HMFuzzer:一种基于人机协同的物联网设备固件漏洞挖掘方案

况博裕[1),2)] 张兆博[2)] 杨善权[2)] 苏 铓[2)] 付安民[1),2)]

[1)](南京理工大学网络空间安全学院 南京 210094)

[2)](南京理工大学计算机科学与工程学院 南京 210094)

摘 要 模糊测试是一种针对物联网设备固件漏洞挖掘的主流方法,能够先攻击者一步发现安全威胁,提升物联网设备的安全性.但是目前大部分的模糊测试技术关注于如何自动化地实现漏洞挖掘,忽略了专家经验对于设备固件漏洞挖掘工作的优势.本文提出一种基于人机协同的物联网设备固件漏洞挖掘方案 HMFuzzer,设计了基于设备固件前后端交互的设备固件关键信息提取方法,通过模拟设备固件、设备管理界面以及用户三方交互模式获取固件潜在的关键信息,并通过二进制文件定位和函数分析技术解析出固件关键函数.此外,HMFuzzer 通过在模糊测试的预处理、测试和结果分析阶段引入专家经验,利用上一阶段获取的关键信息,结合强化学习算法,优化种子变异和模糊测试流程,显著提升了模糊测试的覆盖率、效率以及漏洞挖掘能力.实验结果表明,相比于现有的固件漏洞挖掘方法,HMFuzzer 的漏洞识别成功率能提高 10%以上,具备更强的漏洞检测能力.特别是,针对真实厂商的物联网设备测试,HMFuzzer 发现了多个 0-day 漏洞,其中已获得 4 个 CVE/CNVD 高危漏洞.

关键词 物联网;漏洞挖掘;模糊测试;人机协同;设备固件;强化学习

中图法分类号 TP309 **DOI 号** 10.11897/SP.J.1016.2024.00703

HMFuzzer: A Human-Machine Collaboration-Based Firmware Vulnerability Mining Scheme for IoT Devices

KUANG Bo-Yu[1),2)] ZHANG Zhao-Bo[2)] YANG Shan-Quan[2)] SU Mang[2)] FU An-Min[1),2)]

[1)](*School of Cyber Science and Engineering, Nanjing University of Science and Technology, Nanjing* 210094)

[2)](*School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing* 210094)

Abstract The popularity of infrastructure such as 5G has greatly facilitated the development of the Internet of Things (IoT), which has become an integral part of our lives. However, with the widespread adoption of IoT technologies in many areas, the security risks of its architecture have also increased and the attack surface for IoT is becoming more diverse. In recent years, malicious attacks and security incidents related to the IoT have been frequent, which are often caused by exploitable security vulnerabilities in IoT devices. In this context, fuzzing currently becomes the mainstream approach for IoT device firmware vulnerability mining, which can improve the security of IoT devices by detecting security threats ahead of attackers. However, most of the existing research works on IoT device firmware vulnerability mining have overly focused on automated vulnerability mining. Although automated vulnerability mining reduces the labor cost, it limits the flexibility and scalability of the solutions and ignores the benefits of expert experience. Expert

收稿日期:2023-06-30;在线发布日期:2024-01-02.本课题得到国家自然科学基金项目(62072239,62372236)、江苏省自然科学基金项目(BK20211192)、未来网络科研基金项目(FNSRFP-2021-ZD-05)、中央高校基本科研业务费专项资金(30921013111)、江苏省青蓝工程和江苏省卓越博士后计划资助.**况博裕**,博士,博士后研究员,主要研究领域为物联网安全、漏洞挖掘. E-mail: kuang@njust.edu.cn. **张兆博**,硕士,主要研究方向为物联网安全、漏洞挖掘. **杨善权**,博士研究生,主要研究方向为物联网安全、漏洞挖掘. **苏 铓**,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为云安全、访问控制与权限管理. **付安民**(通信作者),博士,教授,博士生导师,中国计算机学会(CCF)高级会员,主要研究领域为密码学、隐私保护. E-mail: fam_0522@163.com.

experience can greatly enhance the compatibility of automated fuzzers, improve the efficiency of test seed evolution, and also help automated tools to discern anomalous program states that are difficult to resolve, which in turn improves the ability of vulnerability mining. Therefore, in order to effectively combine expert experience and automated vulnerability mining and improve the efficiency of IoT device firmware vulnerability mining, this paper proposes a human-machine collaborative firmware vulnerability mining scheme for IoT device, named HMFuzzer. HMFuzzer obtains the target device firmware information through various methods and designs a device firmware key information extraction method based on the interaction between the front and back ends of the device firmware. The method simulates the three-party interaction mode among the device firmware, the management interface, and the user, which can effectively obtain the target device's potential firmware key information. Besides, HMFuzzer obtains firmware key functions through binary file location and function analysis for guiding the subsequent fuzzing seed generation. In addition, HMFuzzer introduces expert experience into the pre-processing, testing, and result analysis phases, optimizing the seed variation and fuzzing process based on the key information obtained in the previous phase. Furthermore, it combines with the reinforcement learning algorithm, which significantly improves the coverage rate and efficiency of fuzzing as well as the vulnerability mining capability of target device firmware. To verify the effectiveness and detection capability of HMFuzzer, we conduct three parts of experiments, including reinforcement learning algorithm experiments, vulnerability detection capability comparison experiments, and 0-day vulnerability mining experiments. The reinforcement learning algorithm experiments demonstrate that compared to existing vulnerability mining solutions for IoT devices, the reinforcement learning algorithm is able to significantly improve the coverage of target programs by improving the efficiency of seed screening. The vulnerability detection capability comparison experiments collect 110 vulnerabilities in 37 different devices from 7 vendors. The results show that HMFuzzer is able to improve vulnerability identification success rate by more than 10% compared to existing methods, providing a stronger vulnerability detection capability. The 0-day vulnerability mining experiments analyze three real vendor's IoT devices. Finally, HMFuzzer finds multiple 0-day vulnerabilities, including 4 CVE/CNVD high-risk vulnerabilities.

Keywords Internet of Things; vulnerability mining; fuzzing; human machine collaboration; device firmware; reinforcement learning

1 引 言

5G等基础设施的普及极大促进物联网(Internet of Things,IoT)的发展,物联网已成为我们生活中不可或缺的一部分[1].然而随着物联网技术在关键基础设施、工业部门和智能家居等多个领域的广泛应用,其体系架构的安全风险也随之增加,面向物联网的攻击面也越来越多.近年来,有关物联网的恶意攻击与安全事件频发[2],2010 年位于伊朗的上千台核设施离心机遭受"震网"病毒大面积感染,直接导致伊朗原定的核计划倒退两年;2020 年特斯拉、波音、SpaceX 的零件供应商 Visser Precision 遭勒索软件攻击,大量敏感数据遭到泄漏;同年 4 月,以色列供水部门工控设施遭到网络攻击;2022 年立陶宛能源公司 Ignitis Group 遭受了十年来最大的网络攻击,大量分布式拒绝服务(Distributed Denial of Service,DDoS)攻击破坏了其数字服务和网站.而这些攻击的起因,往往是由于系统中的物联网设备存在可利用的安全漏洞,进而导致了整个物联网系统遭受攻击[3-4].

面对逐年递增且愈发严峻的攻击形式,国内外安全研究人员提出通过面向物联网设备的漏洞挖掘方法[5-6],结合已有的安全策略对潜在的安全漏洞进行检测,分析目标设备未知的安全风险,先攻击者一步发现物联网中的安全威胁,及时采取相应的安全

保障措施维护系统的安全稳定运行，进而将安全风险最小化.

目前的设备漏洞挖掘研究更多关注于物联网设备中 0-day 漏洞的挖掘，利用模糊测试的方法自动化挖掘物联网设备固件中存在的漏洞. IoTFuzzer[7] 提出了一种自动模糊化框架，通过识别和重用特定于程序的逻辑来变异测试用例，旨在在没有访问其固件镜像的情况下发现物联网设备中的内存损坏漏洞. SRFuzzer[8] 利用两个输入语义模型——KEY-VALUE 数据模型和 CONF-READ 通信模型，持续有效地生成测试用例，并协调多样化的变异规则与多种监测机制，以触发多种类型的漏洞. FirmFuzz[9] 提出了自动化设备无关仿真和动态分析框架，通过基于灰盒的模糊测试，利用静态分析和系统内省，在固件分析中提供有针对性和确定性的检测. Firm-AFL[10] 首次提出了用于物联网固件的高吞吐量的灰盒模糊测试方案，利用 POSIX 兼容固件的模糊化以解决兼容性问题，并采用增强过程仿真的技术解决了系统模式仿真造成的性能瓶颈. DIANE[11] 结合静态和动态分析方法，在 Android 配套应用中找到模糊测试中的触发位置，然后实现自动对物联网设备进行模糊测试. Feng 等人[12] 提出了设备仿真框架 P^2IM，能够连续执行给定的固件二进制文件，并抽象各种外围设备，基于自动生成的模型实时处理固件 I/O，通过引导来自模糊测试的输入，从而实现硬件独立和可扩展的固件测试. KARONTE[13] 利用通信建模的方法，跟踪多二进制文件之间的交互，从而分析嵌入式设备固件. 但是 KARONTE 更为关注跨后端二进制文件的分析，忽略了前端的信息，导致其存在漏报的问题. SaTC[14] 提出一种新型污点分析方法，该方法用于检测前后端二进制文件之间共享的字符串关键字，跟踪前后端之间用户输入的数据流，以此检测潜在的安全隐患. FuzzWare[15] 提出了一种轻量级程序分析、重新托管和模糊测试相结合的方法，将模糊测试集中于重要输入的变异，从而提高模糊测试的有效性. IoT-Scope[16] 针对设备的隐藏网络接口，通过固件分析构建探测请求以测试物理设备，利用差分分析过滤掉不相关的请求和接口来缩小识别范围.

固件漏洞挖掘工作的难点在于其依赖于真实或者仿真设备的运行和状态监督，而且大多需要针对二进制文件进行解析. 然而，现有的大部分物联网设备固件漏洞挖掘的研究方案过度关注于提升自动化模糊测试的性能，自动化模糊测试虽然降低了人工成本，但是限制了方案的灵活性和拓展性，主要存在 3 方面的缺陷：(1) 由于部分设备信息难以通过自动化的方式完整提取，导致后续测试过程不准确，甚至不可行；(2) 一些质量较低的种子会导致模糊测试器花费大量时间；(3) 自动化模糊测试器会误判一些难以分辨的程序异常状态，进而产生漏报与误报. 相比之下，专家经验能够极大增强自动化工具对部分难以兼容的固件的适配能力，提高测试种子进化的效率，还可以帮助自动化工具分辨难以解析的程序异常状态，进而提升漏洞挖掘的能力.

因此，为了能够有效地将专家经验和自动化漏洞挖掘工作结合，提高设备固件漏洞挖掘的效率，本文提出了一种基于人机协同的设备固件漏洞挖掘方案 HMFuzzer. 它结合设备管理界面信息和用户模拟交互信息分析目标设备固件，提取其中的关键信息，以指导后续的模糊测试种子生成. 在预处理阶段、测试阶段和结果分析阶段引入专家经验，利用强化学习算法优化模糊测试种子变异过程，显著提升了模糊测试覆盖率、效率以及漏洞挖掘能力. 本文工作的主要贡献如下：

(1) 提出了一种基于设备固件前后端交互的设备固件关键信息提取方法，该方法模拟设备固件、管理前端以及用户三方交互模式，能够有效地获取目标设备关键信息.

(2) 提出了一种基于人机协同的设备固件漏洞挖掘方案 HMFuzzer，利用之前获得的关键信息，在测试的全流程引入专家经验，结合强化学习算法，优化种子变异和模糊测试流程，进而提高模糊测试的覆盖率和效率.

(3) 对比实验表明，相比于现有的漏洞挖掘方法，HMFuzzer 的漏洞识别成功率能提高 10%以上，具备更强的漏洞检测能力. 特别是，针对真实的物联网设备，HMFuzzer 发现了多个 0-day 漏洞，其中已获得 4 个 CVE/CNVD 高危漏洞.

本文第 2 节介绍相关的漏洞挖掘方法；第 3 节从问题定义、方案流程、关键参数设定和人机协同四个方面详细阐述了本文的方案；第 4 节给出 HMFuzzer 在强化学习模糊测试、漏洞检测能力和 0-day 漏洞挖掘的实验对比与分析；第 5 节总结全文.

2　相关工作

漏洞挖掘是一种发现潜在威胁的手段，包括了静态分析、代码比对、模糊测试等多种方法. 其中，模

糊测试是目前最流行的漏洞挖掘技术之一,通过向目标应用程序生成大量正常和异常输入,然后将生成的输入反馈给目标应用程序并监视执行状态来检测异常.与其他技术相比,模糊测试易于部署,且具有良好的可扩展性和适用性,在有/无源代码的场景下均可实现[17].本文所涉及的漏洞挖掘技术包括以下两个方面:物联网固件模糊测试漏洞挖掘和基于机器学习的漏洞挖掘,如图 1 所示.

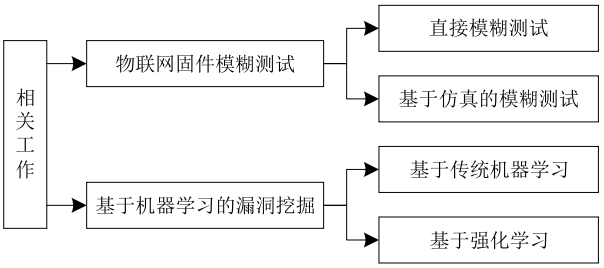

图 1 相关技术结构图

2.1 物联网固件模糊测试漏洞挖掘

利用模糊测试实现自动化漏洞挖掘研究成为了近几年国内外安全研究中应用最为广泛的方法之一,模糊测试根据测试期间所需的关于待测目标程序的信息获取量可以分为黑盒、白盒与灰盒三类.黑盒模糊测试是指对于待测程序只了解输入和输出特性,不关心程序内部实现细节的一种测试方法.白盒模糊测是指在测试过程中获得程序内部的信息,包括代码、变量、路径等,从而对程序进行更加深入的测试.灰盒模糊测试则是综合了黑盒和白盒两种测试方法的优点,既考虑了程序的外部行为特征,又能够检查程序的内部逻辑和数据结构,因此近几年的研究多采用灰盒的方法对待测目标设备进行测试[18].

随着软件模糊测试的发展,研究者们也将模糊测试方法应用于固件漏洞挖掘领域.但是固件模糊测试与传统的软件模糊测试存在巨大差别,主要体现在两个方面:(1)固件模糊测试的执行过程依赖于真实设备或者仿真程序的支持;(2)固件模糊测试往往针对二进制程序展开,难以直接获取函数跳转等程序内部信息.根据对设备固件的处理方式,固件漏洞模糊测试可以分为直接模糊测试和基于仿真的模糊测试两种方法.

直接模糊测试也可以被称为基于真实设备的模糊测试[19],即通过被动或主动的方式连接目标设备进行测试.被动方式指连接设备开放的网络服务或蓝牙等端口进行黑盒模糊测试,但这个方法只能测试设备固件中的特定应用程序.而主动方式则需要手动连接设备的调试接口,通过 UART 或 JTAG 等接口[20]获取设备后台 shell,并对设备的操作系统和程序进行测试.

基于仿真的模糊测试依赖于可获得的固件包,然后在对设备仿真的基础上进行测试.固件包可以通过调试接口/Flash 芯片进行提取,或通过固件更新网址等直接从设备官网下载.基于仿真的模糊测试可以进一步分为全系统模式仿真、混合仿真和用户模式仿真.全系统模式仿真模拟所有的硬件,能够执行整个固件的操作系统和应用程序,缺点是十分耗费资源[10,21].混合仿真只模拟部分硬件,将外围 API 请求转发给实际设备[22-24],提高了仿真的准确性.用户模式仿真则在保留系统模式仿真器内核的情况下,用主机系统运行用户级程序[10],因此能够实现最大的吞吐量.

2.2 基于机器学习的漏洞挖掘

人工智能作为一种先进高效的技术手段,也逐渐被应用于漏洞挖掘领域[25-26].近年来,国内外安全研究人员开始将机器学习作为优化传统漏洞挖掘方法的一种途径,以解决误报率高、重复性工作多等问题,提高漏洞挖掘的准确性和效率.随着固件代码量逐渐增多、漏洞数据不断增加,基于机器学习的漏洞挖掘方法也愈发主流[27].Genius[28]提出了一种基于机器学习的方法来生成鲁棒的平台无关函数特征向量,两个函数的相似性被转换为两个函数特征向量之间的距离.VulSeeker[29]提出了一种基于语义学习的二进制漏洞查找方法,克服了不同平台的限制,该方法根据给定目标函数和脆弱函数,提取基本块特征作为向量,基于余弦距离测量两个函数的相似性,以此来实现漏洞挖掘.VulSeeker-pro[30]在 VulSeeker 方法的基础上,提出了一种增强的二进制漏洞查找方法,将函数语义分析集成在语义学习的后端,能够有效地缓解安全分析人员的手动识别工作量,提高漏洞检测的效率.IoTSeeker[31]提出基于函数语义学习的跨平台二进制漏洞搜索方法,通过计算两个嵌入向量的余弦距离,确定二进制函数是否包含已知漏洞.Asm2Vec[32]将二进制代码转换为向量表示,利用 Word2Vec 模型和基于窗口的上下文学习方法,将汇编指令转换为向量表示.DeepBinDiff[33]提出了一种无监督的程序范围代码表示学习技术,结合代码语义信息和控制流信息来生成基本块嵌入,提升方案的准确度和可扩展性.

此外,随着研究的不断深入,强化学习作为机器学习的分支之一,成为漏洞挖掘方法优化的重要趋

势. 强化学习通过智能体与环境的状态、执行动作和获取奖励等方式进行交互，利用在交互过程中学习到的策略，使得其能够获得更多的奖励. 强化学习一方面能够很好地平衡漏洞挖掘中存在的探索与利用问题，另一方面能够避免出现重复性的工作，降低检测过程的复杂度与随机性.

目前强化学习在漏洞挖掘中的应用还比较少[34-36]，AFLFast[34]提出了在模糊测试过程中偏向探索更低频路径的思想，利用马尔科夫链模型量化了种子输入与执行路径之间的概率关系. 考虑到模糊测试过程中种子的总数会不断增加导致种子的奖励期望不断降低的问题，EcoFuzz[35]提出了对抗性多臂老虎机变体(VAMAB)模型. 类似的，AFLHIER[36]也将路径转换表述为一个多臂老虎机问题，提出了分级调度策略，将待测目标的函数、边、基本块等作为分级指标用于种子聚类，并结合不同级别的多个度量对种子进行评估. 然而，现有的基于强化学习的漏洞挖掘方案大多针对开源软件进行漏洞检测，以能够直接运行的 C/C++文件为检测目标，而物联网设备固件漏洞挖掘依赖于真实/仿真设备的执行，且通常以二进制文件为检测目标，因此现有的基于强化学习的漏洞挖掘方案难以实现对物联网设备固件的漏洞挖掘[26]；而且这些方案都采用了自动化漏洞挖掘方法，摈弃了专家经验对于漏洞挖掘能力的优势.

3　本文方法

3.1　问题定义

虽然现有的针对物联网设备固件的安全分析研究已经提出了部分的工作，但是仍然存在很多问题以及挑战需要去解决. 本文利用人机协同的方法设计了一个设备固件漏洞挖掘方案，主要针对以下 3 方面的问题.

(1) 专家知识与自动化漏洞挖掘方法结合问题. 传统的人工分析的方式虽然能够深入分析物联网设备固件中存在的漏洞，但是效率低下且开销巨大，而自动化漏洞挖掘方法在漏洞挖掘能力上又有所欠缺. 因此，利用专家知识来辅助增强自动化漏洞挖掘方法是一种新的途径，但是如何使它们有效结合、相互理解仍是目前的关键问题.

(2) 物联网设备关联分析问题. 大部分物联网设备固件安全分析方法更关注于对目标设备本身的安全性分析，但是忽略了设备在整个物联网体系中与其他应用及设备的交互问题，目标设备与其相关联的其他组件或应用的交互同样会造成潜在的漏洞，难以被固件安全工具发现.

(3) 覆盖率与开销的平衡问题. 传统的模糊测试过程通常是通过随机生成一些测试用例来测试目标设备，从而发现潜在的漏洞和缺陷. 但是这种方法存在一些不足，例如测试用例的生成效率低、覆盖率不足等. 现有的方法常结合其他的技术，如污点分析、符号执行、机器学习等技术，但是在开销与覆盖率的平衡问题上并没有得到较好的效果.

3.2　方案流程

针对上述问题，本文提出了基于人机协同的设备固件漏洞挖掘方案 HMFuzzer，具体流程如图 2 所示. HMFuzzer 的设备固件分析方法主要可以分为两大阶段：基于模拟交互的多层级关键信息提取阶段和基于前后端人机协同的固件分析阶段.

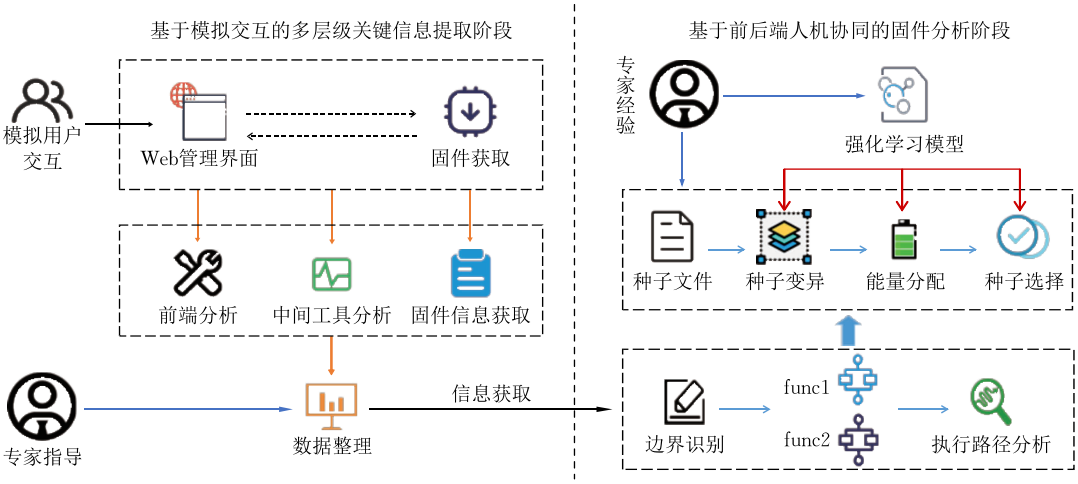

图 2　基于人机协同的设备固件漏洞挖掘流程

基于模拟交互的多层级关键信息提取阶段. 该阶段基于用户、设备以及管理页面三方交互的模式，获取细粒度的目标设备信息并加以整合. 具体而言，HMFuzzer 方案通过官网下载、第三方网站获取、设备 Flash 芯片读取等方法获取目标设备固件，通过 Binwalk① 等自动化工具对设备固件进行解压，得到固件架构信息、组件信息以及启动信息等设备固件自身的关键信息.

此外，通常情况下，物联网设备会提供控制该设备的 Web 管理前端界面，该 Web 管理前端会为设备用户提供一个用于管理设备以及升级固件的可视化交互界面. 本方案通过分析 Web 管理前端程序代码，确定与设备后端进行交互识别的相关参数，并构造出文件的标签树，以此分析前端页面中 html 类型、asp 类型和 xml 类型的文件，利用模拟用户点击操作并提交表单的方式，与设备后端进行交互，获取更多潜在的和设备通信相关的关键参数以及 API 信息.

基于前后端人机协同的固件分析阶段. 根据得到的关键信息，对与设备 Web 服务数据相关联的边界二进制文件进行定位. 基于上一阶段中用户与设备的交互先初步确定后端需要定位的文件，然后结合常用的固件中与 Web 服务相关联的二进制文件，进一步进行精确定位. 接下来，对接受用户交互数据的函数进行识别与分析，主要采用两种方法：(1) 根据相关参数与 API 分析函数功能来识别边界函数；(2) 根据常见的物联网固件函数库以及设备厂商提供的常用的物联网库函数，与目标设备的函数进行比较，进一步补充缺少的关键函数.

在获得了设备固件的关键信息以及关键函数后，HMFuzzer 结合当前获得的信息，利用模糊测试技术对目标设备固件进行基于人机协同的漏洞挖掘工作，以此来检测目标设备固件中潜在的安全威胁. 针对传统模糊测试效率低、覆盖率不足等问题，HMFuzzer 选用了强化学习的方法优化模糊测试整体流程. 强化学习算法可以通过与测试系统的交互来学习如何生成更有效的测试用例. 具体来说，可以将测试系统看作环境，测试用例生成器看作智能体，将测试用例生成过程建模为一个马尔可夫决策过程 (Markov Decision Process，MDP)，并使用强化学习算法求解最优策略. 模糊测试中的测试用例的目标是学习到一个策略 π，使得在状态 s 下执行动作 a 所获得的总奖励最大.

为了能够增强模糊测试流程在固件中找到新执行路径的概率，HMFuzzer 的初始种子样本集设定为空，并维护一个有效的种子样本队列 Qs 和已有种子样本执行过的路径信息的集合 P. 如果经过模糊测试的种子样本在执行过程中产生了新的代码覆盖，则该种子样本的反馈奖励 R 增加，并将该种子样本加入有效队列并更新路径集合 P，并且将 s_{t+1} 作为下一时间的状态输入. 其中，反馈奖励 R 的值基于前期对设备固件的分析并结合识别的关键函数信息进行设定，具体方法见第 3.3 节.

HMFuzzer 利用了 TD3(Twin Delayed DDPG) 算法[37] 作为方案中对模糊测试变异方法的优化算法. TD3 算法是一种用于解决连续控制问题的强化学习算法，它是基于 DDPG(Deep Deterministic Policy Gradient) 算法[38] 进行改进和优化得到的. 为了避免高估问题，TD3 算法使用了双 Critic 的设计，即同时训练两个独立的 Critic 网络，选择其中 Q 值更小的一个作为更新 Actor 网络时的估计值. 此外，TD3 算法还引入了延迟更新机制，即将 Target 网络的更新间隔设定为 Critic 网络的更新间隔的两倍. 延迟更新机制可以使网络更加稳定，防止过度调整参数导致的震荡现象.

HMFuzzer 提出了基于策略梯度算法的变异方法优化算法，如算法 1 所示. 首先，根据当前的输入样本数据 s_t，结合模型训练中学习到的策略 π，具体选择种子样本的变异方法 a_t；其次，根据变异方法 a_t，对输入种子样本数据 s_t 进行修改以及变异，并将其传送给已经完成预处理的物联网设备固件程序，等待种子样本完成模糊测试并获得本次执行得到的反馈奖励数据 R 以及下一轮的数据 s_{t+1}，然后通过分析得到的 R 和 s_{t+1} 对 TD3 算法中的网络参数进行更新. 如果程序出现异常状态，则记录种子样本并暂时终止流程，交由专家验证. 值得注意的是，反馈奖励 R 的取值受到当前样本覆盖率和经过的关键函数数量的影响 (具体的设定方法见第 3.3 节)，其中关键函数来源于对上一阶段提取到的关键信息的分析，如前文所述主要包括两类函数：(1) 利用设备通信相关的关键参数与 API 信息识别出的边界函数；(2) 利用现有函数库与设备函数对比分析出的其他关键函数. 图 3 为变异方法优化算法的功能结构图，其中本文中将 critic 网络和 actor 网络都设置

① https://github.com/ReFirmLabs/binwalk

为三层全连接神经网络.

算法 1. 基于策略梯度算法的变异方法优化算法.

输入：初始种子 s_0

输出：优化变异后的种子

1. 初始化 critic 网络 $Q(s,a)$ 和 actor 网络 $\mu(s|\theta_\mu)$ 的权重 θ_Q，θ_μ
2. 初始化 target 网络 Q' 和 μ' 的权重 $\theta'_Q \leftarrow \theta_Q$ 和 $\theta'_\mu \leftarrow \theta_\mu$
3. 初始化 replay buffer D
4. FOR $episode=1,M$ DO //重复执行 M 个 $episode$
5. 初始化用于动作探索的随机过程 N
6. 接收初始种子 s_0
7. FOR $t=1,T$ DO //重复执行 T 步
8. 根据当前策略和探索噪声选择变异动作 $a_t=\mu(s_t|\theta_\mu)+N_t$
9. 执行动作 a_t 并观测到反馈奖励 R_t 和新状态 s_{t+1}
10. 将 (s_t,a_t,r_t,s_{t+1}) 存储在 replay buffer D 中
11. 从 D 中抽样 N 个转移数据 (s_i,a_i,r_i,s_{i+1})
12. 计算目标值：
13. $y_i=r_i+\gamma \cdot \min_j(Q'(s_{i+1},\mu'(s_{i+1}|\theta'_\mu+N'_t)),j)$
14. 通过最小化损失函数来更新 critic 网络：
15. $L=\frac{1}{N}\sum_i(y_i-Q(s_i,a_i))^2$
16. 每两个 critic 网络更新一次，更新一次 actor 网络：
17. $\nabla_{\theta_\mu}J=\frac{1}{N}\sum_i \nabla_a Q(s_{i+1},\mu(s_{i+1}|\theta_\mu))|_{a=\mu(s_i|\theta_\mu)}$
18. 更新 target 网络的权重：
19. $\theta'_Q \leftarrow \tau * \theta_Q+(1-\tau) * \theta'_Q$ //τ 为更新率
20. $\theta'_\mu \leftarrow \tau * \theta_\mu+(1-\tau) * \theta'_\mu$
21. END FOR
22. END FOR

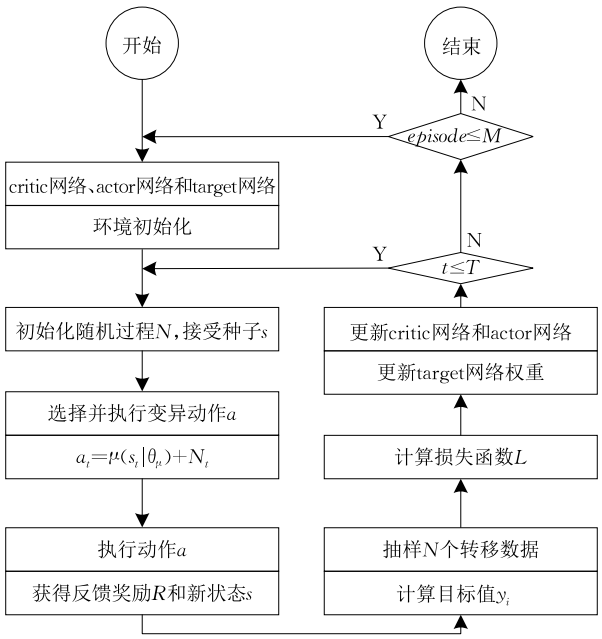

图 3　变异方法优化算法功能结构图

考虑到物联网设备的特性以及整体方案的效率，HMFuzzer 利用设备固件模糊测试常见的种子样本的变异方法，并结合现有的模糊测试工具的设计思路和实现方法，选择八种变异动作空间 A，具体变异方法动作及其作用如表 1 所示.

表 1　HMFuzzer 变异方法动作说明

序号	方法	详细说明	作用
1	基于字节翻转的变异	将输入数据中的某些字节按位翻转	字节级精准变异
2	基于随机数的变异	在输入数据中插入随机数或替换某些字节	小范围的随机变异，容易到达稀有的路径
3	基于字典的变异	将输入数据中的部分字节替换为预定义的字典中的单词或短语	生成具有特定含义的测试用例，基于人工经验的设计更符合程序输入规则
4	基于偏移量的变异	对输入数据进行多次随机变异	多位置的随机变异，生成差异化的输入
5	基于长度的变异	增加或减少输入数据的长度	生成不同大小的测试用例，改变输入携带的信息量
6	基于算术运算的变异	对输入数据中的数字进行算术运算	检测程序的边界条件和异常情况
7	基于结构化的变异	从其他测试用例中截取一部分数据，插入到当前测试用例中	拼接覆盖率更高的测试用例片段
8	基于组合的变异	将输入数据按照不同的顺序进行排列组合	大范围变异，生成差异化更大的输入

3.3　关键参数设定

(1) 反馈奖励 R

在传统的模糊测试过程中，模糊测试最终的目标在于触发发现目标设备固件中的潜在的安全隐患，通常利用设备固件是否产生崩溃或挂起等异常状态作为一个关键的衡量指标. 但模糊测试整个流程开销较大，设备仿真后的吞吐量也较低，所以时间成本较高，如果仅仅以是否发现潜在安全隐患作为种子样本的反馈奖励，那么模糊测试工具往往难以及时调整变异方法，进而会导致计算资源的浪费，使得模糊测试有效性降低.

HMFuzzer 对模糊测试方法进行了调整，设备固件程序在接收种子变异样本状态 s_t 并执行结束后会记录本次的执行路径信息记做 M_t，同时会计算相应的反馈奖励 R. R 的大小取决于当前样本在执行时经过的跳转边所占目标程序所有跳转边的比例，以及测试种子经过关键函数 kp 的数量. 式(1)给出了一个经验性的计算方法，其中 $Path_{\text{total}}$ 表示路径总

数，β 为一个常量参数. 即测试种子对目标程序的路径覆盖率越高，经过的关键函数越多，其作为下一轮变异算法输入的可能性越大.

$$R(t)=\frac{M_t}{Path_{\text{total}}}+\beta\,\text{count}(kp_t) \tag{1}$$

(2) 激活函数设定

在 TD3 算法中，激活函数的选择会对训练效果产生很大的影响. 在通常情况下，在 TD3 算法中可以使用 softplus、elu、linear、relu、sigmoid、softsign、tanh、leaky relu 等几种激活函数，较为常用的有relu、tanh、leaky relu、elu 四种激活函数.

tanh 函数是一种 S 形函数，它可以将输入值映射到[−1,1]的范围内，通常用于输出层或者某些隐藏层，tanh 激活函数可以帮助模型输出符合实际动作空间的连续值. relu 是一种非线性激活函数，可以将负数部分置为 0，从而实现非线性映射，能够有效地提高模型的表达能力和训练速度. 在 relu 的基础上，leaky relu 和 elu 分别进行了优化，leaky relu 函数在 $x<0$ 时不再完全为 0，而是输出一个非零斜率，从而解决了 relu 中存在的死亡神经元问题；与 leaky relu 函数相比，elu 函数在 $x<0$ 时输出一个更平滑的非零斜率，对上述问题进行了进一步优化.

结合上述相关的因素，经过对比实验的分析(见第 4.1 节)和综合考量后，HMFuzzer 选取了 elu 作为激活函数.

3.4 人机协同

随着物联网的迅速发展和设备安全需求的提高，设备固件模糊测试逐渐由最开始的人工手动分析转变为现有主流的自动化工具模糊测试. 然而，自动化设备固件模糊测试工具在减轻安全分析人员手动分析工作量的同时，也为设备固件模糊测试工作代入了一些不确定性因素，增加了误报率. 这是由于模糊测试工具是在以往的安全分析经验的基础上进行设计实现的，而其分析目标是设备固件中存在的安全隐患，工具难以像人工一样对整体流程以及最终结果做出最合适的判断和操作.

因此，HMFuzzer 结合人工分析与自动挖掘两方面的优势，设计了交互式的人机协同漏洞挖掘方法，如图 4 所示. 人工分析阶段能够凭借安全分析人员的经验和判断力适应物联网中复杂多变的环境，对较为复杂的安全问题也能够有更深入的研究. 自动挖掘阶段凭借着自动化工具出色的性能能够适应大规模检测，且和人工分析相比降低了人工时间成本，显著提高了评估效率.

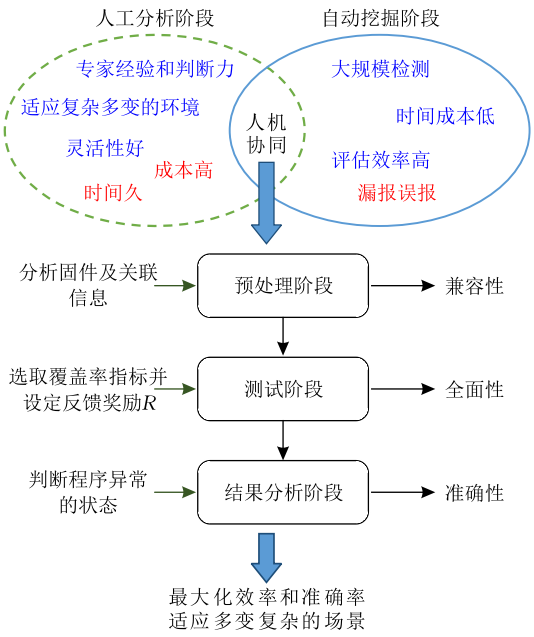

图 4 人机协同的设备固件漏洞挖掘

具体而言，HMFuzzer 在预处理阶段、测试阶段以及最后的结果分析阶段，通过引入人工专家知识经验，实现了方案整体流程的人机协同，将自动化分析和人工分析相结合，以此来提升方案的兼容性、全面性和准确性.

在预处理阶段，HMFuzzer 引入专家经验辅助完成对获得的设备固件的预处理工作. 具体来说，通过安全人员对设备操作手册以及芯片说明的分析，提取设备固件 MCU 类型、操作系统类型、芯片内存分布、外设端口等关键信息，进而为后续多层级关键信息提取与目标固件分析提供基础，同时能够有效地降低方案中的错误判断，避免计算资源的浪费. 例如，固件 Netgear R8300 在使用自动化工具 Binwalk 解包之后，在进行仿真时会缺失存储路由器配置信息的非易失性随机访问存储器(Non-Volatile Random Access Memory，NVRAM)，此时就需要通过人工进行预处理，结合设备操作手册手动创建 NVRAM 文件，并根据配置信息定位关键依赖库，将 NVRAM 读写操作重定向到该文件中，进而为后续固件分析提供支撑.

在测试阶段，HMFuzzer 引入专家经验辅助选取覆盖率指标，进而影响反馈奖励 R 的值，提升模糊测试的变异效率，优化种子变异方向，提升模糊测试方法的覆盖率. 具体而言，传统的自动化模糊测试方案只关注于执行路径占目标程序所有路径的比例，而 HMFuzzer 利用专家经验设定 β 值，增加了关键函数对于种子优先级的影响程度.

对于结果分析阶段，HMFuzzer 引入专家经验辅助，结合自动化测试阶段获取的程序异常状态. 具体来说，设备异常状态包括三种类型：(1) 仿真器对部分底层硬件支持不充分导致的固件仿真缺陷；(2) 设备本身程序设计不合理(比如异常处理机制不完善)导致的异常执行，但不会产生可攻击的漏洞；(3) 可被攻击者利用的漏洞导致的异常. 其中通常只有第 3 类的异常类型是漏洞检测工具所需要关注的，而现有的自动化方法难以准确区分上述三种异常状态，导致了漏报和误报. 因此，HMFuzzer 在该阶段引入专家经验，通过 GDB 调试器等方法分析程序异常的状态，判断其是否为种子样本所导致目标设备固件产生异常，就能有效提升整体方案处理结果的准确性.

4　实验结果与比较分析

HMFuzzer 方案针对物联网设备固件进行漏洞挖掘研究，本文的实验主要分为基于强化学习的模糊测试、漏洞检测能力对比分析以及 0-day 漏洞挖掘三部分内容.

实验使用 Python 开发实现了 HMFuzzer 方案的原型，并将其部署在硬件环境为 12th Gen Intel(R) Core(TM) i7-12700 2100 MHz 12 cores、内存 1.3 TB、运行内存为 64 GB、操作系统为 Ubuntu 20.04.5 LTS x86_64 GNU/Linux 的实验环境中.

4.1　基于强化学习的模糊测试实验

在本节实验中，我们利用 P^{2}IM[12] 的实验数据集选取了对应的 10 个开源物联网设备作为验证场景，包括机器人①、PLC②、网关③、无人机④、数控机床⑤、回焊炉⑥、控制台⑦、转向控制装置⑧、电烙铁装置⑨和热压机②，这些设备不仅在物联网环境中具有代表性，而且对其评估还能更直观地对比展示 HMFuzzer 强化学习算法的有效性.

(1) 强化学习算法对比

首先，我们为了选择出覆盖率最高的强化学习算法，对比分析了不同的强化学习算法对覆盖率的提升程度，如表 2 所示. DQN(Deep Q-Network)[39]、DDPG[38] 和 TD3[37] 算法是三种最为常用的强化学习算法. 这些算法都通过使用神经网络来近似价值函数或策略函数，从而学习最优策略. DQN 算法是一种 Q-learning 算法，它使用神经网络来学习 Q 函数. DQN 算法在训练过程中使用经验回放和目标网络来缓解神经网络中出现的协方差漂移问题. 此外，DQN 算法还使用了贪心策略进行动作选择，该算法主要适用于离散动作空间的问题. DDPG 是一种连续动作空间的策略梯度算法，应用于多种连续控制任务中，它同时学习策略和 Q 函数. DDPG 的策略网络输出连续的动作，而 Q 函数网络则预测当前状态下采取某个动作能得到的最大奖励值. 为了稳定训练，DDPG 还使用了目标网络和经验回放. 相比之下，TD3 算法通过将 Double Q-Learning 算法的思想融入到 DDPG 算法中，结合了深度确定性策略梯度算法和双重 Q 学习，表 2 的实验结果也表明其更加适应 HMFuzzer 固件模糊测试场景，表现出了更高的覆盖率. 因此，本文选用了 TD3 算法作为本方案的强化学习框架.

表 2　HMFuzzer 在不同强化学习算法下的覆盖率对比

设备类型	DQN 覆盖率/%	DDPG 覆盖率/%	TD3 覆盖率/%
机器人	46.3	57.2	62.3
PLC	43.3	51.8	64.3
网关	47.1	54.6	65.6
无人机	50.4	61.9	70.2
数控机床	57.1	70.6	76.4
回焊炉	44.6	57.9	67.1
控制台	39.3	45.9	57.8
转向控制装置	29.8	38.6	47.5
电烙铁装置	56.3	63.3	69.6
热压机	38.4	50.6	56.7

(2) 激活函数对比

如第 3.3 节所述，我们分别使用 relu、tanh、leaky relu、elu 四种常用的激活函数对 10 个设备进行了测试，以对比不同激活函数对模糊测试覆盖率的影响. 实验结果如图 5 所示，HMFuzzer 在使用 elu 函数时覆盖率明显比其它激活函数更高，因此我们选取其作为激活函数.

(3) 模糊测试算法覆盖率对比

我们验证了强化学习算法给模糊测试方法对目标程序覆盖率带来的提升，将基于强化学习的 HMFuzzer 与 P^{2}IM、P^{2}IM+ 的模糊测试方法的覆盖率进行对比，如表 3 所示. 其中，P^{2}IM 方案是一个典型的

① https://github.com/mbocaneg/Inverted-Pendulum-Robot
② https://github.com/CONTROLLINO-PLC/CONTROLLINO_Library/tree/master/MAXI
③ https://github.com/firmata/arduino
④ https://github.com/heethesh/eYSIP-2017_Control_and_Algorithms_development_for_Quadcopter
⑤ https://github.com/deadsy/grbl_stm32f4
⑥ https://github.com/rocketscream/Reflow-Oven-Controller
⑦ https://github.com/RIOTOS/RIOT/tree/master/examples/default
⑧ https://github.com/jabelone/car-controller
⑨ https://github.com/Ralim/ts100

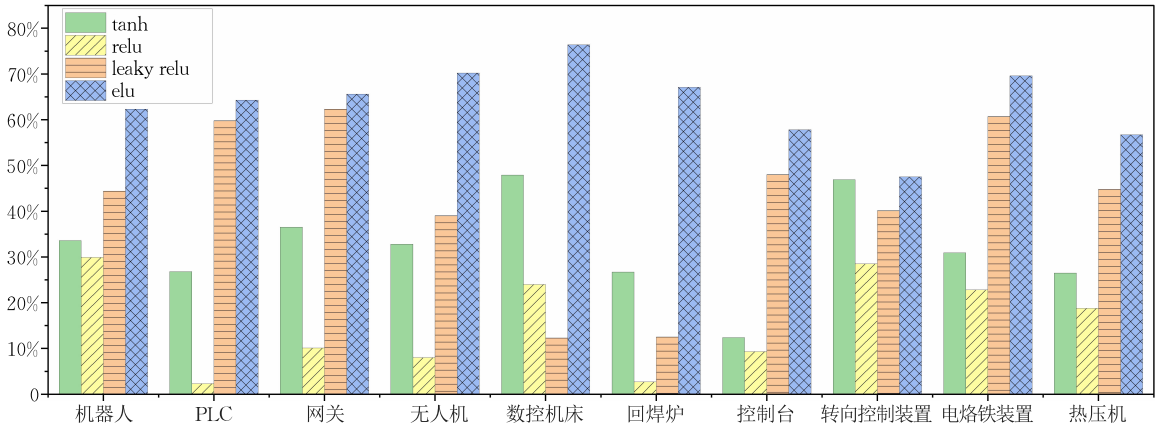

图 5 不同激活函数对模糊测试覆盖率的影响

表 3 不同漏洞挖掘方法的漏洞检测能力对比

设备厂商(漏洞个数)	HMFuzzer	IoTFuzzer	KARONTE	SaTC
Netgear(12)	11	8	9	9
D-Link(24)	22	16	18	20
Cisco(7)	6	4	4	5
TOTOLink(7)	7	3	4	7
H3C(20)	17	11	14	14
TrendNet(13)	9	6	7	7
Tenda(27)	26	23	24	25
识别成功率/%	**89.1**	**64.5**	**72.7**	**79.1**

自动化模糊测试工具,提出了基于外设接口的抽象建模,适用于多种架构的嵌入式设备,具有较强的通用性.同时,我们将 P^{2}IM 原文中提到的方案优化方法整合并实现为 P^{2}IM+,具体包括以下 2 个方面的优化:(1)将内核的 AFL 组件替换为了 Taintscope,以提升 P^{2}IM 的对复杂路径条件的处理能力;(2)改进了内存错误检测器,采用多种启发式跟踪方法加强内存异常访问的检测能力.因此,相比于 P^{2}IM,P^{2}IM+具备更高的覆盖率.

图 6 展示了 HMFuzzer 与现有模糊测试方案在 10 个目标设备上覆盖率与时间的关系.实验结果表明,HMFuzzer 相较于现有技术,在覆盖率方面有显著提升,而且在更短时间内就能够达到 P^{2}IM 和 P^{2}IM+的最大覆盖率,这意味着 HMFuzzer 在实际漏洞挖掘过程中能更快地探索程序路径.这是因为相比于现有的自动化物联网设备漏洞挖掘方案,HMFuzzer 结合了专家经验对反馈奖励 R 值的影响和强化学习算法,提高了种子筛选的效率,从而显著提升了测试工具对目标程序的覆盖率.

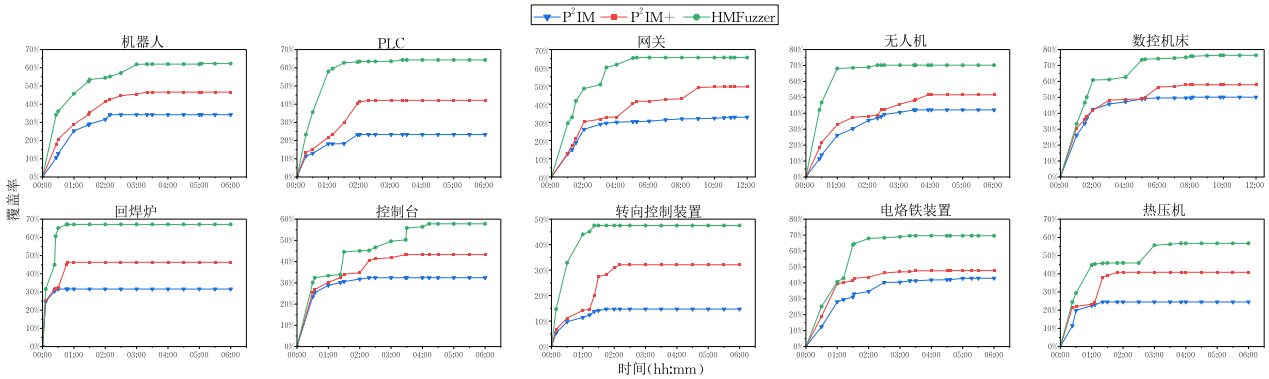

图 6 不同模糊测试算法覆盖率时间对比

4.2 漏洞检测能力对比实验

本文收集了 Netgear、D-Link、Cisco、TOTOLink、Tenda、H3C、TrendNet 七个设备厂商的 37 个已知存在安全漏洞的设备固件,然后整理出了针对这些固件的已披露的 110 个漏洞.

接下来,我们将 HMFuzzer 方案应用到这些设备上进行检测,并与 IoTFuzzer[7]、KARONTE[13]以及 SaTC[14]三个现有的物联网固件漏洞挖掘方案进行了对比,以证明 HMFuzzer 的漏洞检测能力,其中 IoTFuzzer 是一个经典的前后端交互式自动化漏洞挖掘方案,能够在不依赖任何协议规范知识的前提下实现目标设备探测;KARONTE 是一个基于静

态分析的漏洞挖掘方案，通过对二进制的交互分析检测不安全的数据传播；SaTC 则是一个模块化的漏洞挖掘工具，通过提取前端关键字并利用污点分析实现高效漏洞检测. 而相比于其他自动化固件漏洞挖掘工具，HMFuzzer 首先通过对这些设备操作手册以及芯片说明的分析和设备固件前后端交互，提取设备固件 MCU 类型、操作系统类型、芯片内存分布、外设端口、关键函数等关键信息. 然后在测试阶段，我们将常量参数 β 设定为 0.015，进而影响反馈奖励 R 的值. 最后，在结果分析阶段，通过人工对异常的程序状态进行分析，从而得到最终的漏洞检测结果.

表 3 展示了不同漏洞挖掘方法的漏洞检测能力对比实验结果. 值得注意是，因为在漏洞检测的过程中会涉及时间难以量化的人工开销，因此我们只对比了这些固件漏洞挖掘方案的漏洞识别成功率. 表 3 的实验结果表明面向不同设备厂商存在漏洞的设备，HMFuzzer 具备更强的漏洞检测能力. 具体来说，与已有的方法相比，HMFuzzer 识别漏洞的总成功率能够提升 10%～25%. 其根本原因主要包含两个方面：(1) 相比于自动化漏洞挖掘工具，HMFuzzer 通过专家经验对反馈奖励 R 值的影响和强化学习算法能够覆盖更多的程序路径，进而挖掘到稀有的路径漏洞；(2) 现有自动化工具存在较多的漏报和误报现象，专家经验能够更加准确地判断程序的异常状态，进而挖掘出更多的被自动化工具忽略的漏洞信息.

4.3　0-day 漏洞挖掘实验

我们将 HMFuzzer 应用于分析真实的物联网设备固件，利用多层级信息提取的方法，结合前后端关联分析，定位后端如 websgetvar、recvmsg 等关键函数，然后通过模糊测试进行漏洞检测，基于预先定义好的反馈奖励 R 计算公式，对测试用例进行评估，如果出现异常状态，则交由安全分析人员进行进一步确认. 具体来说，我们对 Tenda、H3C、TOTOLink 三个厂商的设备进行了逐一分析，排除已知漏洞、方案误报等情况，实现对 0-day 漏洞的挖掘.

最终，HMFuzzer 发现了多个 0-day 漏洞，而且我们对其中的 4 个漏洞构造了漏洞利用样本，并提交给了专业的漏洞平台进行验证，最终获得了 4 个 CVE/CNVD 高危漏洞编号. 具体而言，针对 TOTOLink 路由器，拿到了漏洞编号 CVE-2023-30053 以及 CVE-2023-30054，CVSS 评分均达到 9.8 分；针对 H3C 和 Tenda 的设备，拿到了高危漏洞编号 CNVD-2023-16187 和 CNVD-2023-15074. 这些漏洞利用缓冲区溢出，使得攻击者可以通过精心构造的 payload 获取一个稳定的 root shell，拿到服务器的权限，并且攻击者还可能以此作为跳板，进而对整个物联网系统造成不可估量的损害. 经过我们分析发现，触发这些漏洞的种子在路径覆盖的表现上并不突出，因此不受现有的自动化漏洞检测工具重视，这些漏洞也难以被发现. 而 HMFuzzer 通过专家经验对反馈奖励 R 的设定，提高了关键函数对种子优先级的影响，从而增强了 HMFuzzer 对 0-day 漏洞检测能力.

综上所述，0-day 漏洞挖掘实验结果表明，HMFuzzer 方案具备更强的漏洞挖掘能力，确实能够发现真实的物联网设备中的 0-day 漏洞.

5　结束语

本文提出了一种基于人机协同的设备固件漏洞挖掘方案 HMFuzzer. 针对现有设备固件分析方案目标单一的问题，提出了基于设备固件前后端交互的设备固件关键信息提取方法，通过设备固件、管理前端以及用户三方交互模式，能够有效地获取目标设备关键信息，并定位出目标设备固件边界二进制文件，分析出后端关键函数. 此外，HMFuzzer 将专家经验引入了漏洞挖掘的全流程中，结合强化学习算法，对目标设备固件实现了基于人机协同的漏洞挖掘工作，显著增强了模糊测试的覆盖率和效率，并提升了方案漏洞挖掘的能力. 实验结果表明，相比于现有方法，HMFuzzer 在漏洞识别成功率提高了 10%以上，具备更强的漏洞检测能力. 此外，针对不同厂商的真实物联网设备固件，HMFuzzer 发现了多个 0-day 漏洞，其中已获得 4 个 CVE/CNVD 高危漏洞.

参　考　文　献

[1] Xiong Jin-Bo, Bi Ren-Wan, Tian You-Liang, et al. Security and privacy in mobile crowdsensing: Model, progresses, and trends. Chinese Journal of Computers, 2021, 44(9): 1949-1966(in Chinese)
(熊金波，毕仁万，田有亮等. 移动群智感知安全与隐私：模型、进展与趋势. 计算机学报，2021，44(9)：1949-1966)

[2] Kuang B, Fu A, Gao Y, et al. FeSA: Automatic federated swarm attestation on dynamic large-scale IoT devices. IEEE

Transactions on Dependable and Secure Computing, 2023, 20(4): 2954-2969

[3] Yue Hong-Zhou, Zhang Yu-Qing. Vulnerability analysis and exploitation of location privacy leakage in webcasting platforms. Chinese Journal of Computers, 2019, 42(5): 1095-1111(in Chinese)
(乐洪舟, 张玉清. 网络直播平台主播地理位置泄露漏洞的分析与利用. 计算机学报, 2019, 42(5): 1095-1111)

[4] Kuang B, Fu A, Susilo W, et al. A survey of remote attestation in Internet of Things: Attacks, countermeasures, and prospects. Computers & Security, 2022, 112: 102498

[5] Zhang Wei-Yao, Zhang Lei, Mao Jian-Ling, et al. An automated method of unknown protocol fuzzing test. Chinese Journal of Computers, 2020, 43(4): 653-667(in Chinese)
(张蔚瑶, 张磊, 毛建瓴等. 未知协议的逆向分析与自动化测试. 计算机学报, 2020, 43(4): 653-667)

[6] Yu Ying-Chao, Chen Zuo-Ning, Gan Shui-Tao, Qin Xiao-Jun. Research on the technologies of security analysis technologies on the embedded device firmware. Chinese Journal of Computers, 2021, 44(5): 859-881(in Chinese)
(于颖超, 陈左宁, 甘水滔, 秦晓军. 嵌入式设备固件安全分析技术研究. 计算机学报, 2021, 44(5): 859-881)

[7] Chen J, Diao W, Zhao Q, et al. IoTFuzzer: Discovering memory corruptions in IoT through app-based fuzzing//Proceedings of the Network and Distributed Systems Security Symposium. San Diego, USA, 2018: 1-15

[8] Zhang Y, Huo W, Jian K, et al. SRFuzzer: An automatic fuzzing framework for physical SOHO router devices to discover multi-type vulnerabilities//Proceedings of the Annual Computer Security Applications Conference. San Juan, USA, 2019: 544-556

[9] Srivastava P, Peng H, Li J, et al. FirmFuzz: Automated IoT firmware introspection and analysis//Proceedings of the International ACM Workshop on Security and Privacy for the Internet-of-Things. London, UK, 2019: 15-21

[10] Zheng Y, Davanian A, Yin H, et al. Firm-AFL: High-throughput greybox fuzzing of IoT firmware via augmented process emulation//Proceedings of the USENIX Security Symposium. Santa Clara, USA, 2019: 1099-1114

[11] Redini N, Continella A, Das D, et al. DIANE: Identifying fuzzing triggers in apps to generate under-constrained inputs for IoT devices//Proceedings of the IEEE Symposium on Security and Privacy. San Francisco, USA, 2021: 484-500

[12] Feng B, Mera A, Lu L. P^{2}IM: Scalable and hardware-independent firmware testing via automatic peripheral interface modeling//Proceedings of the USENIX Conference on Security Symposium. Virtual Event, 2020: 1237-1254

[13] Redini N, Machiry A, Wang R, et al. Karonte: Detecting insecure multi-binary interactions in embedded firmware//Proceedings of the IEEE Symposium on Security and Privacy. San Francisco, USA, 2020: 1544-1561

[14] Chen L, Wang Y, Cai Q, et al. Sharing more and checking less: Leveraging common input keywords to detect bugs in embedded systems//Proceedings of the USENIX Security Symposium. Virtual Event, 2021: 303-319

[15] Scharnowski T, Bars N, Schloegel M, et al. FuzzWare: Using precise MMIO modeling for effective firmware fuzzing//Proceedings of the USENIX Security Symposium. Boston, USA, 2022: 1239-1256

[16] Xie W, Chen J, Wang Z, et al. Game of hide-and-seek: Exposing hidden interfaces in embedded web applications of IoT devices//Proceedings of the ACM Web Conference 2022. Lyon, France, 2022: 524-532

[17] Zheng H, Zhang J, Huang Y, et al. FishFuzz: Catch deeper bugs by throwing larger nets//Proceedings of the USENIX Security Symposium. Anaheim, USA, 2023: 1343-1360

[18] Fang Hao-Ran, Guo Fan, Li Hang-Yu. TaintPoint: Fuzzing taint flow efficiently with live trace. Journal of Software, 2022, 33(6): 1978-1995(in Chinese)
(方浩然, 郭帆, 李航宇. TaintPoint:使用活跃轨迹高效挖掘污点风格漏洞. 软件学报, 2022, 33(6): 1978-1995)

[19] Costin A, Zarras A, Francillon A. Automated dynamic firmware analysis at scale: A case study on embedded web interfaces//Proceedings of the ACM on Asia Conference on Computer and Communications Security. Xi'an, China, 2016: 437-448

[20] Rosenfeld K, Karri R. Attacks and defenses for JTAG. IEEE Design & Test of Computers, 2010, 27(1): 36-47

[21] Chen D D, Egele M, Woo M, et al. Towards automated dynamic analysis for Linux-based embedded firmware//Proceedings of the Network and Distributed Systems Security Symposium. San Diego, USA, 2016: 1-16

[22] Kammerstetter M, Burian D, Kastner W. Embedded security testing with peripheral device caching and runtime program state approximation//Proceedings of the International Conference on Emerging Security Information, Systems and Technologies. Nice, France, 2016: 21-26

[23] Koscher K, Kohno T, Molnar D. SURROGATES: Enabling near-real-time dynamic analyses of embedded systems//Proceedings of the USENIX Workshop on Offensive Technologies. Washington, USA, 2015: 1-15

[24] Zaddach J, Bruno L, Francillon A, et al. AVATAR: A framework to support dynamic security analysis of embedded systems' firmwares//Proceedings of the Network and Distributed Systems Security Symposium. San Diego, USA, 2014: 1-16

[25] Hu Yu-Tao, Wang Su-Yuan, Wu Yue-Ming, et al. Slice-level vulnerability detection and interpretation method based on graph neural network. Journal of Software, 2023, 34(6): 2543-2561(in Chinese)
(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. 软件学报, 2023, 34(6): 2543-2561)

[26] Li Yun, Huang Chen-Lin, Wang Zhong-Feng, et al. Survey of software vulnerability mining methods based on machine learning. Journal of Software, 2020, 31(7): 2040-2061(in Chinese)
(李韵，黄辰林，王中锋等. 基于机器学习的软件漏洞挖掘方法综述. 软件学报，2020，31(7)：2040-2061)

[27] Yu Ying-Chao, Gan Shui-Tao, Qiu Jun-Yang, et al. Binary code similarity analysis and its applications on embedded device firmware vulnerability search. Journal of Software, 2022, 33(11): 4137-4172(in Chinese)
(于颖超，甘水滔，邱俊洋等. 二进制代码相似度分析及在嵌入式设备固件漏洞搜索中的应用. 软件学报，2022，33(11)：4137-4172)

[28] Feng Q, Zhou R, Xu C, et al. Scalable graph-based bug search for firmware images//Proceedings of the ACM SIGSAC Conference on Computer and Communications Security. Vienna, Austria, 2016: 480-491

[29] Gao J, Yang X, Fu Y, et al. VulSeeker: A semantic learning based vulnerability seeker for cross-platform binary//Proceedings of the IEEE/ACM International Conference on Automated Software Engineering. Montpellier, France, 2018: 896-899

[30] Gao J, Yang X, Fu Y, et al. VulSeeker-pro: Enhanced semantic learning based binary vulnerability seeker with emulation//Proceedings of the ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. Lake Buena Vista, USA, 2018: 803-808

[31] Gao J, Jiang Y, Liu Z, et al. Semantic learning and emulation based cross-platform binary vulnerability seeker. IEEE Transactions on Software Engineering, 2019, 47(11): 2575-2589

[32] Ding S H H, Fung B C M, Charland P. Asm2Vec: Boosting static representation robustness for binary clone search against code obfuscation and compiler optimization//Proceedings of the IEEE Symposium on Security and Privacy. San Francisco, USA, 2019: 472-489

[33] Duan Y, Li X, Wang J, et al. DeepBinDiff: Learning program-wide code representations for binary diffing//Proceedings of the Network and Distributed Systems Security Symposium. San Diego, USA, 2020: 1-16

[34] Böhme M, Pham V T, Roychoudhury A. Coverage-based greybox fuzzing as Markov chain. IEEE Transactions on Software Engineering, 2017, 45(5): 489-506

[35] Yue T, Wang P, Tang Y, et al. EcoFuzz: Adaptive energy-saving greybox fuzzing as a variant of the adversarial multi-armed bandit//Proceedings of the USENIX Security Symposium. Virtual Event, 2020: 2307-2324

[36] Wang J, Song C, Yin H. Reinforcement learning-based hierarchical seed scheduling for greybox fuzzing//Proceedings of the Network and Distributed Systems Security Symposium. Virtual Event, 2021: 1-17

[37] Fujimoto S, Herke H, David M. Addressing function approximation error in actor-critic methods//Proceedings of the International Conference on Machine Learning. Stockholm, Sweden, 2018: 1589-1596

[38] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971, 2015

[39] Watkins C J, Dayan P. Q-learning. Machine Learning, 1992, 8: 279-292

KUANG Bo-Yu, Ph. D., postdoctoral researcher. His current research interests include IoT security and vulnerability mining.

ZHANG Zhao-Bo, M. S. His current research interests include IoT security and vulnerability mining.

YANG Shan-Quan, Ph. D. candidate. His current research interests include IoT security and vulnerability mining.

SU Mang, Ph. D., associate professor. Her main research interests include cloud security, access control, and right management.

FU An-Min, Ph. D., professor, Ph. D. supervisor. His research interests include cryptography and privacy preserving.

Background

The Internet of Things (IoT) is a global system of many computing devices that communicate with each other and has become an integral part of our lives. With the widespread adoption of IoT technologies in many areas such as critical infrastructure, industrial sectors, and smart homes, the security risks of its architecture have increased and the attack surfaces have become more diverse. In recent years, malicious attacks and security incidents related to the IoT have been frequent,

which are often caused by exploitable security vulnerabilities in IoT devices.

In this context, security researchers propose vulnerability mining approaches for IoT devices to identify security threats ahead of attackers and take corresponding security measures to maintain the safe and stable operation of the system, thereby minimizing security risks. However, most of the existing research works on IoT device firmware vulnerability mining have overly focused on automated vulnerability mining. Although automated vulnerability mining reduces the labor cost, it limits the flexibility and scalability of the solution and ignores the benefits of expert experience. Expert experience can greatly enhance the compatibility of automated fuzzers, improve the efficiency of test seed evolution, and also help automated tools to discern anomalous program states that are difficult to resolve, which in turn improves the ability of vulnerability mining.

In order to effectively combine expert experience with automated vulnerability mining and improve the efficiency of device firmware vulnerability mining, this paper proposes a human-machine collaboration-based device firmware vulnerability mining scheme, HMFuzzer. It analyzes the target device firmware by combining device management interface information and user simulation interaction information. Then HMFuzzer extracts the key information and guides the subsequent fuzzing seed generation. Besides, HMFuzzer introduces expert experience and combines with the reinforcement learning algorithm, optimizing the seed variation and improving the coverage rate and efficiency of fuzzing.

The experimental results show that compared to existing methods, the vulnerability identification success rate of HMFuzzer can be improved by more than 10% and has a stronger capability to detect vulnerabilities. Especially, HMFuzzer finds multiple 0-day vulnerabilities against real vendor's IoT devices, including 4 CVE/CNVD high-risk vulnerabilities.

This work is supported by the National Natural Science Foundation of China (Nos. 62072239, 62372236), the Natural Science Foundation of Jiangsu Province, China (No. BK2021-1192), the Future Network Scientific Research Fund Project (No. FNSRFP-2021-ZD-05), the Fundamental Research Funds for the Central Universities (No. 30921013111), the Qing Lan Project of Jiangsu Province, and Jiangsu Funding Program for Excellent Postdoctoral Talent.