基于变异的测试用例生成技术综述

彭渤生 [1)] 邢骏洋 [1)] 王 京 [1)] 肖家生 [1)]

[1)](南京大学软件学院, 南京, 210093)

摘 要 * 中文摘要内容置于此处 (英文摘要中要有这些内容)，字体为小 5 号宋体。摘要贡献部分，要有数据支持，不要出现 “... 大大提高”、“... 显著改善” 等描述，正确的描述是 “比…提高 X%”、“在…上改善 X%”。* 摘要

关键词 * 关键词（中文关键字与英文关键字对应且一致，应有 5-7 个关键词)；关键词；关键词；关键词 *

中图法分类号 TP DOI 号: * 投稿时不提供 DOI 号

A Survey for Test Generation Techniques Based on Mutation Technology

PENG Bo-Sheng[1)] XING Jun-Yang[1)] WANG Jing[1)] XIAO Jia-Sheng[1)]

[1)](Software Institute, Nanjing University, Nanjing 210093, China)

Abstract (500 **英文单词，内容包含中文摘要的内容**). 字体为 Times new Roman, 字号 5 号 * Abstract

Do not modify the amount of space before and after the artworks. One- or two-column format artworks are preferred. and Tables, create a new break line and paste the resized artworks where desired. Do not modify the amount of space before and after the artworks. One- or two-column format artworks are preferred. All Schemes, Equations, Figures, and Tables should be mentioned in the text consecutively and numbered with Arabic numerals, and appear below where they are mentioned for the first time in the main text. To insert Schemes, Equations, Figures, and Tables, create a new break line and paste the resized artworks where desired. Do not modify the amount of space before and after the artworks. One- or two-column format artworks are preferred.Do not modify the amount of space before and after the artworks. One- or two-column format artworks are preferred. and Tables, create a new break line and paste the resized artworks where desired. Do not modify the amount of space before and after the artworks. One- or two-column format artworks are preferred. All Schemes, Equations, Figures, and Tables should be mentioned in the text consecutively and numbered with Arabic numerals, and appear below where they are mentioned for the first time in the main text.

Keywords 中文关键字与英文关键字对应且一致，**不要用英文缩写**); key word; key word; key word* * 字体为 5 号 Times new Roman * Key words

---

彭渤生，性别男，xxxx 年生，学位 (或目前学历)，职称，是/否计算机学会 (CCF) 会员（提供会员号），主要研究领域为 *****、****.E-mail: **********. 邢骏洋，性别男，2004 年生，本科生，职称，主要研究领域为图形化界面软件测试.E-mail: xingjunyang@smail.nju.edu.cn. 王京（通信作者），性别男，xxxx 年生，学位 (或目前学历)，职称，是/否计算机学会 (CCF) 会员（提供会员号），主要研究领域为 *****、****.E-mail: **********. 肖家生男，性别，xxxx 年生，学位 (或目前学历)，职称，是/否计算机学会 (CCF) 会员（提供会员号），主要研究领域为 *****、****.E-mail: **********. 第 1 作者手机号码 (投稿时必须提供，以便紧急联系，发表时会删除): ……, E-mail: ……* 此部分 6 号宋体 *

1 引言

随着信息化的发展，软件系统的数量、规模和复杂度不断增大。在软件系统的开发过程的，软件测试与漏洞发掘总是至关重要的。遗留的漏洞可能遭受黑客的利用，造成重大的经济损失和信息泄露。例如，2024 年 7 月美国网络安全公司 CrowdStrike 发布异常更新，导致全球超过 850 万台 Windows 设备陷入“蓝屏死机”循环，甚至无法启动。此事件导致银行、航空、电信等多个关键行业的业务中断，重创了包括澳大利亚政府在内的多个组织的基础设施。尽管这一问题迅速被修复，但它揭示了软件更新测试不足和对关键依赖过于集中所带来的高风险 [1]。另一个例子是 2017 年的勒索软件 WannaCry 攻击，这次攻击影响了大约 150 个国家的约 23 万至 30 万台计算机，估计全球经济损失达 40 至 80 亿美元。这次攻击发生的原因是用户没有使用 Microsoft Windows 发布的安全补丁更新他们的系统 [2]。

为了减少软件漏洞带来的损害，各类软件测试技术蓬勃发展，其中模糊测试已成为当前最流行的方法之一。模糊测试是一种通过随机或半随机的方式生成测试用例，将其输入到目标系统后监听异常结果来发现安全漏洞的安全检测技术。在模糊测试的流程中，输入生成的主要作用是通过不同策略生成能够引发程序异常行为、潜在漏洞或者新代码路径的输入数据。这类样例也被称为“有趣”的样例。确保样例的“有趣”才能保证在后续的测试运行时高效，覆盖率高。从而提高程序的安全性与鲁棒性。

模糊测试输入生成通常遵循以下流程：首先，选择模糊测试目标。可以是源代码、二进制代码，应用程序等。然后必须了解目标接受的输入格式和协议，确保输入数据服务目标预期。生成测试用例时，可以参考之前经过验证的有趣的用例集，或者直接生成新用例。用例的生成策略主要有随机生成和基于语法生成。当测试执行完成之后，对于触发异常的用例，可以保存以供重用或变异后产生新的测试样例。

基于变异的随机生成测试用例 [3–5] 是一种常见的技术，通过注入随机且不可预测的输入来检测应用程序中的漏洞和缺陷。这种方法有助于生成在代码演化或测试阶段尚未探索的新测试用例。然而，随机生成的局限性在于其变异能力的边界限制 - 它只能在现有种子语料库的范围内生成测试用例。如果种子语料库中不包含特定的输入模式或数据结构，就无法生成探索这些特定条件下应用程序行为的测试用例。然而，基于变异的技术可以在更高的抽象级别上操纵输入数据。它可以通过转换输入数据结构（添加/删除）而不是随机更改单个字节来生成新的测试用例。

一个扩展的想法是基于语法生成测试用例 [6,7] 来为基于生成和变异的模糊测试其提供更好的能力。这种方法利用目标程序接受的输入的形式化语法规范来指导测试用例的生成。通过遵循预定义的语法规则，可以生成在语法上有效的输入，提高测试的效率。与随机生成相比，基于语法的方法能够更好地处理具有复杂输入格式的程序，如编译器、解释器等。此外，将代码片段内联到语法结构中有助于探索代码中的复杂路径，并通过做出使用上下文无关语法无法执行的精确决策来执行复杂操作。但是，它需要对输入语言语法有深入的了解，并且可能需要为复杂或专有的输入语言开发自定义语法。然而，生成的测试用例可能并不总是包含意外或恶意内容，需要进一步变异或生成以增加发现漏洞和错误的可能性。

2 基础变异策略

2.1 AFL 的经典变异策略及其变种

AFL（American Fuzzy Lop）作为一个里程碑式的模糊测试工具，其变异策略具有代表性。AFL 采用了一种混合变异策略，包括确定性变异和随机变异两种方式。确定性变异（Deterministic Strategy）的含义是指基于一组预定义的规则或模式进行的变异操作，这些操作是可以预测的，这在 AFL 中主要体现在以下几个方面，首先是位翻转（Bit Flip），即将输入中的某一位取反。其次是字节翻转（Byte Flip）即将输入中的某一个字节取反；还包括算术变异，即对输入中的整数进行加减操作以及按序插入已知的特殊整数值。而非确定性变异（Non-Deterministic Strategy）则是指基于随机性的变异操作，这些操作是不可预测的，堆叠式位翻转 (Stacked bit flips)，插入操作 (Insertions)，删除操作 (Deletions)，算术运算 (Arithmetics)，测试用例拼接 (Test case splicing) 等都属于这方面。

AFL 同时基于进化算法进行输入队列管理，采用类似自然选择的机制，保留那些能触发新路径的测试用例，通过能量分配机制，优先处理更有潜力的测试用例，定期进行队列精简，去除冗余用例 [8]。然而，AFL 的变异策略也存在一些局限性，而针对 AFL 的哈希冲突问题，CollAFL 提供了更准确的路径覆盖信息 [9]，提出了三种变异引导策略：

- 未触及邻居分支引导：关注当前执行路径的直接可达区域
- 未触及邻居后代引导：扩展搜索范围到可传递

达到的程序区域

- 内存访问引导：特别关注可能导致内存问题的程序位置

这样一个完整的变异决策体系，不仅提高了变异的精确性，还保持了低开销的插桩。除此之外，对于 AFL 主要的优化方向的例子如下：

- 多种优化方式的集成：在 AFL 的基础上集成其他先进 fuzzer 的优化方式，如 AFL++[10]
- 改进变异策略：例如 FairFuzz 针对难抵达的分支进行优化，AFLGo 专注于获取代码的某些部分的覆盖，PerfFuzz 用于查找可能显著减慢程序速度的测试用例，Pythia 预测找到新路径的难度，Angora 使用新的策略进行变异以增加覆盖范围，Neuzz 利用神经网络进行模糊测试 [11]
- 基于 QEMU 和 Unicorn 的改进：对于基于 QEMU 的改进，有 TriforceAFL 和 afl-unicorn 等项目，它们允许在 QEMU 模式下对整个操作系统进行模糊测试 [11]。

3 高级变异技术

3.1 基于程序分析的变异

在模糊测试输入生成的评估中，如何提高路径尤其是复杂深层次的路径以及提高测试用例的生成效率是非常重要的参考指标。AFL 作为里程碑式的工具通过反馈机制实现了简单易用以及相对高效的随机生成机制。但是 AFL 仍然存在着复杂和深层次路径难以探索、针对复杂输入格式时难以生成符合规范的输入和路径探索相对随机等问题。在后期发展的各种框架工具中，基于程序分析的符号执行和污点跟踪技术很好的解决了上述问题。

魔法数字和嵌套校验和也是模糊测试中常见的两大难题。魔法数字通常指的是程序中用于验证输入完整性的特定值，而嵌套校验和则涉及到更复杂的数据完整性验证，如在多层数据结构中计算和比较校验和。这些校验机制的存在显著增加了模糊测试的难度，因为随机生成的输入很难通过这些校验，从而导致测试覆盖率降低。为了克服这些难题，研究人员开发了多种技术，包括基于污点分析的方法和基于符号执行的方法，以提高模糊测试的效率和效果。这些技术通过分析程序的行为和输入数据的流向，自动生成能够通过校验的测试用例，从而深入探索程序的状态空间，发现潜在的安全漏洞。

TaintScope[12] 是一个经典的基于动态污点分析和符号执行技术的模糊测试工具，它在无需访问源代码或精确了解平台指令集的情况下，自动检测输入中的校验字段并精确定位程序中的校验基于完整性检查。该工具的主要贡献在于其能够通过控制流修改来绕过这些校验，从而显著提高模糊测试的覆盖率和效率。TaintScope 不仅能够识别和绕过校验检查，还能自动修复生成输入中的校验值，使其能够触发潜在的漏洞。这一方法有效地结合了污点跟踪和符号执行的优点。

GreyOne[13] 是一种先进的数据流敏感模糊测试技术，其关键技术包括轻量级且准确的污点推断 FTI、基于污点的输入优先级模型以及约束一致性引导的进化策略。FTI 通过监控输入变异时变量值的变化来推断变量的污点状态，这种方法不仅避免了传统污点分析中的劳动密集型工作，而且提高了污点分析的准确性和运行时效率。GreyOne 利用 FTI 提供的污点信息来确定哪些输入字节应该优先变异以及如何变异，以探索更多未触及的代码路径。此外，通过考虑污点变量与未触及分支中预期值的距离（即约束一致性），GreyOne 能够调整其进化方向，更有效地探索难以到达的分支。这些技术的结合使得 GreyOne 在代码覆盖率和漏洞发现方面超越了多种现有的模糊测试工具，展现出其在模糊测试领域的显著优势。

T-Fuzz[14] 是一种基于程序变换的模糊测试方法，它通过移除目标程序中的输入检查来提高代码覆盖率。T-Fuzz 利用覆盖引导的模糊测试生成输入，并在模糊测试无法触发新代码路径时，使用轻量级的动态追踪技术检测输入检查，然后将这些检查从目标程序中移除。这种方法允许之前被检查保护的代码被触发，从而有可能发现潜在的错误。T-Fuzz 面临的挑战包括移除检查可能导致过度近似和误报，以及在变换程序中触发的崩溃输入可能不会在原始程序中触发错误。为了解决这些问题，T-Fuzz 采用了基于符号执行的方法作为辅助的后处理步骤，以过滤掉误报并在原始程序中复现真正的错误。通过变换程序和输入变异，T-Fuzz 能够覆盖更多的代码，并发现比现有技术更多的真实错误。

Angora[15] 是一种基于变异的模糊测试工具，它通过引入一系列关键技术来提高路径覆盖率，有效解决了路径约束问题。Angora 的主要目标是在不使用符号执行的情况下，通过解决路径约束来增加分支覆盖。它采用了可扩展的字节级污点跟踪技术，这种技术通过跟踪输入字节如何流入程序中的每个路径约束，使得 Angora 能够只变异那些影响路径约束的字节，而不是整个输入，从而大幅减少了探索空间。此外，Angora 还采用了基于梯度下降的搜索算法，这种算法在机器学习中非常流行，它避免了昂贵的符号执行，并且能够高效地解决路径

约束。Angora 还进行了类型和形状推断，这使得梯度下降算法能够更有效地搜索，因为它能够识别输入中哪些字节被集体用作单个值，并推断出这些值的类型。最后，Angora 还探索了输入长度，因为某些程序状态只有在输入长度超过某个阈值时才会被探索，而 Angora 能够检测到这一点并适当增加输入长度。

3.2 基于输入与程序状态对应关系的变异

REDQUEEN[16] 是一种新颖的模糊测试方法，它基于输入到状态的对应关系，以提高反馈驱动模糊测试的效率和效果。这种方法特别针对两个常见的模糊测试难题：魔术数字和（嵌套）校验和，这些问题通常需要复杂的程序分析技术，如污点跟踪和符号执行来解决。REDQUEEN 通过在程序执行过程中观察输入与程序状态之间的直接对应关系，实现了一种轻量级且高效的解决方案，无需访问源代码或对平台指令集的精确语义。与前述两篇论文中提到的 Angora 和 T-Fuzz 相比，REDQUEEN 在几个关键方面进行了改进和优化。首先，REDQUEEN 不依赖于符号执行或污点跟踪，这使得它在大型二进制应用程序和未知环境中更容易扩展且高效。其次，REDQUEEN 通过“着色”输入和程序追踪，创建了一种轻量级的污点跟踪近似方法，从而有效地识别和解决魔术数字和校验和问题。此外，REDQUEEN 还能够自动修复通过校验和的输入，而无需人工干预或复杂的符号执行方法，这在 Angora 和 T-Fuzz 中是不具备的。

3.3 基于路径感知的变异

PATA[17] 的主要贡献在于其创新性地将路径感知的污点分析技术融入到模糊测试中，显著提升了对复杂软件漏洞的发现能力。传统的模糊测试技术依赖于随机或基于覆盖率的输入变异，而 PATA 通过精确识别输入数据与程序执行路径中特定变量的依赖关系，实现了对输入数据的智能变异。这种路径感知的污点分析能够区分同一变量在不同执行路径上的多次出现，解决了传统技术在面对循环和多次函数调用时的局限性，从而避免了过污点和欠污点的问题。PATA 首先通过构建代表性变量序列（RVS），记录变量在执行路径上的所有出现及其值，然后通过输入微调并匹配 RVS 来识别影响这些变量值的关键输入字节。这种方法不仅提高了变异的针对性，还增强了模糊测试在探索程序逻辑和发现新漏洞方面的效率。此外，PATA 的先进性还体现在其路径导向的变异策略上。利用路径感知污点分析的结果，PATA 能够精确地识别出对于每个变量出现至关重要的输入字节，并结合变量的特征和值，选择最合适的变异方法来绕过约束，探索程序状态空间中未被覆盖的分支。这种策略使得 PATA 在面对复杂的程序逻辑和深层次的漏洞时，能够更加高效地触发新的执行路径和基本块，从而发现更多的独特路径和基本块覆盖。在多个基准测试和真实世界的项目中，PATA 相较于现有的先进模糊测试工具，展现出了显著的性能提升，不仅在路径和基本块覆盖上超越了其他工具，而且在漏洞发现方面也表现出色，这进一步证明了 PATA 在模糊测试领域的创新性和先进性。

4 定向与智能变异

4.1 定向模糊测试

定向模糊测试（Directed Fuzzing）是一种有目标性的模糊测试方法，其核心思想是针对程序中特定的代码位置或功能进行测试，通过各种策略引导测试过程到达目标位置，集中资源测试可能存在风险的区域。定向模糊测试的主要技术手段主要包含静态分析确定目标位置，符号执行计算路径约束，动态污点分析跟踪数据流以及启发式算法指导种子选择和变异。DGF 是以启发式算法指导种子选择和变异的一个例子，它通过使用模拟退火算法为基础的功率调度，逐渐为接近目标位置的种子分配更多能量，同时减少远离目标的种子的能量，以此实现带有目标性的模糊测试。其主要的变异策略可以概括如下：

- 基于模拟退火的功率调度：AFLGo 使用模拟退火算法动态调整每个种子的能量，优先考虑那些更接近目标位置的种子
- 全局搜索：AFLGo 实现了全局搜索，与 AFL 的局部搜索相比，能够更有效地接近全局最优解，即覆盖最多的目标位置
- 并行搜索：AFLGo 同时搜索所有目标，提高了搜索效率，并允许同时对多个目标进行模糊测试
- 种子距离计算：AFLGo 计算每个种子到目标位置的距离，并根据这个距离来调整种子的能量，使得更接近目标的种子获得更多的测试机会
- 能量分配：AFLGo 根据种子与目标位置的距离和当前时间来分配能量，确保在测试过程中，能量分配能够逐渐向目标位置倾斜

DGF 同时还可以集成现有灰盒模糊测试，与现有的灰盒模糊测试工具（如 AFL）集成，利用其轻量级的程序分析和覆盖信息来指导变异过程 [18]。

4.2 机器学习辅助

机器学习技术在模糊测试领域的应用正在带来革新性的影响。通过对大量测试数据的学习和分析，ML 可以帮助优化测试过程中的多个关键环节。在种子生成和选择方面，机器学习模型能够预测种子的质量和潜在价值，从而更智能地分配测试资源。在变异策略上，ML 可以通过学习历史数据来识别最有效的变异模式，提高发现新执行路径的效率。在路径探索方面，机器学习算法能够分析程序结构特征，预测更可能存在漏洞的代码区域，从而指导测试过程更有针对性地进行探索。深度学习模型可以用于预测程序崩溃的概率，强化学习则可以持续优化测试策略。这些技术的应用大大提升了模糊测试的效率和效果。Learn&Fuzz 的工作展示了机器学习在测试用例生成中的潜力。具体而言，Learn&Fuzz 采用了以下四种变异生成测试的方式 [19]：

- NoSample（无采样：使用学习到的分布来贪婪地预测给定前缀下的最佳字符。这会导致生成的 PDF 对象最有可能是格式良好且一致的。
- Sample（采样）：使用学习到的分布来采样序列中的下一个字符，而不是选择预测最高的字符。这种采样策略能够通过结合模型从训练语料中学习到的各种模式，生成多样化的新 PDF 对象。
- SampleSpace（样本空间）：这种采样策略结合了 Sample 和 NoSample 策略。它只在当前缀序列以空白字符结束时才采样分布来生成下一个字符，而在标记内部（即前缀以非空白字符结束时）使用分布中最佳预测的字符，类似于 NoSample 策略。这种策略预期会生成比 Sample 策略更多格式良好的 PDF 对象，因为采样被限制在仅在空白字符后进行。
- SampleFuzz（采样模糊）：新的算法，用于在采样新对象的同时进行模糊测试。使用学习到的模型生成新的 PDF 对象实例，同时引入异常以测试错误处理代码。SampleFuzz 算法根据学习到的分布决定在何处模糊值。如果在特定时间戳 t，模型对于某个字符 c 的预测概率 p(c) 高于用户提供的阈值 pt，算法将选择一个概率最低的字符 c' 来替换 c。这种修改（模糊）仅在随机硬币投掷的结果 pfuzz 高于输入参数 tfuzz 时发生，允许用户进一步控制模糊字符的概率。
  通过学习已有样本的特征

不过，机器学习辅助的模糊测试也面临一些挑战。首先是获取足够的高质量训练数据比较困难，其次是需要在实时性要求和推理开销之间找到平衡。此外，如何提升模型的泛化能力，以及如何更好地将 ML 技术与传统模糊测试方法融合，都是需要进一步研究的问题。尽管如此，机器学习在模糊测试中的应用前景仍然十分广阔。随着算法和硬件的不断进步，相信这种结合将会产生更多创新性的成果。

致 谢 * 致谢内容.* 致谢

参 考 文 献

[1] Jones, J. (2024, July 19). A 'blue screen of death loop': How a Crowdstrike update crashed Microsoft systems around the world. City AM. Retrieved from https://www.cityam.com

[2] Maria F. Prevezianou. 2021. WannaCry as a Creeping Crisis. Springer International Publishing.

[3] LibFuzzer 2017. libFuzzer—A Library for Coverage-guided Fuzz Testing. Retrieved from https://llvm.org/docs/LibFuzzer.html

[4] Chenyang Lyu, Shouling Ji, Chao Zhang, Yuwei Li, Wei-Han Lee, Yu Song, and Raheem Beyah. 2019. MOPT: Optimized mutation scheduling for fuzzers. In Proceedings of the 28th USENIX Security Symposium, 1949–1966.

[5] Shuitao Gan, Chao Zhang, Peng Chen, Bodong Zhao, Xiaojun Qin, Dong Wu, and Zuoning Chen. 2020. GREYONE: Data flow sensitive fuzzing. In Proceedings of the 29th USENIX Security Symposium, 2577–2594.

[6] Cornelius Aschermann, Tommaso Frassetto, Thorsten Holz, Patrick Jauernig, A.-R. Sadeghi, and Daniel Teuchert. 2019. NAUTILUS: Fishing for deep bugs with grammars. In Proceedings of the Network and Distributed System Security Symposium (NDSS'19). 1–15.

[7] Martin Eberlein, Yannic Noller, Thomas Vogel, and Lars Grunske. 2020. Evolutionary grammar-based fuzzing. In Proceedings of the 12th International Symposium of Search-Based Software Engineering (SSBSE'20). Springer-Verlag, 105–120.

[8] Zhang, S. (n.d.). Technical "whitepaper" for afl-fuzz. Retrieved from http://lcamtuf.coredump.cx/afl/

[9] Gan, S., Zhang, C., Qin, X., Tu, X., Li, K., Pei, Z., and Chen, Z. (2018, May). Collafl: Path sensitive fuzzing. In 2018 IEEE Symposium on Security and Privacy (SP) (pp. 679-696). IEEE.

[10] Fioraldi, A., Maier, D., Eißfeldt, H., and Heuse, M. (2020). AFL++: Combining incremental steps of fuzzing research. In 14th USENIX Workshop on Offensive Technologies (WOOT 20).

[11] Chang, Z. (2019, April 29). AFL 及其相关拓展项目总结. https://zanderchang.github.io/2019/04/29/AFL%E5%8F%8A%E5%85%B6%E7%9B%B8%E5%85%B3%E6%8B%93%E5%B1%95%E9%A1%B9%E7%9B%AE%E6%80%BB%E7%BB%93/

[12] Wang, T., Wei, T., Gu, G., and Zou, W. (2010). TaintScope: A Checksum-Aware Directed Fuzzing Tool for Automatic Software Vulnerability Detection. In Proceedings of the 2010 IEEE Symposium on Security and Privacy (SP'10). https://doi.org/10.1109/SP.2010.37

[13] Gan, S., Zhang, C., Qin, X., Wu, D., Zhao, B., Qin, X., and Chen, Z. (2020). GREYONE: Data Flow Sensitive Fuzzing. In Proceedings of the 29th USENIX Security Symposium (12-14 August 2020). USENIX Association. https://www.usenix.org/conference/usenixsecurity20/presentation/gan

[14] Peng, H., Shoshitaishvili, Y., and Payer, M. (2018). T-Fuzz: Fuzzing by Program Transformation. In 2018 IEEE Symposium on Security and Privacy (SP'18). https://doi.org/10.1109/SP.2018.00056

[15] Chen, P., and Chen, H. (2018). Angora: Efficient Fuzzing by Principled Search. 2018 IEEE Symposium on Security and Privacy, 711-714. DOI: 10.1109/SP.2018.00046x''

[16] Aschermann, C., Schumilo, S., Blazytko, T., Gawlik, R., and Holz, T. (2019). REDQUEEN: Fuzzing with Input-to-State Correspondence. In Network and Distributed System Security (NDSS) Symposium (24-27 February 2019). San Diego, CA, USA. https://dx.doi.org/10.14722/ndss.2019.23371

[17] Liang, J., Wang, M., Zhou, C., Wu, Z., Jiang, Y., Liu, J., Liu, Z., and Sun, J. (2022). PATA: Fuzzing with Path-Aware Taint Analysis. In 2022 IEEE Symposium on Security and Privacy (SP)

[18] Böhme, M., Pham, V.-T., Nguyen, M.-D., and Roychoudhury, A. (2017). Directed Greybox Fuzzing. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17). Dallas, TX, USA: Association for Computing Machinery. https://doi.org/10.1145/3133956.3134020

[19] Godefroid, P., Peleg, H., and Singh, R. (2017). Learn&Fuzz: Machine Learning for Input Fuzzing. In Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE 2017). Urbana-Champaign, IL, USA: IEEE. 978-1-5386-2684-9/17

PENG Bo-Sheng Undergraduate Student. His research interests include machine learning, fuzzing, and software security.

XING Jun-Yang Undergraduate Student. His research interests include machine learning, fuzzing, and software security.

WANG Jing Undergraduate Student. His research interests include machine learning, fuzzing, and software security.

XIAO Jia-Sheng Undergraduate Student. His research interests include machine learning, fuzzing, and software security.

Background

* 论文背景介绍为**英文**，字体为小 5 号 Times New Roman 体 *

论文后面为 400 单词左右的英文背景介绍。介绍的内容包括：

本文研究的问题属于哪一个领域的什么问题。该类问题目前国际上解决到什么程度。

本文将问题解决到什么程度。

课题所属的项目。

项目的意义。

本研究群体以往在这个方向上的研究成果。

本文的成果是解决大课题中的哪一部分，如果涉及 863\973 以及其项目、基金、研究计划，注意这些项目的英文名称应书写正确。