西北大学信息科学与技术学院

本科毕业设计开题报告/答辩登记表

<table>
<tr><td>学生学号</td><td>2021117283</td><td>姓名</td><td>雷璟锟</td><td>年级</td><td colspan="2">2021 级</td></tr>
<tr><td>专业</td><td colspan="6">软件工程</td></tr>
<tr><td>论文（设计）题　　目</td><td colspan="6">嵌入式 JavaScript 引擎模糊测试方法研究</td></tr>
<tr><td>指导教师姓　　名</td><td>叶贵鑫</td><td colspan="2">专业技术职务</td><td>副教授</td><td>开题报告日期</td><td></td></tr>
<tr><td>企业导师姓　　名</td><td colspan="3"></td><td>开题报告成绩</td><td></td></tr>
<tr><td colspan="7">答辩小组成员（姓名，职称）：</td></tr>
<tr><td colspan="7">答辩小组组长签字：　　年　　月　　日</td></tr>
<tr><td colspan="7">开 题 报 告 内 容</td></tr>
<tr><td>选题来源</td><td colspan="6">1．教师指定（ √ ）2．教师课题（ ）3．创新基金项目（ ）4．自选（ ）</td></tr>
<tr><td>设计选题的背景与意义、理论与实证准备、拟解决的问题、研究（设计）方法与技术路线</td><td colspan="6">1．选题的背景与意义
随着嵌入式系统在物联网、智能设备、汽车电子等领域的广泛应用，嵌入式开发中的软件需求日益增加。在这些应用中，JavaScript 作为一种高效且具有灵活性的脚本语言，已经逐渐被应用到嵌入式系统中。嵌入式 JavaScript(JS) 引擎的优化和测试问题因此成为了研究的热点。
模糊测试是目前应用最为广泛的缺陷检测技术，检测方式为通过生成或者突变 的方式生成大量正常和异常的输入，然后将生成的输入提供给目标应用程序并监视程序执行状态来检测异常。相比于符号执行、污点分析等缺陷检测技术，模糊测试易于部署并且具有良好的可拓展性和适用性，并且在有无源代码的情况下都可以执行。此外，由于模糊测试是在程序实际执行中进行的，模糊测试具有很高的准确性，而且，模糊测试不需要对目标应用程序有很高的了解程度，因此常和差异测试同时使用来发现软件的错误。
受限于运行环境，嵌入式 JavaScript 引擎不仅需要满足高效、低资源消耗等硬性要求，还必须具备良好的健壮性和可靠性。为此，测试和验证这些引擎的正确性和性能变得尤为重要。而模糊测试作为一种重要的自动化测试方法，能够有效地发现潜在的安全漏洞、内存泄漏和性能瓶颈，因此在嵌入式 JavaScript 引擎的优化模块中应用模糊测试方法显得十分必要。本课题的意义在于通过研究和优化嵌入式 JavaScript 引擎的测试方法，提高其稳定性、安全性和性能，从而推动编译器安全的应用发展。</td></tr>
</table>

	2．理论与实证准备 1）阅读了嵌入式 JavaScript 引擎及其模糊测试的文献资料。 2）熟悉和选用适用于 JavaScript 引擎的模糊测试工具和框架，或根据需求开发定制化的模糊测试工具。 3）学习嵌入式 JavaScript 引擎的工作原理及其特性。 **3．拟解决的问题** 1）实现自动化的模糊测试流程，包括测试用例的生成、执行以及错误报告机制，以便快速发现和定位引擎中的缺陷。 2）如何根据模糊测试的结果进行引擎优化，确保能够在保持高效执行的同时，提升其容错性和安全性。 3）编译器需要处理各种语言特性和语法规则，测试用例的设计和覆盖面需要非常广泛。需要设计大量的测试用例来覆盖各种语言特性和边界情况，确保编译器在各种输入下都能正确工作。 **4．研究方法与技术路线** 针对上述存在问题，本文会去解决上面三个问题，本系统以变异因子对测试用例进行变异。具体研究内容如下 1）JavaScript 引擎软件测试方法研究。对常见的软件测试方法进行了深入的研究，分析了各种软件测试方法的自动化程度、测试用例产生途径和其特异的测试目标。 2）设计合适的模糊测试输入生成策略。爬取开源 JavaScript 代码构建原始语料库，预处理后执行多种变异生成大量可靠的测试用例。 3）对最终的测试结果进行准确性评估，使用标准的漏洞检测和代码覆盖率工具，计算测试覆盖率、错误报告数量、误报率等。通过测试结果来衡量模糊测试方法的有效性和稳定性。
论文写作提纲	第 1 章 绪论 1.1 研究背景和意义 1.2 国内外研究现状 1.3 本文研究内容 1.4 本文组织结构 第 2 章 相关理论与技术 2.1 JavaScript 引擎 2.2 模糊测试理论 2.3 测试用例变异技术 第 3 章 EJSFuzz 系统实现与设计 3.1 系统设计 3.1.1 框架实现与模块设计 3.1.2 用例变异程序介绍 3.2 系统实现 3.3 系统功能展示 第 4 章 系统实验结果分析 4.1 实验设置 4.2 实验结果

	第 5 章 总结与展望 5.1 总结 5.2 展望 **参考文献：** [1] 姚厚友.面向嵌入式 JavaScript 引擎的差分模糊测试方法研究[D].2021. [2] 田洋.基于标准文档分析的 JavaScript 引擎缺陷检测方法研究[D].2021. [3] 孙力立,武成岗,许佳丽,等.脚本语言执行引擎的模糊测试技术综述[J].高技术通讯.2022,32(12).DOI:10.3772/j.issn.1002-0470.2022.12.002 . [4] 曹帅.基于类型推断的 JavaScript 引擎模糊测试方法研究[D].2020. [5] 卢凌. 面向 JavaScript 引擎报错机制的类别导向模糊测试方法[D]. 辽宁:大连理工大学,2023. [6] 周阳. 基于模糊测试的 JavaScript 引擎缺陷检测方法[D]. 湖北:华中科技大学,2022. [7] 西北大学.一种基于类型推断的具有引导性的测试用例变异方法 :CN202010200651.0[P].2020-03-20. [8] 程勇,秦丹,杨光. 针对 JavaScript 浏览器兼容性的变异测试方法[J]. 计算机应用,2017,37(4):1143-1148,1173. DOI:10.11772/j.issn.1001-9081.2017.04.1143. [9] 刘艺玮. 基于深度学习的 JavaScript 引擎模糊测试方法研究[D]. 四川:电子科技大学,2024. [10] 余启洋, 桑楠, 郭文生. 嵌入式浏览器 JavaScript 引擎的研究与设计[J]. 计算机应用与软件, 2014, 5. [11] 王聪冲. 面向 JavaScript 解析引擎的模糊测试技术研究[D]. 江南大学, 2021. [12] 王允超, 王清贤, 丁文博. 语义感知的 JavaScript 引擎模糊测试技术研究[J]. 信息工程大学学报, 2020. [13] 吴泽君,武泽慧,王允超,等. 基于自然语言处理的 JavaScript 引擎定向模糊测试技术[J]. 信息工程大学学报,2022,23(6):737-745. DOI:10.3969/j.issn.1671-0673.2022.06.014. [14] SOYEON PARK, WEN XU, INSU YUN, et al. Fuzzing JavaScript Engines with Aspect-preserving Mutation[C]//2020 IEEE Symposium on Security and Privacy: IEEE Symposium on Security and Privacy (SP 2020), 18-21 May 2020, San Francisco, CA, USA.:Institute of Electrical and Electronics Engineers, 2020:1629-1642. [15] Dinh S T, Cho H, Martin K, et al. Favocado: Fuzzing the Binding Code of JavaScript Engines Using Semantically Correct Test Cases[C]//NDSS. 2021. [16] Xu H, Jiang Z, Wang Y, et al. Fuzzing JavaScript Engines with a Graph-based IR[C]//Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security. 2024: 3734-3748.

	[17] Groß S, Koch S, Bernhard L, et al. FUZZILLI: Fuzzing for JavaScript JIT Compiler Vulnerabilities[C]//NDSS. 2023. [18] Wang J, Zhang Z, Liu S, et al. {FuzzJIT}:{Oracle-Enhanced} Fuzzing for {JavaScript} Engine {JIT} Compiler[C]//32nd USENIX Security Symposium (USENIX Security 23). 2023: 1865-1882. [19] Bin, Zhang,Jiaxi, Ye,Xing, Bi,等.Ffuzz: Towards full system high coverage fuzz testing on binary executables.[J].PLoS ONE.2018,13(5).e0196733.DOI:10.1371/journal.pone.0196733 . [20] Klaus Greff,Rupesh K. Srivastava,Jan Koutník,等.LSTM: A Search Space Odyssey[J].IEEE Transactions on Neural Networks and Learning Systems",",pubMedId":"27411231.2017,28(10).2222-2232.DOI:10.1109/TNNLS.2016.2582924 .
工作步骤与时间安排	2024 年 11 月 20 日-2024 年 12 月 26 日：阅读相关文献。 2024 年 12 月 26 日-2025 年-1 月 3 日：学习 python 相关语法，模糊测试等技术。 2025 年-1 月 03 日-2025 年 1 月 7 日：完成开题报告并完成答辩。 2025 年 1 月 10 日-2025 年 2 月 1 日： 开发自动化工具搜集 JavaScript 语料库，并进行语料的去重与精简。撰写关于语料数据获取和预处理部分的毕业论文。 2025 年 2 月 1 日-2025 年 3 月 9 日：编写代码，构建基于类型推断的变异策略，进行预实验并优化，撰写关于 EJSFuzz 系统设计的毕业论文。 2025 年 3 月 10 日-2025 年 3 月 25 日：完成论文初稿撰写。 2025 年 3 月 25 日-2025 年 4 月 15 日：继续完善代码，并完成毕业论文的终稿。 2025 年 4 月 15 日-2025 年 5 月 26 日：提交毕业论文并完成毕业答辩。

开题答辩评语	指导教师签名： 年　月　日

注：此表由学生填写后交指导教师签署意见，并交院系教务办保存，最终将作为毕业设计最终评分的依据。