doi:10.11959/j.issn.1000-436x.2015218

UVDA：自动化融合异构安全漏洞库框架的设计与实现

温涛[1]，张玉清[1,2]，刘奇旭[2,3]，杨刚[2]

（1. 西安电子科技大学 综合业务网理论及关键技术国家重点实验室，陕西 西安 710071；
2. 中国科学院大学 国家计算机网络入侵防范中心，北京 101408；
3. 中国科学院 信息工程研究所 信息安全国家重点实验室，北京 100093）

摘 要：安全漏洞是网络安全的关键，漏洞库旨在收集、评估和发布安全漏洞信息。然而，漏洞库相互之间存在数据的冗余和异构，导致漏洞信息共享困难。针对上述问题，收集和分析了 15 个主流漏洞库共计 84.2 万条漏洞数据。基于文本挖掘技术提出了漏洞去除重复的规则（准确率为 94.4%），以及漏洞数据库融合（UVDA, uniform vulnerability database alliance）框架。最后在多个漏洞库上，实现了 UVDA 框架，实现过程完全自动化。生成的 UVDA 数据库已经应用于国家安全漏洞库，并且可以按照产品型号和时间进行统一的检索，推进了漏洞信息发布机制标准化进程。

关键词：信息安全；数据融合；漏洞数据库；文本挖掘；UVDA

中图分类号：TP309.1 **文献标识码**：A

UVDA: design and implementation of automation fusion framework of heterogeneous security vulnerability database

WEN Tao[1], ZHANG Yu-qing[1,2], LIU Qi-xu[2,3], YANG Gang[2]

(1. State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China;
2. National Computer Network Intrusion Protection Center, University of Chinese Academy of Sciences, Beijing 101408, China;
3. State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China)

Abstract: Security vulnerability was the core of network security. Vulnerability database was designed to collect, assess and publish vulnerability information. However, there was redundant and heterogeneous data in vulnerability database which leads to sharing difficulty of vulnerability information among vulnerability database. 15 main vulnerability database with a total of 842 thousands of vulnerability data items were connected and analyzed. Based on text mining technology, a rule of removing duplicate form vulnerabilities whose accuracy rate was 94.4% and vulnerability database fusion framework(UVDA) were proposed. Finally, three representative vulnerability database were used to realize UVDA framework, which made the process fully automatic. The generated UVDA vulnerability database has been used in national security vulnerability database and can be retrieved according to uniform product version and date time, promoting the standardization process of vulnerability information release mechanism.

Key words: information security; data fusion; vulnerability database; text mining; UVDA

1 引言

安全漏洞是网络安全的关键[1]，当漏洞出现的时候，如果没有及时地、有针对性地安装补丁，设计再完美的安全防御机制，效果都将大打折扣[2]。安全漏洞库是漏洞信息收集、管理和发布平台，能够使用户及时全面地了解漏洞，从宏观的角度把握漏洞。安全漏洞的重要性主要体现在以下 2 个方面，

收稿日期：2014-12-08；**修回日期**：2015-03-08

基金项目：国家自然科学基金资助项目（61272481, 61303239, 61572460）；北京市自然科学基金资助项目（4122089）；中国科学院信息工程研究所信息安全国家重点实验室开放课题基金资助项目（2015-MS-04）

Foundation Items: The National Natural Science Foundation of China (61272481, 61303239, 61572460);Beijing Municipal Natural Science Foundation (4122089); Open Project Program of the State Key Laboratory of Information Security(2015-MS-04)

1) 从系统安全的角度来看[3]，补丁的安装存在风险（尤其是大型系统），可能导致系统崩溃、重要数据丢失、关键业务功能禁用等诸多问题，因此需要及时、准确和全面地了解漏洞的属性，从而快速正确地做出决策[4]；2) 从宏观政策的角度来看，漏洞的数量及类型往往是当前和历史网络安全的缩影，政策的制定者需要了解各种类型漏洞的宏观统计规律和趋势[5]，从而对政策的制定提供参考，最大程度地减少网络安全漏洞带来的危害。

因此，如何构建一个统一的漏洞信息发布平台，以保证漏洞信息能够准确、及时、全面发布将尤为重要。漏洞库的建设得到了软件厂商[6]、安全公司[7]、政府部门[8,9]以及科研工作者[10~12]的广泛关注。软件厂商率先建立了针对自身产品的漏洞发布平台，但是用户要想获取全面的漏洞信息，必须逐个查询这些平台。随后，安全公司和政府部门先后建设了安全漏洞库，每个漏洞库可以包含多个软件厂商的漏洞信息，在一定程度上整合了漏洞的发布[13]。然而，这些漏洞库各自所覆盖的漏洞范围有所不同，相比于理想情况下的统一漏洞发布仍然有较大的差距，这是由于：1) 每个漏洞库关注的漏洞类型不同[14]；2) 每个漏洞库的数据来源不同[15]；3) 每个漏洞库记录的漏洞信息字段不同[16]。上述差异[17]导致漏洞库的全面性仍然没有得到很好的体现，迫切需要在现有漏洞库的基础上构建一个更加全面统一的标准漏洞库。

如图 1 所示，目前厂商漏洞发布平台向综合漏洞库的转化已经基本完成，下一步的工作重心将是综合漏洞库向统一标准漏洞库的转化。标准漏洞库的建设主要存在以下 2 方面的挑战。

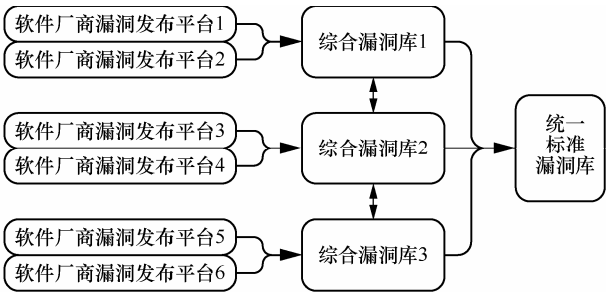


图 1 漏洞整合趋势示意

1) 漏洞数据库之间存在冗余。漏洞数据库之间既包含自身特有的数据[18]，也包含相互重合的数据；数据的来源既有部分是独立获取的，也有一定比例来源于其他漏洞库。因此漏洞数据库的融合不应当是简单的数据堆砌，融合后的数据中不同的条目不应当指向同一个实际的漏洞，而应当满足每一个漏洞记录条目的唯一性。而目前尚不存在判断漏洞同一性的标准。

2) 漏洞数据库之间存在异构。在不同的漏洞库中，漏洞的文本类型字段的表达方式存在较大差异，对于同一个含义的名词，当出现在不同的漏洞库中时，软件难以识别其同一性。因此导致数据融合难以自动化、批量化的进行，而如果采用人工的方式逐个完成，工作量将十分庞大[19]，且不可避免主观性。

针对上述挑战和现有研究工作的不足，考虑到文本挖掘[20]在自动发现已知历史信息的规律和预测未知信息方面的强大功能[21]，本文基于文本挖掘理论提出了一个漏洞数据融合框架。本文的主要贡献总结如下。

1) 本文收集了 NVD（national vulnerability database）等国内外 15 个漏洞库 84.2 万条漏洞数据；分析和归纳了漏洞文本类型字段的异构情况，同时分析和整理了漏洞参考链接引用的拓扑结构，利用该拓扑结构归纳了漏洞之间可能存在的主要关系。

2) 以文本挖掘算法为核心，针对漏洞字段的特点，设计和实现了“漏洞字段相似度算法”。利用该算法在本文中计算了漏洞厂商和产品型号字段字符串的相似度，实验结果表明该算法的正确率为 94.4%，适用率为 100%。有效地解决了漏洞文本字段的异构问题。

3) 提出了“漏洞同一性判别规则”，用于判别不同漏洞库中的 2 条漏洞信息是否指向同一个漏洞。实验结果表明，结合“漏洞字段相似度算法”，判别正确率为 94.4%，覆盖率为 100%。有效解决了漏洞库之间的冗余问题。

4) 提出了漏洞数据库自动化融合（UVDA, uniform vulnerability database alliance）框架，旨在自动化地将已有的漏洞数据库融合为一个统一的漏洞数据库。融合后的 UVDA 数据库具有以下优点：①可以按照厂商、产品型号和发布时间进行统一的检索和统计，而不再需要单独检索每个漏洞库；②漏洞数据库可以阶段性、模块化的添加，而不需要一次性完成全部添加；③融合过程完全自动化，不需要人工参与。

5) 采用 3 个当前有代表性的漏洞库，对 UVDA 框架进行了实现，产生了新的 UVDA 漏洞数据库和 UVDA 厂商产品型号数据库。将实现后的 UVDA

漏洞数据库和原数据库进行了比较，取得了较好的效果，验证了 UVDA 框架的可行性。

2　相关工作与异构数据的分析

2.1　主流漏洞库和标准

SCAP（security content automation protocol）[22]是一个权威的漏洞管理标准，旨在标准化漏洞的各个属性字段。目前仅有 NVD 漏洞数据库全面的支持 SCAP 标准。CVE[23]、CVSS 和 CWE 都是 SCAP 的子标准，分别用于定义漏洞编号、确定漏洞的危害和漏洞分类。目前仅有 NVD 完全采用 SCAP 来展现漏洞信息。UVDA 框架将为 SCAP 标准的全面推广提供数据平台，为漏洞数据的统一化管理和共享奠定基础[24]。

表 1 列出了 15 个国内外主流的漏洞库，给出了每个漏洞库的漏洞总量，有 CVSS 和 CWE 值的漏洞占漏洞总数的比值，例如 SecurityFocus 中有 40%的漏洞具有 CVSS 分值。国外主流的漏洞库包括美国国家漏洞库 NVD[8]、IBM 旗下的 X-Force[6]等；国内知名漏洞库包括绿盟科技的 NSFocus[7]等。

表 1　　国内外主流漏洞库

漏洞库名称 (网址)	数量（万）	参考链接	产品型号	CVSS 分级	CWE 分类
NVD	6.9	100%	100%	100%	100%
SecurityFocus	7.0	100%	100%	42%	42%
OSVDB	11.4	100%	100%	64%	64%
X_Force	9.6	100%	100%	69%	69%
Secunia	5.3	100%	100%	49%	49%
EDB	3.0	100%	92.0%	59%	59%
CXSecurity	2.2	100%	100%	49%	49%
PacketStorm	3.5	100%	100%	6%	6%
CNVD	6.1	100%	100%	48%	48%
CNNVD	7.3	100%	100%	94%	94%
NIPC	7.0	100%	100%	93%	93%
SCAP 中文	7.3	100%	100%	95%	95%
Sebug	2.2	100%	100%	14%	14%
NSFocus	3.0	100%	100%	48%	48%
Wooyun	2.4	100%	100%	0%	0%
均值	5.6	100%	99.5%	55.3%	55.3%
总值	84.2	—	—	—	—

2.2　漏洞数据融合的相关研究

近年来，国内外关于漏洞库合并的研究也有了一定的进展，Zheng 等[18]提出了国际漏洞联盟的概念，旨在将当前的数据库扩展到不同国家和语言，增加漏洞的数量，但是并没有给出具体的实现方案。Wang[19]、Arnold[25]、Gu[26]分别设计了关于漏洞数据的融合方案，但是上述算法只能处理采用 SCAP 协议表示的漏洞，具有较大的局限性。王晓甜[27]提出，对于没有采用 SCAP 协议的漏洞可以根据参考链接进行融合，然而参考链接的引用关系十分复杂，单纯依靠参考链接来融合数据准确率低下，而本文将在考虑参考链接的基础上结合产品型号字段进一步融合数据。

关于漏洞数据融合，目前主要有如下 4 个框架：1) VSMH[19]（vulnerability similarity measurement of hierarchy）；2) ABVD[25]（automatically building framework of vulnerability database）；3) UFVD[26]（unification framework of vulnerability database）；4) SVAS[27]（security vulnerability auto-collection system）。上述 4 个框架，以及本文提出的 UVDA 框架，所采用漏洞字段各不相同，如表 2 所示。同时每个漏洞字段的特点也不同。

表 2　　各个算法用到的字段

算法	参考链接	受影响厂商与产品型号	CVSS 指标	漏洞类型
VSMH	—	采用	采用	采用
ABVD	—	采用	采用	—
UFVD	采用	—	采用	—
SVAS	采用	—	—	—
UVDA	采用	采用	—	—

表 3 从适用率、准确率和计算量 3 个方面比较了各个字段的特点。每个字段的特点将影响到框架的特点，例如 VSMH[19] 需要用到 CVSS 指标字段，但是该字段只有部分漏洞具备，因此 CVSS 只能适用于这些漏洞，如果强制对不含有该字段的漏洞进行处理，则会在较大程度上降低准确率。

表 3　　各个字段特点比较

分析指标	参考链接	受影响厂商与产品型号	CVSS 指标	漏洞类型
适用率	好	好	差	差
准确率	中	好	差	差
计算量	中	差	好	好

需要指出的是，本质上来说，主流漏洞库与上述融合框架要处理的数据都是异构的，即都是为了将格式不同的数据整理成为可以统一存储和检索的数据。但是融合框架的优势在于自动化和批量化的完成这一步骤。主流漏洞库完成了初步的去异构化，融合框架则在此基础上自动化的进一步去异构化，最终实现一个全面统一完整的数据库。融合框架处理漏洞数据的速度大约是主流漏洞库的数百倍。

2.3 文本类型字段的异构情况分析

由于每个漏洞库的表现风格不同，漏洞数据库之间存在严重的异构，对于同一个单词或者内容，在不同的库中写法不同。在本文中，通过分析表 1 中的所有漏洞，以漏洞受影响产品型号为例，发现 NVD 和其他漏洞库中对于同一个产品型号，写法完全相同的概率平均小于 10%。关于漏洞描述等其他字段，在写法上同样存在着差异。本文归纳了如下几类常见的问题。

1) 单词异常或错误：①书写错误，例如 http:// 会写成 http;//；②特殊写法，例如产品型号 facile，在一些漏洞库中会写成 F@cile；③书写风格不同，例如 http://www.osvdb.org 会写成 http://osvdb.org。

2) 单词整词错位：①表达顺序不同，例如产品型号 cgi_guestbook，会写成 Guestbook CGI；②单词缺失，例如产品型号 Symantec Corporation PGP Whole Disk Encryption，会写成 Symantec pgp。

3) 单词变形：①缩写，例如产品型号 Internet Explorer，会写成 IE；②合成词，例如产品型号 Kevin ReynenCreative Commons Module，会写成 creativecommons；③时态和单复数，例如单词 compute，会写成 computes。

2.4 漏洞参考链接的分析

本文分析和整理了 15 个漏洞库中所有漏洞参考链接引用的拓扑结构，利用该拓扑结构归纳了漏洞之间可能存在的关系，主要关系如下。1)“相同”：通过参考链接进行数据融合，在理想情况下为一对一，如 NVD 中的一条漏洞 *A* 引用了 X_Force 中的一条漏洞 *B*，而 *B* 没有再引用 NVD 中除了 *A* 以外的其他漏洞，也没有再被 NVD 中除了 *A* 以外的其他漏洞所引用，这种情况便是参考链接的一对一引用。本文认为这种情况下，2 个之间的关系是“相同”。2) “相关”：实际中往往存在多对多的问题，如一条 NVD 的漏洞会引用多个 X_Force 的漏洞条目，而这些被引用的 X_Force 漏洞条目又会引用其他 NVD 的漏洞条目，如果出现这种情况，通过参考链接融合的数据将存在很大的误差。这种情况下，2 个漏洞可能是同一个漏洞，也可能不是，定义为“相关”。3) “无关”：“相同”和“相关”之外的情况定义为“无关”。

3 UVDA 框架的设计与实现

本节主要阐述了 UVDA 框架的整体流程，以及具体模块的实现算法。

3.1 设计思想和目的

UVDA 的设计思路如图 2 所示。首先收集和整理多个漏洞数据库中的数据，根据漏洞的参考链接、厂商和产品型号，去除漏洞库之间重复的漏洞数据，并且以标准化的形式存储到 UVDA 漏洞数据库中，最后以 Web 和 API 接口的方式对普通用户和安全厂商提供服务。用户可以按照厂商、产品型号和时间的形式，统一读取多个库中的数据，进行查询和统计，而不再需要单独地对每个库进行查询。同时，漏洞数据库可以阶段性的模块化的增加，具有良好的扩展性，图 2 中只示意性地给出了 3 个漏洞库，理论上可以增加任意多个。图 2 中虚线内的部分为 UVDA 框架数据库实现的整体流程，虚线外的部分为 UVDA 框架向用户提供服务的示意。

3.2 整体流程和实现

UVDA 框架去重复和标准化的核心思想是将

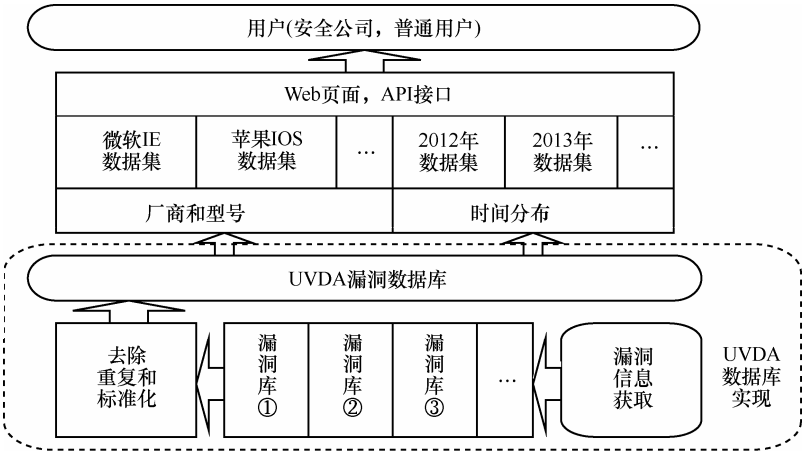


图 2 UVDA 框架的设计思路示意

文本挖掘算法引入到漏洞数据库的数据融合中，从而解决多个库之间的数据冗余和异构问题。UVDA 框架建立了一套算法来统一漏洞库厂商和产品型号的存储形式，并且利用漏洞的参考链接、厂商和型号实现数据库的自动化融合，最终组成一个新的漏洞数据库 UVDA 数据库，并向用户提供服务。假设需要合并的漏洞数据库分别为 I 和 II，数据融合模型的系统架构如图 3 所示。

1) 漏洞数据收集模块。该模块的作用是自动化的获取漏洞数据，该模块具体可以实现如下功能：①可以自动地将目标站点的所有数据或者选定数据保存到本地的数据库中；②具有可扩展性，可以方便地添加新的需要收集的漏洞库站点；③性能稳定，具有断网检测功能和爬虫防御免疫功能，避免获取到错误或者不完整的数据。

2) 漏洞参考链接处理模块。该模块抽取漏洞数据中的参考链接，利用“漏洞字段处理模块”（漏洞字段处理模块将在第 4 节中详细介绍）进行处理，建立参考链接拓扑关系库，按照关系库中的结果将待融合的 2 个漏洞数据库间的漏洞关系分为“相同”、“相关”和“无关”3 类。

3) 漏洞产品型号训练集处理模块。抽取“相同”关系的漏洞作为训练集，提取这些漏洞的产品型号，利用漏洞字段处理模块进行处理，获取训练模型的最优化参数。

4) 漏洞产品型号测试集处理模块。抽取“相关”关系的漏洞作为测试集，提取这些漏洞的产品型号，利用漏洞字段处理模块进行处理，结合训练集产生的最优参数，计算产品型号字符串相关度，建立统一的厂商和产品型号库。

5) UVDA 构建模块。根据漏洞统一性判别规则综合判定各漏洞数据库间漏洞条目的相关情况，消除数据库 I 和 II 间的冗余项，融合各库数据，构建 UVDA 数据库。具体来说，即漏洞相关性判别规则，首先抽取“相同”关系的漏洞，直接合并，然后抽取“相关”关系的漏洞，根据厂商和产品型号是否相同来判断是否进行合并。

6) 提供服务模块。以 Web 页面和 API 的形式提供服务。用户可以按照厂商、产品型号以及时间作为检索条件，同时对多个漏洞库进行统一的检索和统计，而不需要单独统计每个漏洞库。

4　通用漏洞字段处理模块

本节主要阐述了漏洞字段处理模块的实现流程和具体算法。

该模块是漏洞字段相似性算法的实现，是整个 UVDA 框架算法的核心，并且在多处被使用到，由于算法较为复杂，因此单独作为一节进行介绍。该模块是一个通用的标准模块，针对特定的文本性质的漏洞属性字段，可以进行文本挖掘，并且给出漏洞库 I 中的漏洞 *A* 和漏洞库 II 中的漏洞 *B* 的字段 *X*（例如产品型号）的相关性。该模块的实现包括漏洞信息分析、漏洞字段特征提取、漏洞数据处理和漏洞数据存储 4 个阶段。如图 4 所示。

4.1　漏洞信息分析

1) 漏洞字段数据清洗。获取当前漏洞字段后，首先需要规范化每条漏洞字段的字符串，包括书写

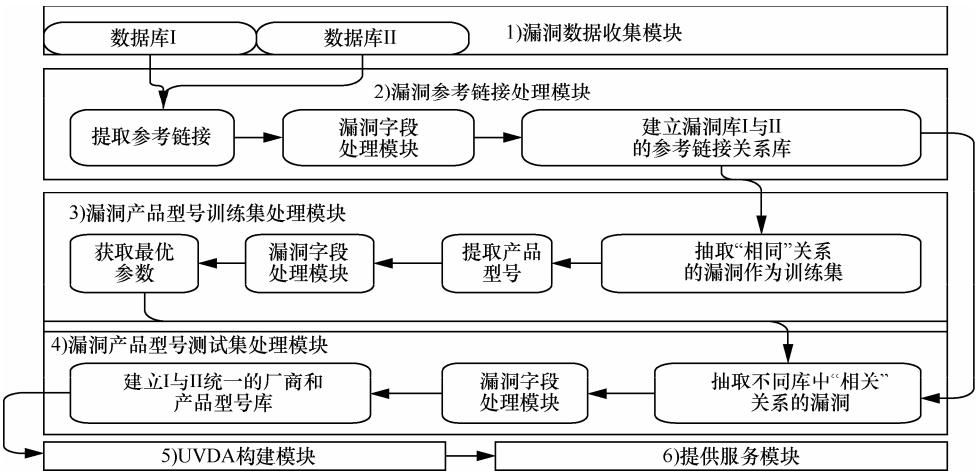


图 3　漏洞数据库建设平台系统架构示意

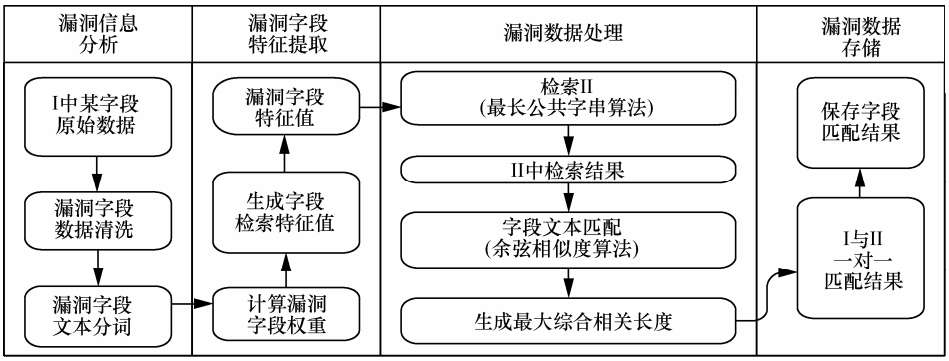


图 4 漏洞属性字段处理模块流程

错误、处理不规则字符、特殊字符和统一大小写等。从而为下阶段提供规则的数据源。

2) 漏洞字段分词。①按照空格分词，将当前要处理的漏洞字段作为一个文本，按照空格分词，如将产品型号 facile interactive web 分为 facile、interactive 和 web，每个被分解的词记为“单词”；②去除停用词，文本中有些功能词对文本的特征没有实质性的影响，如 for、is、to 等，这些词被称为停用词，这些词将被去除；③语法处理，考虑到单词的单复数和时态不同时写法亦不同，需要将单词统一为单数形式，将所有时态统一为一般现在时。

4.2 漏洞字段特征提取

1) 计算漏洞字段权重。权重的计算方法公式为 $T=T_1T_2$，T 表示单词在当前字段中的权重，T_1 表示单词的词频，T_2 表示单词的反向词频。词频的计算方法为 $T_1=\dfrac{tf}{doc\_length}$ tf 表示单词在当前字段单个条目中出现的次数，doc_length 表示当前字段单个条目的长度。反向词频的计算方法为 $T_2=\log\left(\dfrac{N}{df}\right)$，$df$ 表示单词在当前字段所有条目集合中出现的次数，N 表示当前字段的所有条目长度总和。

2) 生成字段检索特征值。按照上一步计算出的单词权重选取检索查全率较高的字符串作为特征值。

4.3 漏洞数据处理

1) 检索 II：利用 I 中生成的特征值在 II 中采用最长公共子字符串算法进行检索，检索结果为下一步字符串的匹配提供数据集 N。该步骤的主要目的在于压缩计算量，因为如果直接在 II 的全集中进行匹配，每条 I 中的字段将与 II 中该字段的数万条数据全部匹配一遍，而 N 通常仅包含数百条数据，计算量几乎压缩 100 倍。

最长公共子字符串算法。该算法属于序列算法的一种，计算 2 个字符串中公共子字符串的最大长度，例如 I 中有产品型号 A = {“facile interactive web”}，II 中有产品型号 B = {“***F@cile*** Interactive Web”}，则 2 个字符串的最大字串为{“cile interactive web”}，长度为 20。

2) 字段文本匹配。按照余弦相似度算法匹配数据集 N 中的漏洞和当前 I 中的漏洞数据。该步骤的目的是从 N 中确定最佳的结果。

余弦相似度算法。利用向量空间中 2 个向量夹角的余弦值作为衡量 2 个个体间差异的大小，注重 2 个向量在方向上的差异，在[0,1]内余弦值越大，说明 2 个向量相似度越大。

3) 生成最大综合相关长度。最大综合相关长度=余弦相似度×最长公共字符串长度，表示 I 中的某个字段条目 A 和 II 中的该字段某个条目 B 的相关性，最大综合相关长度值越大说明相关度越高。例如漏洞库 I 中有产品型号 A = {“facile interactive web”}，漏洞库 II 中有产品型号 B = {“***F@cile*** Interactive Web”}，最大综合相关长度= 0.95 × 20 = 19。

5 环境设置与实验结果分析

本节对 UVDA 框架进行了实现，UVDA 数据库采用了 NVD、X_Force 和 NSFocus 这 3 个具有代表性的漏洞库，其中，NVD 代表了当前国际上最权威的官方漏洞库，X_Force 代表了当前主流非官方漏洞库，NSFocus 代表了当前主流中文漏洞库。并且将构建后的 UVDA 与 3 个单独的漏洞库进行了比较。构建好的 UVDA 漏洞库已经在国家安全漏洞库中得到了应用。

5.1 系统环境设置

操作系统采用 Windows 7 64 bit，开发语言采用

C#（Microsoft Visual Studio 2010），数据存储采用 Microsoft SQL Server 2012 关系数据库。硬件方面，内存要求≥16 GB，硬盘≥2 TB。所有数据公布时间截止到 2015 年 2 月。所有数据由漏洞数据收集模块从互联网抓取。最后将对 3 个数据库和新建数据库的漏洞收录情况进行对比，同时对产品型号数据库的收录情况进行对比。

5.2　UVDA 数据融合过程及结果

漏洞数据库融合过程如图 5 所示，首先将 NVD、X_Force 和 NSFocus 这 3 个漏洞数据库分为 2 组，分别是 NVD+X_Force 和 NVD + NSFocus。对每组数据库的漏洞参考链接进行文本匹配，根据图 3 的步骤，获得同组内 2 个漏洞数据库中相同的漏洞条目和相关的漏洞条目，如表 4 所示，其中，漏洞总数指每个漏洞库所收录的漏洞总量，相同漏洞数表示 2 个漏洞数据库间所指完全相同的漏洞数，相关漏洞数表示所指相关的漏洞数量，相关漏洞存在相同的可能性，但是需要进一步的计算才可以确定，最终合并数表示处理完毕后，确定为相同的漏洞数量。

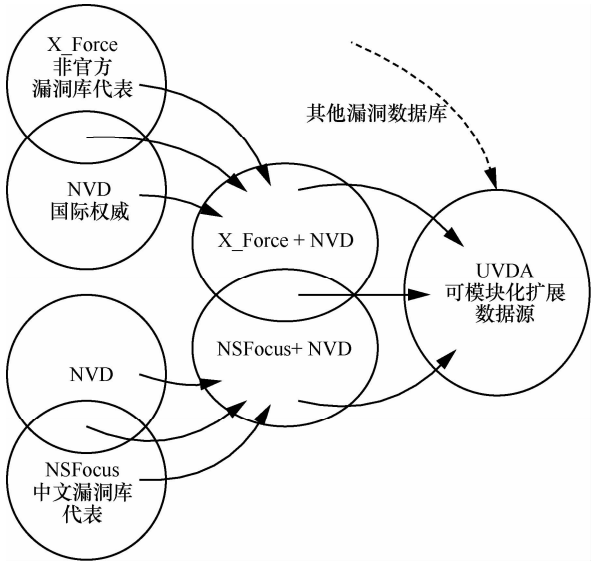


图 5　UVDA 构建示意

表 4　　各漏洞数据库漏洞统计结果

漏洞数量	NVD 关于 X_Force	X_Force 关于 NVD	NVD 关于 NSFocus	NSFocus 关于 NVD
相同漏洞数	28 578	28 129	8 435	8 802
相关漏洞数	67 099	68 287	25 370	21 239
最终合并数	56 726	56 085	15 980	16 675
漏洞总数	68 864	96 367	68 864	29 589

5.3　与其他融合框架的对比

本节将 UVDA 的融合结果与表 2 中的 4 个框架（VSMH[19]、ABVD[25]、UFVD[26]、SVAS[27] 和 UVDA）从适用率、准确率和计算量 3 个维度进行对比。

由于 X_Force 和 NVD 为当前主流的漏洞库，其字段和表现形式具有代表性，因此比较的数据采用 X_Force 关于 NVD 的融合结果。

1) 各个融合框架适用率比较

适用率表示框架可以使用的范围，将表 2 中的“采用”用对应字段的适用率进行替换，得到表 5，最后一列表示每个框架的综合适用率。

表 5　　各个融合框架适用率比较

算法	参考链接	厂商与产品型号	CVSS 指标	漏洞类型	适用率
VSMH	—	100%	69.0%	69.0%	69.0%
ABVD	—	100%	69.0%	—	69.0%
UFVD	100%	—	69.0%	—	69.0%
SVAS	100%	—	—	—	100%
UVDA	100%	100%	—	—	100%

关于参考链接和受影响厂商和产品型号字段，所有的漏洞都会用到这 2 个字段，所以这 2 个字段的适用率为 100%。

关于 CVSS 指标字段和漏洞类型字段，只有采用 SCAP 协议的漏洞才会给出该字段的取值，目前 NVD 完全遵循 SCAP 协议，其他主流漏洞库均部分采用 SCAP，在本文的实验数据中，有 69.0%的漏洞给出了 CVSS 字段值，因此这 2 个字段的适用率为 69.0%。

2) 各个融合框架准确率比较

表 6 中给出了各个融合框架的准确率，以及各个字段的准确率贡献。

表 6　　各个融合框架准确率比较

算法	参考链接	厂商与产品型号	CVSS 指标	漏洞类型	准确率
VSMH	—	85.6%	71.0%	67%	91.6%
ABVD	—	85.6%	71.0%	—	89.2%
UFVD	79.4%	—	71.0%	—	89.2%
SVAS	79.4%	—	—	—	79.4%
UVDA	100%	94.4%	—	—	94.4%

关于参考链接的准确率，根据 UFVD 和 SVAS 的方案，利用本文中的数据得到 79.4%的准确率。本文提出的 UVDA 没有直接利用参考文献，而是首先按照参考链接将 2 个漏洞之间的关系分为了“相同”、“相关”和“无关”3 种关系，“相同”

和“无关”2 种情况的准确率为 100%，剩下的“相关”部分利用受影响厂商和产品型号做进一步的处理。

关于受影响厂商和产品型号的准确率，VSMH 采用通用的字符串余弦相似度算法，ABVD 没有给出具体的算法，因此也按照通用字符串余弦算法进行计算，结果的准确率为 85.6%。本文针对漏洞本文字段的特点设计和实现了“漏洞字段相似度算法”，作为 UVDA 框架的一部分，详见图 4，结合参考链接后，对该字段的准确率为 94.4%。

关于 CVSS 指标字段和漏洞类型字段，由于这 2 个字段的取值较少，因此不能够较好地反映漏洞的特点，准确率也相对较差。

3) 各个融合框架计算量比较

表 7 给出了各个字段的计算复杂度。首先对 X_Force 关于 NVD 的融合结果进行统计，可以得到参考链接有 371 762 种不同的值，厂商与产品型号有 105 352 个不同的值，CVSS 指标为 3^6，即 729 个取值，漏洞类型的分类数取值为 20。

对于 CVSS 指标和漏洞类型的单次操作，属于普通的 C#语句操作，单词计算复杂度记为 1。参考链接的单次操作实际上是一次对于数据库的“select from”查询，由于需要与数据库建立连接，因此单词操作的复杂度记为 100。对于厂商与产品型号的单词计算复杂度，由于每 2 个取值需要执行一次余弦相似度算法（一次余弦相似度算法的计算复杂度记为 100），因此单次操作的复杂度为 $\frac{105\,352\times100}{2}$。

表 7 各个字段计算量分析

指标	参考链接	厂商与产品型号	CVSS 指标	漏洞类型
可取值数	371 762	105 352	729	20
复杂度	100	$\frac{105\,352\times100}{2}$	1	1
数量级	7	11	3	2

UVDA 关于参考链接的计算量与其他框架相同，但是对于厂商与产品型号，UVDA 的单次操作并不是对其他所有的取值分别计算相似度，而是在“相关”关系给定的范围内进行计算。图 6 给出了根据参考链接的漏洞数对应情况，横纵坐标均为对数形式，当横坐标取值大于 1 的时候即为“相关”关系，此时的计算次数为 $\frac{15\,887}{2}$，因此，计算量为 $\frac{15\,887\times100}{2}$。最终各个框架的计算量如表 8 所示，其中，VSMH[19]和 ABVD[25]的计算复杂度数量级为 11，计算量是后 3 种框架的 1 000 倍，因此 VSMH[19]和 ABVD[25]不适合大规模的批量计算。

4) 3 个维度的综合比较

图 7 将适用率、准确率和计算量进行了综合的比较。其中，为了能够直观地进行比较，将计算量做了半定量的处理，即将复杂度数量级为 6 时作为 100%，UVDA 框架的综合性能要优于其他 4 个漏洞融合框架。

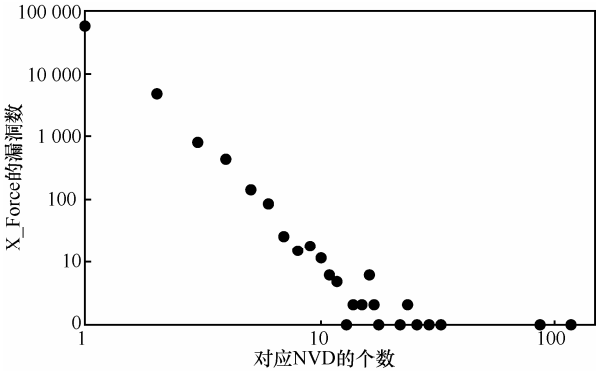


图 6 根据参考链接的漏洞数对应情况

表 8 各个融合框架计算量比较

算法	计算复杂度	复杂度数量级
VSMH	105 352×105 352×50 + 729 + 20	11
ABVD	105 352×105 352×50 + 729	11
UFVD	371 762×100 + 729	7
SVAS	371 762×100	7
UVDA	$371\,762\times100+\frac{15\,887\times100}{2}$	7

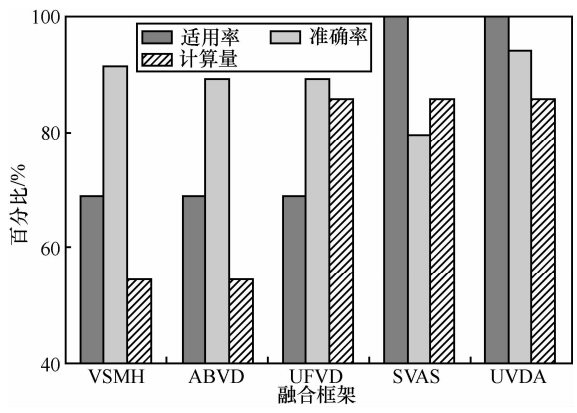


图 7 3 个维度的综合比较

5.4　与其他漏洞数据库的对比

经过 UVDA 框架的处理，最终构建了 X_Force + NVD、NSFocus + NVD、NSFocus + X_Force + NVD 3 组漏洞数据库，分别记为 XF_NV 数据库、NS_NV 数据库和 UVDA，同时生成了 XF_NV、NS_NV 和 UVDA 的受影响厂商产品型号数据库 UVDA_CPE，如图 8 所示。扩充后的漏洞库 UVDA 所包含的漏洞条目分别是 NVD 的 2.28 倍、X_Force 的 1.63 倍、NSFocus 的 5.50 倍，扩充后的漏洞库 UVDA 所包含的受影响厂商产品型号数分别是 NVD 的 2.20 倍、X_Force 的 1.56 倍、NSFocus 的 4.36 倍。

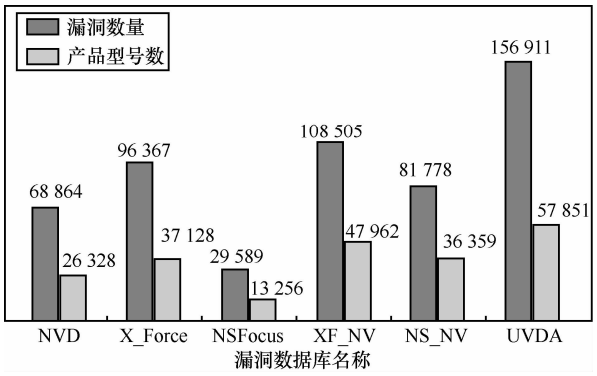


图 8　各漏洞数据库收录漏洞数量及影响产品型号示意

可以看出，经过融合后的漏洞相比于单独的漏洞库在覆盖范围上有了较大的提升。UVDA 的漏洞数据库包括漏洞条目子数据库、产品型号子数据库以及参考链接子数据库。形成 UVDA 的数据处理过程完全按照漏洞字段匹配的程序执行，不需要人工参与，相比于手工操作，提高了执行效率，同时避免了主观因素的影响。

6　结束语

针对目前各安全漏洞库之间存在冗余和异构的问题。本文分析了 15 个漏洞库 84.2 万条漏洞记录的异构情况。基于文本挖掘技术提出了漏洞文本字段相似度算法，可以计算不同漏洞库中产品型号的相似度。在此基础上，本文基于漏洞参考链接和产品型号，提出了一种漏洞同一性判别规则，实验结果表明，正确率为 94.4%，适用率为 100%，均高于其他同类框架。最后，本文基于上述规则提出了漏洞数据库融合框架 UVDA，采用 NVD、X_Force 和 NSFocus 这 3 个有代表性的漏洞库进行了实现，新的漏洞数据库可以按照产品型号和时间统一检索，过程中所有的步骤均可以自动化完成，不需要人工参与。

参考文献：

[1] LIU Q X, ZHANG Y Q. VRSS: a new system for rating and scoring vulnerabilities[J]. Computer Communications, 2011, 34(3): 264-273.

[2] ZHAO D Y, FURNELL S M, AL-AYED A. The research on a patch management system for enterprise vulnerability update[A]. Proc of Information Engineering, WASE International Sonference: 2[C]. Taiyuan,China, 2009. 250-253.

[3] MUNIR R, AWAN I, MUFTI M R. A quantitative measure of the security risk level of enterprise networks[A]. Proc of Broadband and Wireless Computing, Communication and Applications (BWCCA), 2013 Eighth International Conference[C]. Compiegne: IEEE. 2013. 437-442.

[4] WU W, YIP F, YIU E, *et al*. Integrated vulnerability management system for enterprise networks[A]. Proc of e-Technology, e-Commerce and e-Service, The 2005 IEEE International Conference[C]. IEEE, 2005. 698-703.

[5] SHAHZAD M, SHAFIQ M Z, LIU A X. A large scale exploratory analysis of software vulnerability life cycles[A]. Software Engineering (ICSE), 2012 34th International Conference[C]. IEEE, 2012. 771-781.

[6] IBM internet security systems[EB/OL]. http://xforce.iss.net/.

[7] NSFocus [EB/OL]. http://www.nsfocus.net/.

[8] National vulnerability database [EB/OL]. http:// nvd.nist.gov.

[9] 张玉清，吴舒平，刘奇旭，等，国家安全漏洞库的设计与实现[J]. 通信学报，2011, 32(6): 93-100.
ZHANG Y Q, WU S P, LIU Q X, *et al*. Design and implementation of national security vulnerability database[J]. Journal on Communications, 2011, 32(6): 93-100.

[10] OKAMURA H, TOKUZANE M, DOHI T. Security evaluation for software system with vulnerability life cycle and user profiles[A]. Dependable Transportation Systems/Recent Advances in Software Dependability (WDTS-RASD) [C]. IEEE, 2012. 39-44.

[11] LIU Q X, ZHANG Y Q, KONG Y, *et al*. Improving VRSS-based vulnerability prioritization uusing analytic hierarchy process [J]. Journal of Systems and Software, 2012, 85(8): 1699-1708.

[12] WANG L Y K, SUSHIL J, ANOOP S, *et al*. K-zero day safety: a network security metric for measuring the risk of unknown vulnerabilities[J]. IEEE Transactions on Dependable and Secure Computing, 2014, 11(1): 30-44.

[13] GHANI H, LUNA J, KHELIL A. Predictive vulnerability scoring in the context of insufficient information availability[A]. Proc of Risks and Security of Internet and Systems (CRiSIS), 2013 International Conference La Rochelle[C]. IEEE, 2013. 1-8.

[14] ALVI A K, ZULKERNINE M. A natural classification scheme for software security patterns[A]. Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference[C]. IEEE, 2011. 113-120.

[15] VENTER H S, ELOFF J H P, LI Y L. Standardising vulnerability categories [J]. Computers and Security, 2008, 27(3-4): 71-83.

[16] CHEN Z Q, ZHANG Y, CHEN Z R. A categorization framework for common computer vulnerabilities and exposures [J]. The Computer Journal, 2010, 53(5): 551-580.

[17] TRIPATHI A, SINGH U K. On prioritization of vulnerability categories based on CVSS scores[A]. Computer Sciences and Convergence Information Technology (ICCIT), 2011 6th International Conference[C]. IEEE, 2011. 692-697.

[18] ZHENG C, ZHANG Y Q, SUN Y F, *et al*. IVDA: international vulnerability database alliance[A]. Proc of Cybersecurity Summit (WCS), 2011 Second Worldwide[C]. London, UK, 2011. 1-6.

[19] WANG J A, ZHOU L F, GUO M Z, *et al*. Measuring similarity for security vulnerabilities[A]. Proc of System Sciences (HICSS), 2010 43rd Hawaii International Conference[C]. Honolulu, 2010. 1-10.

[20] KOCATEKIN T, ISTANBUL T, UNAY D. Text mining in radiology reports[A]. Proc of Signal Processing and Communications Applications Conference (SIU), 2013 21st[C]. Haspolat, 2013. 1-4.

[21] ABDUL-RAHMAN S, MUTALIB S, KHANAFI N A. Exploring feature selection and support vector machine in text categorization[A]. Proc of Computational Science and Engineering (CSE), 2013 IEEE 16th International Conference[C]. Sydney, NSW, 2013. 1101-1104.

[22] Security content automation protocol [EB/OL]. http://scap.nist.gov/.

[23] Common vulnerabilities and exposures[EB/OL]. http://cve.mitre. org/.

[24] MELL P, SCARFONE K, ROMANOSKY S. Common vulnerability scoring system[J]. Security and Privacy, 2006, 4(16): 85–89.

[25] ARNOLD A D, HYLA B M, ROWE N C. Automatically building an information-security vulnerability database[A]. Proc of Automatically Building an Information-Security Vulnerability Database[C]. West Point, NY, 2006. 376-377.

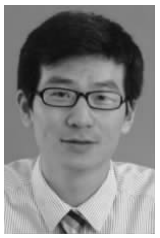
[26] GU Y H, LI PEI. Design and research on vulnerability database[A]. Proc of Information and Computing (ICIC), 2010 Third International Conference: 2[C]. Wuxi, China, 2010. 209-212.

[27] 王晓甜,张玉清. 安全漏洞自动收集系统的设计与实现[J]. 计算机工程, 2006,32(20):177-179.
WANG X T, ZHANG Y Q. Design and implementation of security vulnerability auto-collection system[J]. Computer Engineering, 2006, 32(20):177-179.

作者简介：

温涛（1986-），男，内蒙古巴彦淖尔人，西安电子科技大学博士生，主要研究方向为网络与信息系统安全，包括漏洞评估、漏洞管理、应急响应等。

张玉清（1966-），男，陕西宝鸡人，中国科学院大学教授、博士生导师，主要研究方向为网络与信息系统安全。



刘奇旭（1984-），男，江苏徐州人，中国科学院大学讲师，主要研究方向为网络与信息系统安全，包括漏洞挖掘、漏洞评估、漏洞管理、应急响应等。

杨刚（1991-），男，河北张家口人，中国科学院大学硕士生，主要研究方向为网络与信息系统安全，包括漏洞评估、漏洞管理、应急响应等。