基于改进网络爬虫技术的高性能计算机程序切片级漏洞检测方法

李 妹[1]
Li Shu

摘 要 由于高性能计算机程序通常具有复杂的结构和庞大的代码量，切片级漏洞往往隐藏在这些程序的深处，难以被轻易发现。为此，提出一种基于改进网络爬虫技术的高性能计算机程序切片级漏洞检测方法。将传统的单线程网络爬虫改进为多线程网络爬虫，通过改进后的网络爬虫技术，同时处理多个高性能计算机程序信息的抓取任务，高效爬取计算机程序网页信息。在爬取到的信息中，进行去重处理。根据预设的切片准则，对去重后的信息进行细致切片，提取出可能存在漏洞的计算机程序切片级可疑信息。对可疑信息聚类分析，将相似特征和模式的漏洞切片归为一类，能够准确地识别和检测出高性能计算机程序中的切片级漏洞。实验结果显示，所设计方法在误报率和漏报率方面均控制在 1% 以下，由此证明其在高性能计算机程序切片级漏洞精准检测方面的有效性。

关键词 改进网络爬虫技术；高性能计算机程序；切片级漏洞；多线程网络爬虫；聚类分析

doi：10.3969/j.issn.1672-9528.2024.04.046

0 引言

随着互联网技术的飞速发展，网络安全问题变得越来越严重。传统的漏洞检测方法，如代码审查和静态分析技术，虽然在一定程度上能够检测出程序中的某些常见漏洞，但对于更为复杂的漏洞类型，如格式化字符串漏洞和整数溢出等，其检测效果并不尽如人意。因此，切片级漏洞检测为改进和完善漏洞检测技术提供了新的方法。它有助于提高软件的质量和安全性，保障个人隐私和企业利益，促进漏洞检测技术的进步和创新。

例如，况博裕等人 [1] 提出一种基于人机协同的物联网设备固件漏洞挖掘方案，结合强化学习算法，优化种子变异和模糊测试流程，显著提升漏洞挖掘能力。在处理多个高性能计算机程序信息的挑战上，该方法受限于其单线程工作模式。强化学习算法在训练和更新时无法同时处理多个任务或多个计算机程序信息。因此，难以精确地提取出切片级漏洞的特征，进而影响漏洞挖掘能力和准确性。李妍等人 [2] 提出基于程序过程间语义优化的深度学习漏洞检测方法，应用了包含多头注意力机制的 transformer 神经网络模型执行漏洞检测任务。该方法无法充分捕获和提取复杂程序结构中的各个部分，因而难以精准地提取出切片级漏洞的特征。这可能导致漏洞检测模型的训练数据不够充分，影响漏洞检测的准确性和可靠性。廖雪超等人 [3] 提出基于控制流切片的可编程逻辑控制器代码漏洞检测方法，结合具体漏洞对控制流相关代码进行切片，生成切片控制流图，实现对多层调用 PLC 代码的漏洞检测。该方法所使用的切片控制流图仅限于特定漏洞类型或代码范围。这意味着当遇到其他类型的漏洞或需要处理更广泛的代码范围时，由于解析效率低下和遗漏部分程序信息的可能性，难以精确地提取出切片级漏洞的特征，可能会导致检测结果的准确性受到影响。罗治祥等人 [4] 提出基于污点分析的二进制程序漏洞检测系统，改进了 Java 指针分析中提出的算法，将漏洞模式单独抽离，进行漏洞模式的自定义。基于污点分析技术可以追踪程序中数据流的传播路径，帮助识别潜在的漏洞。但是在处理多个程序信息时，可能难以准确捕获和分析所有污点信息，导致特征提取的不准确性，从而影响切片级漏洞检测的精确性。

鉴于上述方法存在的弊端，为了满足对高性能计算机程序切片级漏洞检测需求，提出基于改进网络爬虫技术的高性能计算机程序切片级漏洞检测方法。利用多线程网络爬虫爬取，能够高效地爬取计算机程序网页信息，并根据切片准则对信息进行切片和提取。通过去重操作，识别出计算机程序切片中存在的漏洞可疑信息，并利用聚类分析的方法对这些信息进行检测和识别。

1. 天津电子信息职业技术学院 天津 300350

1 多线程爬虫并行抓取高性能计算机程序信息

针对高性能计算机程序切片级漏洞的检测，获取程序的运行信息是至关重要的第一步。只有通过对这些信息的深入分析，才能有效识别和检测计算机程序中是否存在切片级漏洞。本文对传统的单线程网络爬虫技术进行了改进，引入多线程技术。改进后的多线程网络爬虫能够同时处理多个请求。通过多个线程并行工作，实现对计算机程序信息的快速抓取，提高爬取效率和精确性。

考虑到信息爬取的效率和精度，此次选择深层网络爬虫。深层网络爬虫在搜寻信息时具有高度的针对性。它从给定的初始统一资源定位符（URL）开始，逐步获取计算机程序页面信息，并持续这一过程直至满足预设的爬虫停止条件。改进网络爬虫主要由控制器、解析器和资源保存库三部分组成。在网络爬行过程中，目标程序网站的入口页 URL 被放入爬取队列中。爬虫从队列头部提取 URL 进行下载，并对所下载的网页信息进行存储、标记识别、信息抽取以及 URL 过滤等操作。处理后的结果 URL 会被插入到爬取队列的末尾，然后爬虫继续从队列中抽取 URL，重复上述步骤，直至完成整个目标程序所有网页的遍历。在爬取过程中由控制器分配爬虫爬取网页的线程，由解析器对计算机程序网页信息下载，最终由资源保存库对下载的计算机程序网页信息保存 [4]。整个爬取过程改进网络爬虫是根据搜索策略，将每个网页看作是一个节点，具体如图 1 所示。

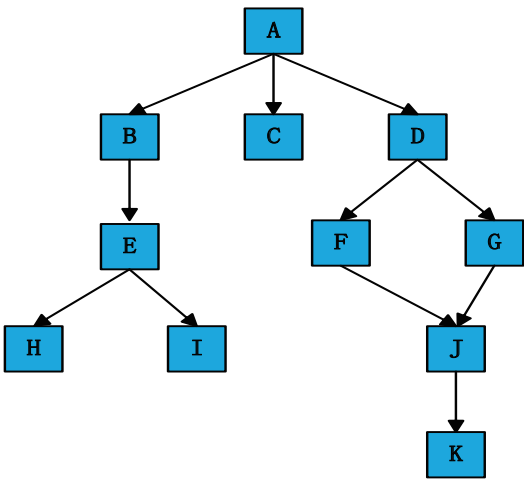

图 1 改进网络爬虫搜索策略图

如图 1 所示，改进网络爬虫以结构相似度为依据，对于结构相似的计算机程序页面，只爬取一个页面的信息。网络爬虫根据深度和节点数将权值平均分配给所有叶子节点，利用递归方式评价两个节点的相似度 [5]。具体步骤为：令深度为 A 的节点权值为 1，对于该节点下方的节点权值为：

$$\varpi = 1/n \tag{1}$$

式中：ϖ 表示根节点下方节点的权值；n 表示根节点下方节点数量 [6]。以此方式得到每个节点的权重。对赋予节点权重后的网络爬虫爬取结构计算相似度，其用公式表示为：

$$sim(T_1,T_2)=\sum_{i=1} E\varpi sim(X_{mn},Y_{mn})Index(X)Index(Y) \tag{2}$$

式中：$sim(T_1,T_2)$ 表示搜索结构 T_1 与 T_2 的相似度；i 表示节点数量；E 表示递归函数；X_{mn}、Y_{mn} 分别表示两个结构中子节点数量；$Index(X)$、$Index(Y)$ 分别表示两个搜索结构第一个子节点的深度 [7]。利用上述公式计算出两个搜索结构的相似度，对于结构相同的结构仅保留一个，以此避免爬虫出现重复爬取。

2 信息去重与细致切片提取可疑漏洞信息

经过基于改进网络爬虫技术的计算机程序信息抓取后，考虑到改进网络爬虫爬取到的网页信息可能存在重复信息，为了保证计算机程序切片级漏洞检测精度，对爬取的计算机程序信息进行去重处理。当两行数据在所有字段上完全一致时，视其为重复数据。在进行去重之前，先创建一个数据副本，以确保在出现任何错误时能够恢复到原始状态 [8]。识别到的重复数据将被自动删除。网络爬虫所抓取的信息主要是计算机程序代码，将这些代码转换为计算机程序代码流图，以清晰地展示源代码的执行顺序，并体现其语法和词法结构。为了检测潜在的切片级恶意代码或异常行为，将生成的代码流图与正常的程序代码流图进行详细的对比分析 [9]。在对比过程中，特别关注与正常程序代码数据控制流图不一致的节点、边等元素。这些不一致之处可能暗示着潜在的问题或恶意行为。为了确保结果的准确性，进一步匹配这些不一致的节点和边到其对应的源代码指令序列 [10]。这一步至关重要，因为它将抽象的图形表示映射回具体的代码层面，提供更精确的可疑数据定位。可疑数据的提取是以分片准则为依据，其用公式表示为：

$$D=(Z,V)sim(T_1,T_2) \tag{3}$$

式中：D 表示可疑切片级漏洞信息切片准则；Z 表示源代码语句编号；V 表示源代码中变量集合 [11]。切片准则可以涵盖各种元素，包括应用程序特定的参数变量、常量，以及表示应用程序接口的语句。为了确保所构建的切片准则具有足够的代表性，选取 20 个具有代表性的计算机程序切片级漏洞源代码样本，以此为基础设计一组全面的切片准则 [12]。准则不仅涵盖了基础行为方面的准则，还特别针对反分析行为设计了相应的切片准则。利用该可疑数据分片准则采用二进制代码长度的滑动窗口对计算机程序源代码遍历，对包含可疑行为的源代码提取。

3 可疑信息聚类分析实现切片机漏洞精确检测

在对计算机程序信息进行去重及可疑信息切片后，采用聚类分析法来判断提取出的可疑切片级代码是否为计算机程序中的切片级漏洞代码。利用聚类分析技术，建立一个切片级漏洞因子库，并将提取到的可疑代码与因子库中切片级漏洞因子进行聚类。通过聚类分析，能够将相似特征和模式的可疑代码归为同一类别。如果某个可疑代码与切片级漏洞因子的特征相似，并且聚类结果表明其属于切片级漏洞因子所在的类别，那么可以判断该可疑代码是一个切片级漏洞。将切片级漏洞拆分为闭合类因子、注释类因子和逻辑类因子，其用公式表示为：

$$D_{CPS}=D\left(C_{LOSE},L_{GC},N_{OTE}\right) \tag{4}$$

式中：D_{CPS} 表示计算机程序切片级漏洞攻击向量；C_{LOSE} 表示闭合类因子，即通过双引号或者单引号将当前的程序字段参数闭合；L_{GC} 表示注释类因子，即通过“#”等符号注释将当前程序字段参数之后的部分进行注释，使代码不能正确有效地执行；N_{OTE} 表示逻辑类因子，即通过“or”逻辑关键字将程序逻辑部分构成恒等式。由三个因子构建计算机程序切片级漏洞因子库，通过聚类计算出提取到的可疑代码与因子库中相似度，其计算公式为：

$$\mu=u\left(P_{sub}\cup K_{sub}\right)D_{CPS} \tag{5}$$

式中：μ 表示可疑代码与因子库相似度；u 表示条件概率；P_{sub} 表示置信度；K_{sub} 表示支持度。相似度越高，则表示可疑代码与计算机程序切片级漏洞因子库中因子越接近，计算机程序存在漏洞的可能性越大。设定一个阈值，如果 μ 值大于阈值，则判定检测样本为计算机程序切片级漏洞，以此实现高性能计算机程序切片级漏洞检测。

4 实验论证

4.1 实验准备与设计

本文根据所设计的高性能计算机程序切片级漏洞检测方法，对目标程序进行切片级漏洞检测，并对最终的检测结果进行深入分析，以验证该方法的有效性和可行性。实验环境为 Node.js 平台，使用 Java 语言进行开发。实验环境的硬件配置为 Intel（R）Core i8-3660 CPU，内存为 16 GB，主频为 2.05 GHz，配备 1 T 硬盘，确保实验的高性能和稳定性。本次实验选择的检测站点为 IFAAHF 测试站点。为了充分测试漏洞检测方法的性能，在该网站上设置了 200 个切片级漏洞，并设置切片级漏洞检测的目标程序的入口地址，然后开始执行程序。在执行计算机程序过程中将改进网络爬虫爬取信息显示到控制台中，实验中采用改进网络爬虫技术共爬取 10 000 个程序信息样本，将检测结果上传到服务器的文本文件中保存。

4.2 实验结果与讨论

4.2.1 数据去重率结果分析

为了验证本文方法的有效性，在实验中进行了数据去重率的测试。较高的数据去重率对于减少重复信息、提高漏洞检测的精确性和效率至关重要。具体的实验结果如图 2 所示。通过这一测试，期望能够评估本文方法在去重方面的表现，并为其在实际应用中的效果提供有力依据。

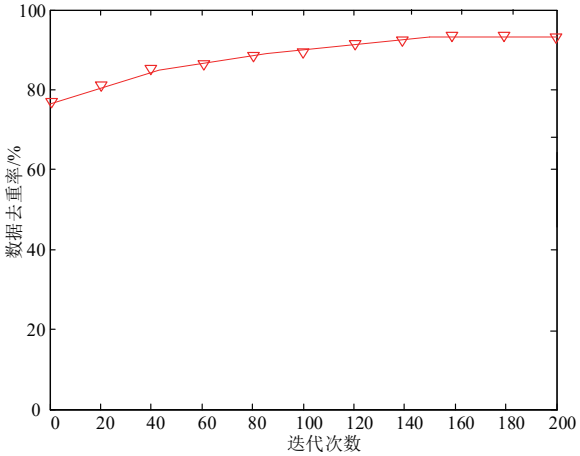

图 2 数据去重率结果

由图 2 可知，本文方法在数据去重方面取得了较高的成功率，最高达到 92%。这意味着本文方法在数据去重方面取得了良好的效果，成功减少了重复信息的存在。这一结果对于漏洞检测至关重要，减少了重复信息的存在，能够有效提高漏洞检测的精确性和效率，并为后续切片级漏洞检测提供可靠的数据基础。

4.2.2 对比结果分析

为了全面评价高性能计算机程序切片级漏洞检测的精度性能，本次实验将从误报率和漏报率两个核心指标进行考量。误报率是指错误检测样本占总检测样本的比例，它反映了在切片级漏洞检测中，那些并未真正构成威胁的代码片段被误判为有漏洞的情况。而漏报率是指未能检测到的切片级漏洞样本占总检测样本的比例，它揭示了实际存在、可能对系统安全造成影响的漏洞，因检测方法的不足或误判而未被正确识别的问题。误报率和漏报率的高低直接反映了检测方法的精确性，二者数值越高，意味着检测精度越低。

为了使实验数据更具说明性和对比性，本次实验将本文提出的切片级漏洞检测方法，与文献 [1] 中提出的基于人机协同的物联网设备固件漏洞挖掘方案，以及文献 [2] 中提出的基于程序过程间语义优化的深度学习漏洞检测方法进行了对比分析。实验结果如表 1 和表 2 所示。

表 1 高性能计算机程序切片级漏洞误报率 /%

程序总数	本文方法	文献 [1] 方法	文献 [2] 方法
1000	0.16	5.62	10.26
2000	0.18	6.35	11.24
3000	0.19	5.95	10.36
4000	0.21	5.75	10.25
5000	0.19	5.46	10.42
6000	0.17	5.62	10.25
7000	0.16	5.42	10.35
8000	0.15	5.62	10.45
9000	0.16	5.41	10.46
10 000	0.18	5.36	10.75

表 2 高性能计算机程序切片级漏洞漏报率 /%

程序总数	本文方法	文献 [1] 方法	文献 [2] 方法
1000	0.08	10.26	5.62
2000	0.16	10.36	6.25
3000	0.18	10.45	5.74
4000	0.16	10.35	5.46
5000	0.15	10.25	5.06
6000	0.17	10.22	5.26
7000	0.21	10.62	5.42
8000	0.16	10.86	5.93
9000	0.28	10.75	5.16
10 000	0.26	10.85	5.35

通过对比表 1 中的数据，可以看出本文方法在高性能计算机程序切片级漏洞检测中的误报率平均仅为 0.18%。这一数值相较于文献 [1] 方法降低了近 5.2%，与文献 [2] 方法相比更是降低了近 10.5%。同样，从表 2 的数据中，可以发现本文方法在高性能计算机程序切片级漏洞检测中的漏报率平均为 0.26%，与文献 [1] 方法相比，本文方法的漏报率降低了近 10.6%，而与文献 [2] 方法相比，则降低了近 5.1%。这些对比数据充分证明了本文方法在高性能计算机程序切片级漏洞检测方面具有显著优势。本文方法将传统的单线程网络爬虫改进为多线程网络爬虫，实现了同时处理多个信息抓取任务，高效地爬取计算机程序网页信息。去重处理确保了后续分析的准确性和高效性。根据预设的切片准则进行细致切片，定位可能存在漏洞的计算机程序切片级信息。通过提取可疑信息，集中分析潜在漏洞区域，以提高检测准确性。聚类分析将具有相似特征和模式的漏洞切片归为一类，揭示漏洞共性和规律，准确识别和检测高性能计算机程序中的切片级漏洞。这些关键步骤共同作用，使得本文方法不仅能够大幅减少误报和漏报的情况，提高检测的准确性，还能为实际应用提供更加可靠的安全保障。

5 结语

为了提高切片级漏洞检测的准确性和效率，本研究采用更先进的网络爬虫技术，提出了一种新的检测思路，即利用改进的网络爬虫技术来检测高性能计算机程序中的切片级漏洞。通过多线程爬虫去除重复信息，按切片准则提取可疑信息，并通过聚类分析成功识别了漏洞。实验结果显示，该方法的误报率和漏报率均低于 1%，证明了其精准性。然而，尽管这项研究取得了一定的成果，但仍有许多值得深入研究和探索的方向。随着技术的不断发展和挑战的出现，相信未来的研究将为该领域带来更多的创新和突破。

参考文献：

[1] 况博裕 , 张兆博 , 杨善权 , 等 .HMFuzzer：一种基于人机协同的物联网设备固件漏洞挖掘方案 [J]. 计算机学报 , 2024, 3: 1-14.

[2] 李妍 , 羌卫中 , 李珍 , 等 . 基于程序过程间语义优化的深度学习漏洞检测方法 [J]. 网络与信息安全学报 ,2023,9(6):86-101.

[3] 廖雪超 , 孟航宇 . 基于控制流切片的可编程逻辑控制器代码漏洞检测方法 [J/OL]. 计算机集成制造系统 ,1-13[2024-04-07].https://doi.org/10.13196/j.cims.2023.0428.

[4] 罗治祥 , 向栖 , 李乐言 . 基于污点分析的二进制程序漏洞检测系统设计与实现 [J]. 网络安全与数据治理 , 2023, 42(11): 1-7.

[5] 江姝晨 , 牛保宁 , 高彦 . 基于混合语义的切片级智能合约重入漏洞检测 [J/OL]. 计算机工程与应用 ,1-9[2024-04-07].http://kns.cnki.net/kcms/detail/11.2127.TP.20231012.1547.008.html.

[6] 邹德清 , 李响 , 黄敏桓 , 等 . 基于图结构源代码切片的智能化漏洞检测系统 [J]. 网络与信息安全学报 ,2021,7(5):113-122.

[7] 王旭阳 , 秦玉海 , 任思远 . 基于机器学习的 Android 混合应用代码注入攻击漏洞检测 [J]. 信息安全研究 , 2023, 9(10): 940-946.

[8] 庄园 , 曹文芳 , 孙国凯 , 等 . 基于生成对抗网络与变异策略结合的网络协议漏洞挖掘方法 [J]. 计算机科学 , 2023, 50(9): 44-51.

[9] 白英民 , 师智斌 , 信文阁 , 等 . 基于词嵌入与 Shapelet 时序特征的智能合约漏洞检测方法研究 [J]. 中北大学学报 (自然科学版), 2023,44(4):381-387.

[10] 胡雨涛 , 王溯远 , 吴月明 , 等 . 基于图神经网络的切片级漏洞检测及解释方法 [J]. 软件学报 ,2023,34(6):2543-2561.

[11] 程亚维 , 王东霞 . 基于网络爬虫技术的网页 SQL 注入漏洞检测方法 [J]. 信息与电脑 (理论版),2023,35(4):236-238.

[12] 王剑 , 匡洪宇 , 李瑞林 , 等 . 基于 CNN-GAP 可解释性模型的软件源码漏洞检测方法 [J]. 电子与信息学报 , 2022, 44(7): 2568-2575.

【作者简介】

李姝（1987—），女，天津人，本科，工程师，研究方向：计算机软件工程。

（收稿日期：2024-02-22）