等人在[14]中指定的算法用于生成猜测之前需要进行训练。Wang等人共享的代码需要四个文件进行训练，所有必需的文件都是使用我们的训练数据生成的。

1）PCFG数据。该数据文件包含三个主要部分：仅包含数字（D），仅包含字母（L）和仅特殊字符（S）的密码。每个部分都有9个子部分，其长度为1到9，即，...，，，...，和，...，。每个小节包含10个最流行的密码，这些密码与该结构及其在该小节中的出现概率相匹配。例如，P（'1234'）= C（'1234'）/ C（），其中C（w）表示w的概率，或者某节中密码的概率之和。

2）马尔可夫数据。该文件包含4元字符，仅包含字母，数字和特殊字符以及它们的概率。例如，

3）反向马尔可夫数据。该文件除了n-gram是在反转密码之后计算出来的，其他与先前的马尔科夫数据文件相似。

4）最高密码数据。该文件包含来自训练数据的前104个密码及其概率。

**F.实践中有针对性的密码破解实验**

康奈尔大学拥有大规模的身份验证系统，其中包括近一百万个账户。学生，教职员工都拥有自己的账户，并且根据政策校友账户在学生毕业后不会停用。这些账户已注册到可以访问电子邮件和其他系统的单点登录（SSO）系统中，因此攻击者经常将其作为目标。

ITSO当前具有许多使远程猜测攻击变得困难的机制。（1）它们要求密码至少由八个字符组成，并且必须涵盖至少三个字符类，即大写字母，小写字母，数字或符号。此外，系统还将拒绝包含来自非公开用户标识符字典，名字和姓氏，常用密码和常用英语单词的包含一个或多个单词的密码。例如，“passw0Rd”是允许的密码，但“password”不被允许。 （2）ITSO订阅了一项服务，该服务将发现异常的账户并通知他们。如果该账户的电子邮件密码对与该违规行为相匹配，则这些账户的密码hash会被打乱，用户必须选择一个新密码才能重新获得访问权限。（3）选择新密码时，不允许用户选择旧密码。因此，ITSO使用了许多最新的保护措施，包括凭证填充对策。当然，身份验证系统已经随着时间的推移而发展，并且某些账户所选择的密码与当前的密码策略不同。我们将在下面解释这一细微差别。

**实验装置。**我们与ITSO合作，安全地利用我们的定向攻击对康奈尔账户的脆弱性进行了实验性评估。特别的，ITSO使用Kerberos来存储包括密码哈希在内的身份验证信息。我们计划对一个具有Kerberos身份验证数据库的镜像测试服务器进行访问，由ITSO工作人员进行中介和监督，这确保了我们不会干扰正常的认证通道。研究团队不会拥有直接访问该服务器的权限，所有访问权限均由ITSO员工负责，从而确保不会泄露敏感信息。我们将密码散列视为特别敏感，并在下面讨论如何安全地处理它们。

我们首先确定哪些账户出现在第III节中描述的异常数据集中。在异常数据集中发现了19,868个账户，这意味着我们至少有一个以前被确定为异常的密码。这些潜在的易受攻击账户的最后一次密码重置时间偏长，平均时间为5年，但有些账户的密码重置时间最晚是2018年。

自2009年以来，ITSO维护着一个密码更改事件日志，该日志记录了密码是由用户更改还是密码被加扰（散列方式对密码加扰,使密码不能再被解读）。在该漏洞出现的19,868个账户中，有3,106个账户的最后一个密码在2009年之前更改，因此，我们不确定这些账户中有多少部分包含密码加扰。在剩下的16762个具有最近密码更改的账户（2009年1月1日之后）中，可以肯定的有986个账户在研究之时密码已被打乱；15,776个账户具有用户选择的密码，我们将其称为有效账户。

为了比直接模拟在线攻击更快地进行实验，我们小心地将19,868个账户密码在加盐哈希后导出到受保护的研究机器，并对每个账户进行有限数量猜测的脱机哈希破解。该计算机只能从Cornell网络访问，除SSH之外没有侦听服务，需要二次身份验证才能登录，并且磁盘卷已加密。我们还通过使用4,096个SHA-256迭代对哈希散列进行哈希来进一步保护Kerberos散列，其中每个加盐散列具有128位，并且使用强胡椒（充当秘密密钥）。在ITSO完成披露程序后，我们将删除Pepper（通过密码删除这些哈希）和整个哈希数据库。

破解是在此安全服务器上执行的，该服务器是具有8GB RAM的Core i5计算机。它需要五天时间才能针对所有15776个账户测试所有4500万个密码。

我们比较了三种猜测过程——基线无目标经验攻击，Wang等人方法和pass2path的变体。基于目标的无经验攻击首先会猜测泄露的密码，然后猜测我们的异常数据集中符合Cornell密码策略的最有可能的密码。对于15776个账户中的每个账户，我们根据Wang等人的攻击算法得出了1000个猜测。不幸的是，我们无法根据Cornell密码策略对它们进行定制，因为明显的方法——拒绝采样——太慢了。

我们使用转移学习来构建pass2path模型的变体，该模型针对Cornell的密码的字符类要求而定制。我们没有尝试根据常用词词典检查进行自定义，因为该词典不是公开可用的。我们采用经过训练的pass2path模型，并在较小的数据集上再训练了三个时期，其中包含目标密码满足康奈尔字符类别要求的那些泄漏的密码对。

对于王等人的方法和自定义的pass2path，我们为每个目标账户生成了1,000个猜测，其中泄漏的密码是第一个猜测。对于具有多个泄露密码的账户，我们使用了第五节中所述的循环方法。平均而言，在目标账户中，对于Wang等人方法中的77％的猜测和自定义pass2path方法中的15％的猜测不符合康奈尔的要求。

**结果。**总体而言，这三种有针对性的攻击可能累计危害1688个账户（包括314个非活动账户，这些账户的最新密码在2009年之前更改）。我们已将这些易受攻击的账户通知ITSO。 ITSO正在采​​用一种多管齐下的方法来保护Cornell用户，包括加扰用户密码，对这些账户使用额外的监视以及在服务器端使用vec-ppsm。

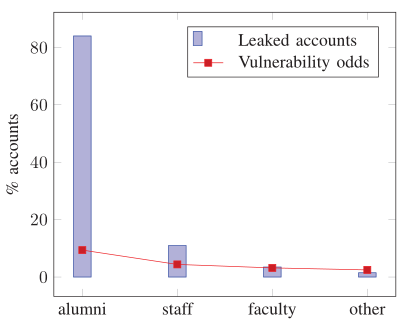


图11：在异常数据集中发现的不同类型帐户的分布以及受到三种在线猜测攻击之一攻击的几率。“其他”类别包括当前的学生，合同工和会员。

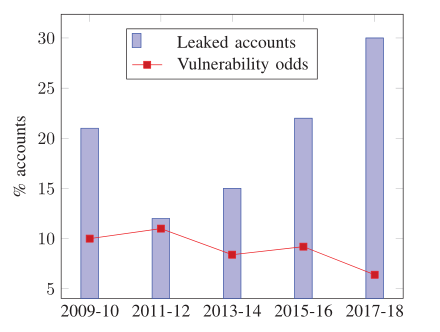


图12：根据上次重置密码的年份，活动帐户在泄漏的数据集中的分布。 红线显示他们容易受到有针对性的在线猜测攻击的可能性。

在1,688个脆弱的账户中，有93％的账户属于康奈尔大学的校友。我们的实验还发现，许多属于当前教职员工和学生的账户很容易受到有针对性的攻击，从而帮助ITSO更好地保护了这些账户。

在图11中，我们展示了泄漏中发现的账户分布以及容易受到pass2path影响的账户。我们还注意到每种类型的账户容易受到有针对性的在线猜测攻击的几率，因为每种类别中的账户中容易受到任何模拟的有针对性的在线攻击的几率。我们可以看到，与其他账户相比，校友账户更容易受到此类攻击。

有趣的是，我们还发现，最近重置的密码不太容易受到攻击。在图12中，我们显示了上次密码重置年与那些账户容易受到在线猜测攻击的可能性之间的关系。我们只考虑该图表的15776个活动账户（自2009年以来至少更改了一次密码）。与2013年之后创建的密码相比，2013年之前创建的密码容易受到针对性的在线猜测攻击的可能性高60％。

**引用**

[1] F. Stajano, “Pico: No more passwords!” in International Workshop on Security Protocols. Springer, 2011, pp. 49–81.

[2] C.-T. Li and M.-S. Hwang, “An efficient biometrics-based remote user authentication scheme using smart cards,” Journal of Network and computer applications, vol. 33, no. 1, pp. 1–5, 2010.

[3] X. Li, J. Niu, M. K. Khan, and J. Liao, “An enhanced smart card based remote user password authentication scheme,” Journal of Network and Computer Applications, vol. 36, no. 5, pp. 1365–1371, 2013.

[4] J. Bonneau, C. Herley, P. C. van Oorschot, and F. Stajano, “The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes,” in 2012 IEEE Symposium on Security and Privacy, May 2012.

[5] M. L. Mazurek, S. Komanduri, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, P. G. Kelley, R. Shay, and B. Ur, “Measuring password guess- ability for an entire university,” in Proceedings of the 2013 ACM SIGSAC 2009-10 2011-12 2013-14 2015-16 2017-18

[6] A. Adams and M. A. Sasse, “Users are not the enemy,” Communications of the ACM, vol. 42, no. 12, pp. 40–46, 1999.

[7] R. Chatterjee, A. Athalye, D. Akhawe, A. Juels, and T. Ristenpart, “password typos and how to correct them securely,” IEEE Symposium on Security and Privacy, may 2016, full version of the paper can be found at the authors’ website.

[8] S. Pearman, J. Thomas, P. E. Naeini, H. Habib, L. Bauer, N. Christin, L. F. Cranor, S. Egelman, and A. Forget, “Let’s go in for a closer look: Observing passwords in their natural habitat,” in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2017, pp. 295–310.

[9] Troy Hunt, “Have I Been Pwned?” https://haveibeenpwned.com/ Passwords/, 2018. [10] 4iQ, “Identities in the Wild: The Tsunami of Breached Iden-tities Continues,” https://4iq.com/wp-content/uploads/2018/05/2018 IdentityBreachReport 4iQ.pdf/, 2018.

[11] Shape Security, “2017 Credential spill report,” http://info.shapesecurity.com/rs/935-ZAM- 778/images/Shape-2017-Credential-Spill-Report.pdf/, 2018.

[12] P. A. Grassi, J. Fenton, E. Newton, R. Perlner, A. Regenscheid, W. Burr, J. Richer, N. Lefkovitz, J. Danker, Y. Choong et al., “Nist special publication 800-63b. digital identity guidelines: Authentication and lifecycle management,” Bericht, NIST, 2017.

[13] A. Das, J. Bonneau, M. Caesar, N. Borisov, and X. Wang, “The tangled web of password reuse.” in NDSS, vol. 14, 2014, pp. 23–26.

[14] D. Wang, Z. Zhang, P. Wang, J. Yan, and X. Huang, “Targeted on-line password guessing: An underestimated threat,” in Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. ACM, 2016, pp. 1242–1254.

[15] Y. Zhang, F. Monrose, and M. K. Reiter, “The security of modern password expiration: an algorithmic framework and empirical analysis,” in ACM Conference on Computer and Communications Security (ACM CCS), 2010, pp. 176–186. [Online]. Available: http://doi.acm.org/10.1145/1866307.1866328

[16] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in Advances in neural information processing systems, 2014, pp. 3104–3112.

[17] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” arXiv preprint arXiv:1301.3781, 2013.

[18] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” arXiv preprint arXiv:1607.04606, 2016.

[19] A. Narayanan and V. Shmatikov, “Fast dictionary attacks on passwords using time-space tradeoff,” in Proceedings of the 12th ACM conference on Computer and communications security. ACM, 2005, pp. 364–372.

[20] “John the Ripper password cracker,” http://www.openwall.com/john/, Referenced March 2014.

[21] M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek, “Passwordcracking using probabilistic context-free grammars,” in IEEE Symposium on Security and Privacy (SP), 2009, pp. 162–175.

[22] S. Komanduri, “Modeling the Adversary to Evaluate Password Strength with Limited Samples,” 2016.

[23] J. Ma, W. Yang, M. Luo, and N. Li, “A study of probabilistic password models,” in Proceedings of the 2014 IEEE Symposium on Security and Privacy (SP). IEEE Computer Society, 2014, pp. 689–704.

[24] W. Melicher, B. Ur, S. M. Segreti, S. Komanduri, L. Bauer, N. Christin, and L. F. Cranor, “Fast, lean and accurate: Modeling password guess- ability using neural networks.”

[25] B. Hitaj, P. Gasti, G. Ateniese, and F. Perez-Cruz, “PassGAN: A deep learning approach for password guessing,” arXiv preprint arXiv:1709.00440, 2017.

[26] D. L. Wheeler, “zxcvbn: Low-budget password strength estimation,” in Proc. USENIX Security, 2016.

[27] W. E. Burr, D. F. Dodson, and W. T. Polk, Electronic authentication guideline. US Department of Commerce, Technology Administration, National Institute of Standards and Technology, 2004.

[28] X. D. C. De Carnavalet, M. Mannan et al., “From very weak to very strong: Analyzing password-strength meters.” in NDSS, vol. 14, 2014, pp. 23–26.

[29] M. M. Devillers, “Analyzing password strength,” Radboud University Nijmegen, Tech. Rep, 2010.

[30] M. Dell’Amico and M. Filippone, “Monte carlo strength evaluation: Fast and reliable password checking,” in Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. ACM,

2015, pp. 158–169.

[31]J. Casal, “1.4 Billion Clear Text Credentials Discovered in a Sin-gle Database,” https://medium.com/4iqdelvedeep/1-4-billion-clear-text-credentials-discovered-in-a-single-database-3131d0a1ae14, Dec, 2017.

[32] J. Bonneau, “The science of guessing: analyzing an anonymized corpus of 70 million passwords,” in IEEE Symposium on Security and Privacy (SP). IEEE, 2012, pp. 538–552.

[33] Wikipedia, “Email address,” 2018. [Online]. Available: https://en. wikipedia.org/wiki/Email address#Local-part

[34] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” in Soviet physics doklady, vol. 10, no. 8, 1966, pp.707–710.

[35] C. M. Wilt, J. T. Thayer, and W. Ruml, “A comparison of greedy search algorithms,” in Third Annual Symposium on Combinatorial Search,2010.

[36] J. R. Rao, P. Rohatgi, H. Scherzer, and S. Tinguely, “Partitioning attacks: or how to rapidly clone some gsm cards,” in Security and Privacy, 2002.Proceedings. 2002 IEEE Symposium on. IEEE, 2002, pp. 31–41.

[37] S. Mavoungou, G. Kaddoum, M. Taha, and G. Matar, “Survey on threats and attacks on mobile networks,” IEEE Access, vol. 4, pp. 4543–4572,2016.

[38] M. Dell’Amico, P. Michiardi, and Y. Roudier, “Password strength: An empirical analysis,” in INFOCOM, 2010 Proceedings IEEE. IEEE, 2010, pp. 1–9.

[39] B. Ur, F. Alfieri, M. Aung, L. Bauer, N. Christin, J. Colnago, L. F. Cranor, H. Dixon, P. Emami Naeini, H. Habib et al., “Design and evaluation of a data-driven password meter,” in Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. ACM, 2017, pp. 3775–3786.

[40] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp.1532–1543.

[41] H. Jegou, M. Douze, and C. Schmid, “Product quantization for nearest neighbor search,” IEEE transactions on pattern analysis and machine intelligence, vol. 33, no. 1, pp. 117–128, 2011.

[42] “4iQ,” https://4iq.com/, 2018.

[43] “Lastpass,” https://lastpass.com.

[44] J. Johnson, M. Douze, and H. Jégou, “Billion-scale similarity search with gpus,” arXiv preprint arXiv:1702.08734, 2017. [Online]. Available: https://github.com/facebookresearch/faiss

[45] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[46] M. Sundermeyer, R. Schlüter, and H. Ney, “Lstm neural networks for language modeling,” in Thirteenth annual conference of the international speech communication association, 2012.

[47] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[48] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard et al., “Tensorflow: a system for large-scale machine learning.” in OSDI, vol. 16, 2016, pp. 265–283.

[49] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[50] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, “Sequence level training with recurrent neural networks,” arXiv preprint arXiv:1511.06732,2015.

[51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez,L. Kaiser, and I. Polosukhin, “Attention is All You Need,” in Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.

[52] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhut-dinov, “Dropout: a simple way to prevent neural networks from over-fitting,” The Journal of Machine Learning Research, vol. 15, no. 1, pp.1929–1958, 2014.