Towards the Detection of Inconsistencies in Public Security Vulnerability Reports

CDra90n / 2019-09-01 10:19:00 / 浏览数 3068 安全技术 技术讨论 顶(1) 踩(0)

浅谈公共安全漏洞报告（CVE&NVD）中不一致性的检测
原文章发表于USENIX2019：https://www.usenix.org/conference/usenixsecurity19/presentation/dong

摘要

公共漏洞库如公共漏洞和暴露（CVE）、国家漏洞数据库（NVD）在促进漏洞披露和缓解方面取得了巨大的成功。虽然这些数据库已经积累了大量数据，但人们对它们的信息

使用VIEM和过去20年里的78,296个CVE
ID和70,569个漏洞报告组成的大数据集检查信息一致性。结果表明，漏洞软件版本不一致非常普遍。只有59.82%的漏洞报告/CVE摘要严格匹配标准化NVD条目，并且不一致

一、漏洞报告介绍

CVE。当人们发现一个新的漏洞时，他们可以向CVE编号机构（CNAS）请求一个唯一的CVE-ID号。Mitre公司是编辑和主要CNA。然后，CNA将对漏洞进行研究以确定详细
ID，并通过CVE列表公开发布相应的漏洞信息。

CVE列表由MITRE维护，作为一个网站，CVE团队在其上发布每个报告的漏洞的摘要。在编写CVE摘要时，CVE团队将分析（公开）第三方漏洞报告，然后在其描述中包括详

除了摘要之外，每个CVE条目还包含一个外部引用列表。外部参考是指向第三方技术报告或博客/论坛帖子的链接，这些报告或博客/论坛帖子为CVE团队提供所需的信息，以

NVD。NVD（国家漏洞数据库）由与CVE不同的组织（即NIST）维护。NVD与CVE列表完全同步。目标是对CVE的任何更新都将立即出现在NVD中。当一个新的CVE
ID出现在CVE列表上后，NIST NVD团队将首先进行分析，以在创建NVD条目之前添加增强信息，如严重性评分。

与CVE相比，NVD提供了两个附加功能。首先，NVD数据条目是结构化的。NIST
NVD团队会将非结构化的CVE信息转换为结构化的JSON或XML，其中，易受攻击的软件名称和版本等信息字段将根据通用通用弱点枚举规范（CWE）进行格式化和标准化。

CVE和NVD数据库主要通过人工维护，这导致了许多重要问题。首先考虑到可能会在许多不同的地方报告和讨论一个漏洞，CVE/NVD数据库中的信息（例如，漏洞软件名称

二、VIEM设计

为了精确定位和匹配感兴趣的对象，设计了VIEM（漏洞信息提取模型）来完成三个单独的任务为了便于未来的研究和应用开发，标记数据集和VIEM的源代码在：https://gi

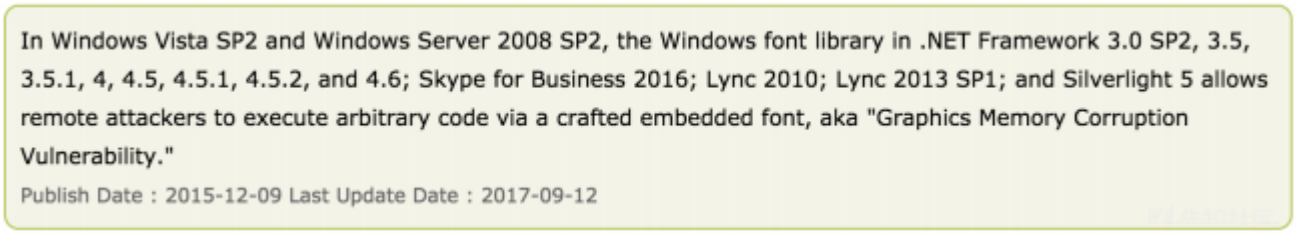
命名对象识别模型。首先，VIEM利用最先进的命名对象识别（NER）模型来识别感兴趣的对象，即漏洞软件的名称和版本、漏洞组件的名称和版本以及漏洞软件依赖的基础

这种设计背后的原因有两个。首先，NER模型根据输入文本的结构和语义来定位对象，这使得能够跟踪训练数据中未观察到的软件名称。第二，NER模型可以学习和区分与漏

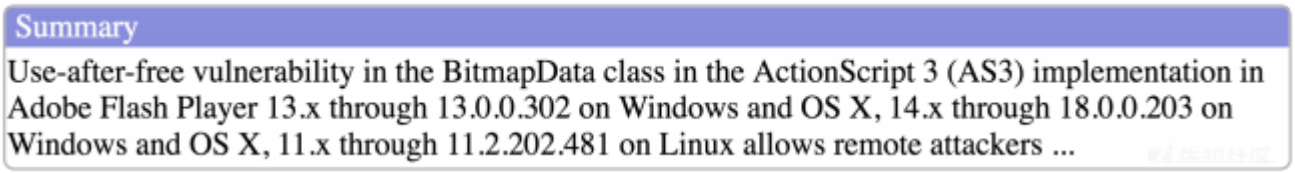
关系提取模型。对于提取的对象，VIEM的下一个任务是对已识别的对象进行相应的配对。软件名称和版本共同出现在报表中是很常见的。因此，一种本能的反应是将附近的

为了解决这个问题，VIEM首先要进行版本和软件名称之间所有可能的组合。然后，它使用关系提取（RE）模型来确定最可能的组合，并将它们视为正确的对象对。这种设计
in",“employed by” 和“capital of”。给定在文本“Steve Jobs was born in California,and was the CEO of Apple.”中的两对对象P1 = (“Steve Jobs”, “Apple”) 和
P2 = (“Steve Jobs”,“California”)，一个RE模型将“employed by”分配给P1，将“born in”属性分配给P2。

(a)Openwall报告，其中包含漏洞的软件版本(2.3.x)和非漏洞版本(3.0.0及更高版本)。



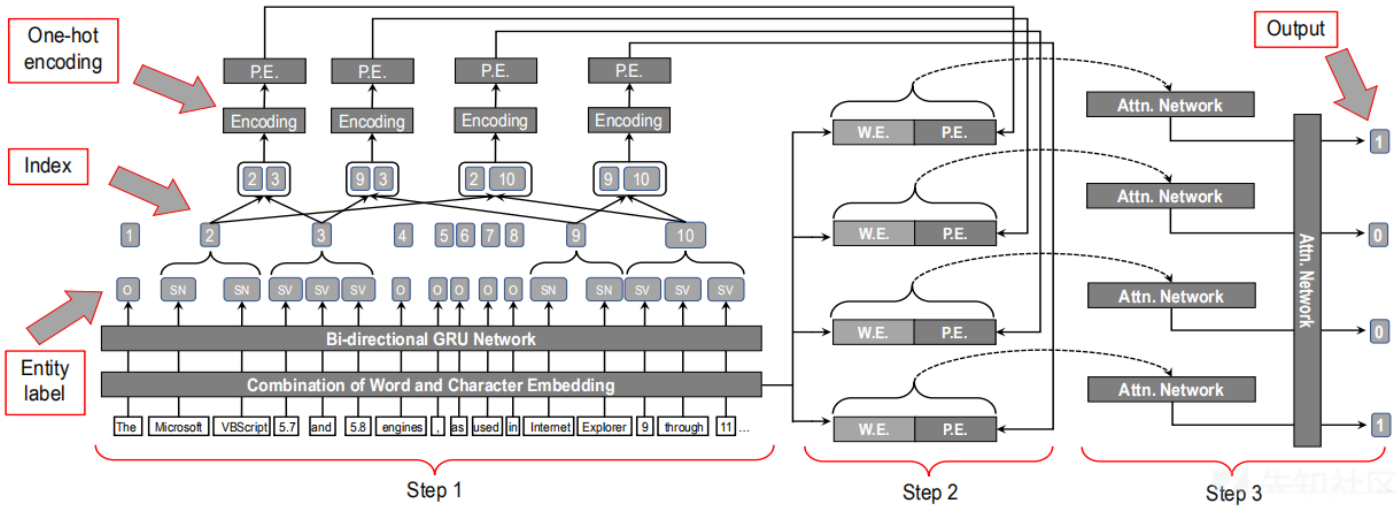
(b)包含与漏洞软件相关的多个实体的CVE摘要（易受攻击组件：Windows font library;漏洞的软件：.NETFramework、SkypeforBusiness、Lync、Silverl。
从属软件：windows；绑定到这些实体的软件版本)



(c)CVE摘要，其中易受攻击的软件的名称和版本不相邻

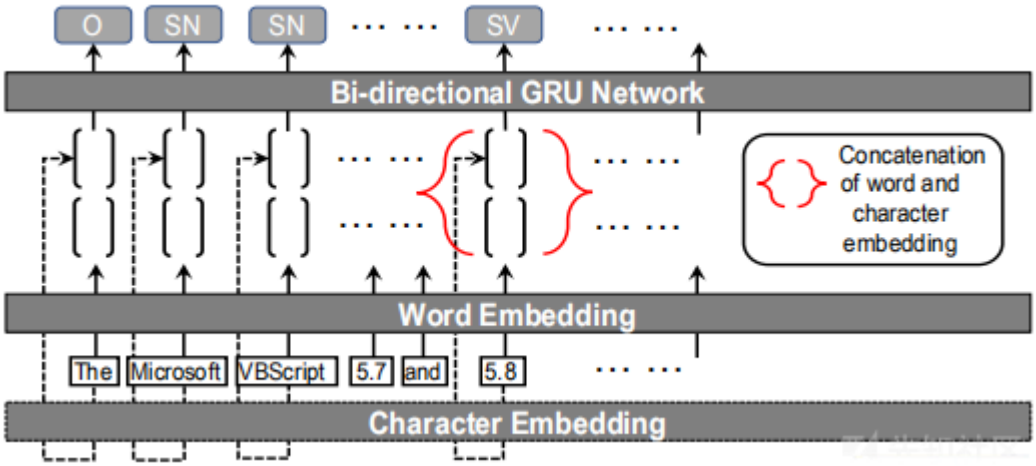
在模型中，每个可能的版本和软件组合都可以被视为一对单独的对象。利用关系提取模型的思想，VIEM为每对对象分配一个属性，表明对应实体关系的真实性。然后，它将
vbscript和Internet

Explorer）表示，2个版本范围（5.7、5.8和9到11）。它们可以用4种不同的方式组合。通过将这些组合视为4对不同的对象，可以使用一个RE模型为每个组合分配一个二分



首先从文本中提取命名对象（漏洞软件名称和版本）,除此之外还集成了一个名录，以提高其提取漏洞软件名称的准确性。在高层，NER模型首先使用单词和字符嵌入的串联

对于单词和字符嵌入，NER模型首先使用标准的单词嵌入方法将每个单词编码为矢量表示。然后，它利用双向门控循环联合（Bi-GRU）网络在字符级别执行文本编码。如下

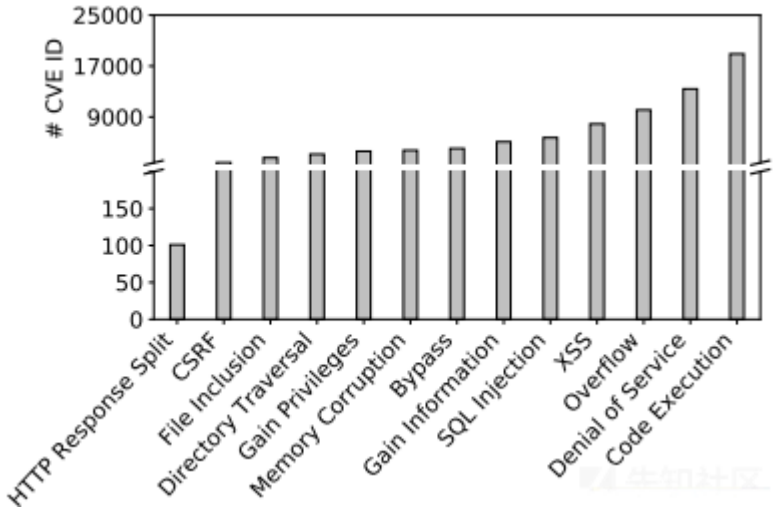


迁移学习。本文需要测评不同漏洞类型的漏洞报告，不同漏洞类型的报告不一定共享相同的数据分布。因此，使用单一机器学习模型来处理所有漏洞报告是不可行的，除非

三、数据集

本实验收集了过去20年大量的公共漏洞报告和CVE和NVD条目。对这些漏洞报告中的一个子集进行人工标记（真实值），并使用标记的数据评估VIEM的性能。

CVE IDs。首先从cvedetails.com获取一个CVE
ID列表，将cve/nvd数据库中索引的安全漏洞分为13类。为了收集公开报告的漏洞的代表性数据集，对从1999年1月到2018年3月（超过20年）的每个漏洞类别的CVE
ID进行了爬行。CVE ID是公开漏洞的唯一标识符。尽管CVE网站声称他们拥有105000多个CVE ID，但其中许多CVE
ID尚未公开，或已被合并或撤销。总共获得了78296个CVE ID，涵盖了下图所示的所有13个类别。每个CVE ID对应于表1所示漏洞的简短摘要。



漏洞报告。每个CVE
ID的网页还包含指向外部报告的外部引用列表。本文研究集中在5个具有代表性的源网站上，以获取CVE引用的漏洞报告，包括ExploitDB、SecurityFocus、SecurityTracke
总共获得了与56,642个CVE ID相关的70,569个漏洞报告。 这些CVE ID覆盖了所有78,296个公共CVE ID的72.34%。 这意味着72.34%的CVE
ID都有来自5个源网站之一的漏洞报告，证实了它们的受欢迎程度。有来自SecurityTracker和SecurityFocus的45，812份结构化报告，以及DevelopitDB、OpenWall和Se

NVD条目。 对于每个CVE ID还解析NVD条目的JSON版本，其中包含结构化数据字段，例如易受攻击的软件名称及其版本。 总共获得78,296个NVD条目。

数据提取和预处理。 对于结构化报告，直接按照固定格式解析易受攻击的软件名称和版本信息。
对于非结构化漏洞报告和CVE摘要，使用NLTK工具包提取文本信息，删除所有Web链接，并对句子进行标记化。
请注意没有从非结构化文本中删除任何停用词或符号，因为它们通常是软件名称和版本的一部分。

四、模型评估

使用内存损坏漏洞报告及其CVE摘要(3，448个CVE ID)的数据对NER和RE模型进行评估。

1）NER模型。 给定一个文档，NER模型会提取漏洞软件名称和漏洞版本。
提取过程首先在单词级别，然后具有SN或SV标签的连续单词将被分组为软件名称或软件版本。
在单词级别提取中使用三个评估指标：（1）精确度表示相关对象在提取的实体上的比例;（2）召率回代表在相关对象总数中提取的相关对象的比例;
（3）总体准确度代表所有预测的正确预测的分数。 分别计算软件名称提取和版本提取的精度和召回率。

以8：1：1的比例切分数据集，用于训练，验证和测试。 这里预训练的单词嵌入的维度是300， 为了对齐输入序列，只考虑每个句子的前200个单词。
根据经验，观察到绝大多数句子短于200个单词， 除了单字级嵌入权重W（使用FastText方法）之外，NER模型中的所有层都被联合训练。
默认批量大小为50，轮次数为20。采用一种改进的随机梯度下降方法ADAM作为优化器，可以自适应地调整学习速度减少收敛时间，也采用丢弃法（dropout）来防止过拟

通过随机分割数据集进行10次重复实验，下表中显示了平均精度，召回率和准确度。即使不使用名录（即字典），NER模型也非常准确。
可以提取漏洞软件名称和版本，精度为0.978，召回率为0.991。 此外，名录字典证明软件名称提取的性能是可预期的。 应用名录后，整体精度高达0.9969。
这种高精度的NER是理想的，因为任何错误都可能传播到后来的RE模型。

Metric		w/o Gazetteer	w/ Gazetteer
Software Version	Precision	0.9880	0.9880
	Recall	0.9923	0.9923
Software Name	Precision	0.9773	0.9782
	Recall	0.9916	0.9941
Overall	Accuracy	0.9969	0.9970

2）RE模型。 首先检查RE模型本身的性能，然后通过组合NER和RE来评估端到端性能。 与之前类似，将数据集与8：1：1的比例分开进行训练，验证和测试。
在这里将预训练的单词嵌入的维度设置为50。位置嵌入的维度为10。默认批量大小为80，轮次数为200，将双向图层的数量设置为2
。与NER模型一样，RE模型也使用预训练的单词嵌入权重W。位置嵌入权重（即Ws和Wv）随机初始化并与模型中的其他参数一起训练。

首先进行实验以单独评估RE模型。 更具体地说，假设已经正确提取了命名对象，只使用RE测试"配对"过程。 这假设早期的NER模型具有完美的性能。
如下表所示RE模型也非常准确。 该模型的精度为0.9955，召回率为0.9825

Metric	Ground-truth Software Name/Version as Input	NER Model's Result as Input	
		w/o Gazetteer	w/ Gazetteer
Precision	0.9955	0.9248	0.9411
Recall	0.9825	0.9931	0.9932
Accuracy	0.9916	0.9704	0.9764

其次评估端到端性能，并使用NER的输出作为RE模型的输入。 这样，NER的错误可能会影响RE的性能。
如上表所示，准确度从0.9916降至0.9704（没有名录）和0.9764（使用名录）。
退化主要发生在精确度上。进一步的检查表明，NER模型错误地提取了一些非软件名称的对象，这些对象成为RE的错误输入并损害了分类精度。
此外，在NER和RE组合后，名录的好处也出现了，将精度从0.9248提高到0.9411（不会损害召回率）。
结果证实模型能够准确地从非结构化文本中提取漏洞软件名称和相应的版本。

五、不一致性测评及结果

匹配软件名称。 给定CVE ID，首先匹配NVD数据库中列出的漏洞软件名称和非结构化文本中列出的漏软件名称。 更具体地说，让C =
{（N1，V1），（N2，V2），...，（Nn，Vn）}是从NVD中提取的漏软件名称 - 版本对，并且C'= {（N1， V '1），（N2，V'2），...，（Nm，V'
m）}是从外部文本中提取的名称版本对。 在数据集中，大约20%的CVE
ID与多个软件名称相关联。本文只关注NVD和外部端口之间匹配的软件名称，本文匹配方法可以灵活地处理相同软件名称略微不同的格式。如果匹配词的数量高于或等于不
Internet Explorer"和"Internet Explorer"匹配，因为匹配的单词比不匹配的单词更多。

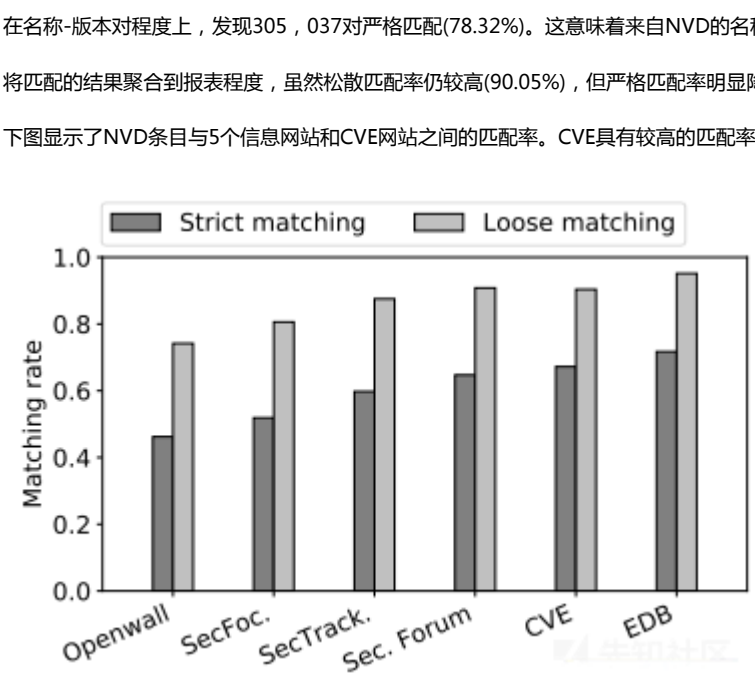
版本一致性测评。给定软件名称N1，寻求测评报告版本V1和V′1的一致性。 检查两种类型的匹配。首先严格匹配即V1和V′1彼此完全匹配（V1 = V′1）。
其次，松散匹配意味着一个版本是另一个版本的超集（V1⊃V′1或V1⊂V′1）。 请注意，松散匹配的案例包含严格匹配的案例。
除了松散匹配之外，它意味着V1和V′1每个都包含一些不被另一个报告的漏洞版本（即冲突的信息）。

在所有78296个CVE
ID中，总共提取18764个唯一的漏洞软件名称。这些漏洞软件名称对应于CVE摘要中的154569个软件名称-版本对、外部漏洞报告中的235350个名称-版本对以及NVD数据库

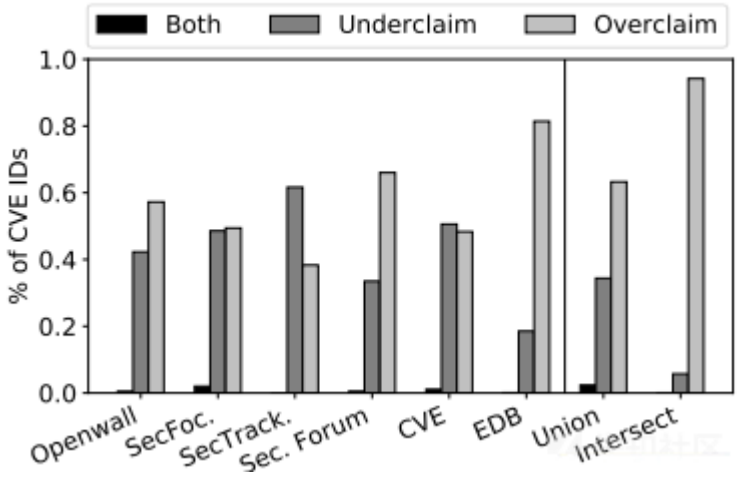
在名称-版本对程度上，发现305，037对严格匹配(78.32%)。这意味着来自NVD的名称版本对中约有22%与外部信息源不匹配。如果在松散匹配条件下，发现361，005对松

将匹配的结果聚合到报表程度，虽然松散匹配率仍较高(90.05%)，但严格匹配率明显降低。只有59.82%的漏洞报告/CVE摘要严格匹配NVD条目。这是因为严格匹配的报告要

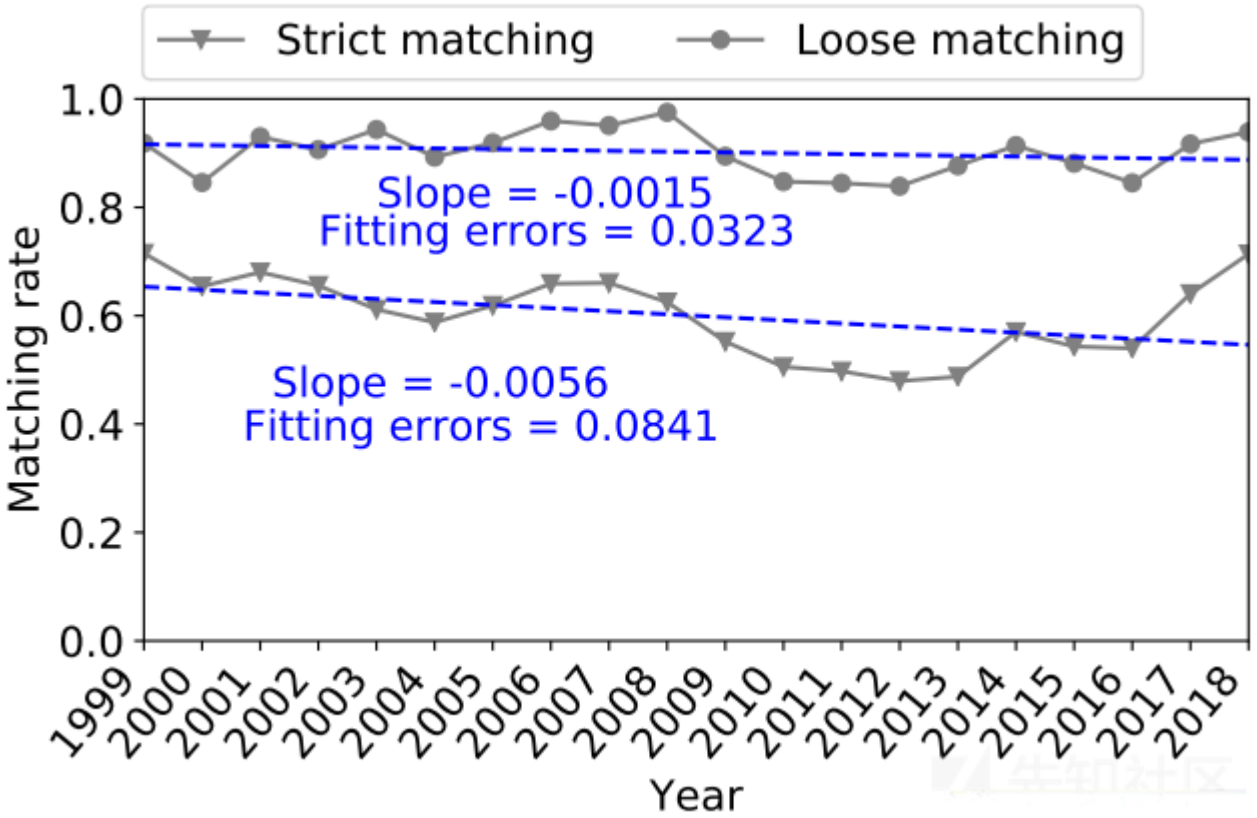
下图显示了NVD条目与5个信息网站和CVE网站之间的匹配率。CVE具有较高的匹配率（约70%的严格匹配率）。考虑到NVD据称与CVE反馈同步，这并不奇怪。更有趣的是



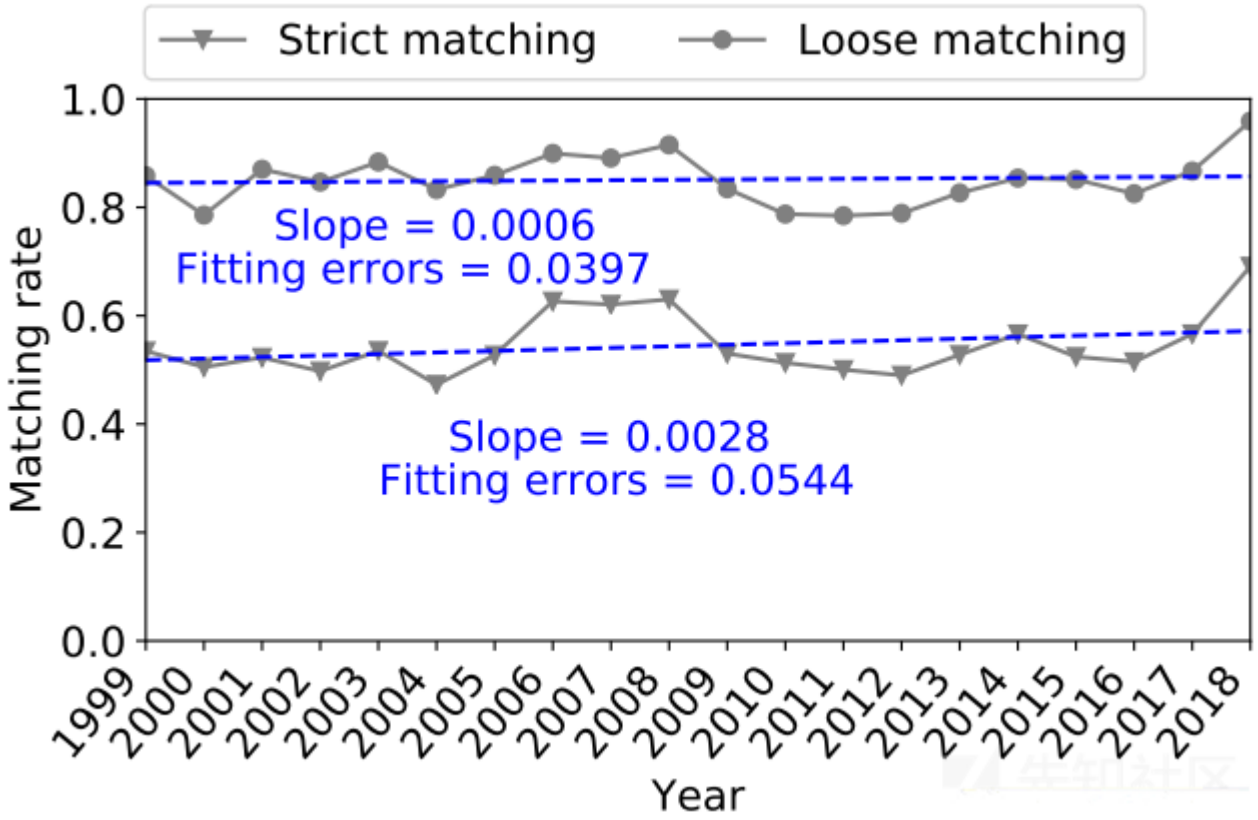
下图显示了松散匹配对中声明漏洞版本范围过高或过低的NVD条目的百分比。此分析中不包括"严格匹配"对，并不惊讶地发现NVD条目声明漏洞版本范围过高。考虑到NVD



下图显示了NVD和其他6个信息源（CVE和5个报告网站）之间的一致性水平随着时间的推移而下降。严格匹配率随时间有一定波动，但仍呈下降趋势。对两个匹配率进行线



当比较CVE和5个外部网站之间的一致性时，下图显示了不同的趋势。CVE与外部站点的一致性水平相对稳定，略有上升趋势。对两个匹配率进行线性回归，这两个匹配率均



最后，通过分析与外部报告发布时间相关的NVD条目创建/更新时间来推断不一致的原因。更具体地说，NVD为每个CVE
ID维护一个"更改历史记录"，这允许提取条目创建时间以及添加/删除新软件版本的时间。然后可以将其与5个网站上相应报告的发布时间进行比较。为此随机选择了5000个C
ID，其在NVD中的易受攻击版本与5个网站中的版本不一致。

本文发现66.3%的NVD条目自第一次创建以来从未更新过。这包括5.8%的NVD条目，这些条目是在5个网站中的任何一个发布报告之前创建的。例如，对于CVE-2006-6516
of Gnome 3.10.2在2016年8月存在漏洞。一个月后，创建了不包括3.10.2版本的NVD条目。自那时以来，没有进行任何更新。

对于剩下的33.7%的NVD条目，它们在条目创建后至少漏洞版本进行了一次更新。对此比较了5个网站上外部报告的最新更新时间和发布时间，发现所有的NVD条目在一些外

本文得出的结果意味着系统管理员或安全分析师不能简单地依赖NVD/CVE信息来确定受影响软件的漏洞版本。
至少，浏览外部漏洞报告有助于更好地覆盖潜在的漏洞版本。深入的案例研究证实，NVD/CVE数据库和第三方报告要么错过了真正漏洞软件版本，要么错误地包含了非漏洞

错误的信息可能会使漏洞软件无法打补丁，或者增加安全分析师的手动工作以进行风险评估。 社区正日益需要系统地纠正漏洞报告中不准确的声明。

上一篇：关于thinkphp使用bind注... 下一篇：Google Bugbounty:...

1. 0 条回复
   - 动动手指，沙发就是你的了！