符号执行技术综述

符号执行根据发展状况可以分为传统的符号执行、动态符号执行和选择性符号执行。传统的符号执行并不真实的执行，而是通过解析程序通过符号值模拟执行。动态符号执行结合了真实执行和传统符号执行技术的优点，在真实执行的过程中同时进行符号执行。选择性符号执行可以对程序员感兴趣的部分进行符号执行，其它的部分使用真实值执行。

在过去的40年中，人们对软件测试的符号执行产生了极大的兴趣，因为它能够生成高覆盖率的测试套件并在复杂的软件应用程序中发现深层次的错误。符号执行已经在数十种开发工具中孵化，从而在许多突出的软件可靠性应用中取得重大的实际突破。

许多安全和软件测试应用程序需要检查程序的某些属性是否适用于任何可能的使用场景。例如，识别软件漏洞的工具可能需要排除任何后门的存在，以绕过程序的身份验证。一种方法是使用不同的，可能是随机的输入来测试程序。由于后门可能只针对非常具体的程序工作负载，因此自动探索可能的输入空间至关重要。符号执行通过同时系统地探索许多可能的执行路径，在不需要具体输入的情况下，为问题提供了一个优雅的解决方案。该技术不是采用完全指定的输入值，而是抽象地将它们表示为符号，使用约束解算器来构造可能导致属性冲突的实际实例。

符号执行作为一种生成高覆盖测试套件和在复杂软件应用中发现深层错误的有效技术，引起了人们的广泛关注。符号执行是一种流行的程序分析技术，在70年代中期引入，用于测试某个软件是否可以违反某些属性。本文概述了现代符号执行技术，并从路径探索、约束解决和内存建模方面讨论了它们的挑战。请注意，本文并不打算在这里提供对该领域现有工作的全面调查，而是选择使用简明的示例来说明一些关键挑战和建议的解决方案。

符号执行是一种程序分析技术，它可以通过分析程序来得到让特定代码区域执行的输入。顾名思义，使用符号执行分析一个程序时，该程序会使用符号值作为输入，而非一般执行程序时使用的具体值。在达到目标代码时，分析器可以得到相应的路径约束，然后通过约束求解器来得到可以触发目标代码的具体值。

软件测试中的符号执行主要目标是: 在给定的探索尽可能多的、不同的程序路径。对于每一条程序路径，(1)生成一个具体输入的集合；(2)检查是否存在各种错误，包括断言违规、未捕获异常、安全漏洞和内存损坏。

生成具体测试输入的能力是符号执行的主要优势之一：从测试生成的角度来看，它允许创建高覆盖率的测试套件，而从bug查找的角度来看，它为开发人员提供了触发bug的具体输入，该输入可用于确认和调试打开的错误。生成它的符号执行工具的数据。

此外，请注意，在查找给定程序路径上的错误时，符号执行比传统的动态执行技术更强大，这取决于触发错误的具体输入的可用性。最后，与某些其他程序分析技术不同，符号执行不仅限于查找诸如缓冲区溢出之类的一般性错误，而且可以导致更高级别的程序属性，例如复杂的程序断言。

混合符号执行方法的提出已近十年，其方法本身的新颖、简洁和优雅是吸引 大量研究者关注的重要因素，对此领域的研究远未结束，现在的关注热点是其在 不同场景下的应用，如智能系统¨71和工控系统。未来研究的主要内容大概可以分 为三个方面： (1)混合符号执行以执行路径为契合点，将程序的具体执行和静态符号执行 混合到一处，是将动态和静态分析结和为一体的一次较为成功的尝试。程序分析 和程序验证领域有极为丰富的已有成果，受此启发，是否能对一些经典理论和方 法进行改进，为其赋予动静态结合的新特性[1⋯，是进一步研究方向之一。 (2)从技术的角度来看，现有工具实现主要是针对有限的几种操作系统、语言 和指令系统，实际使用的程序运行于多种不同软硬件平台，如工控系统、车载设备 和移动终端等，而其开发过程中所采用的程序设计语言不仅门类众多，且时常在 高级语言代码中嵌入汇编指令，如多媒体播放器、特效渲染器等。未来技术的发展应该针对更多的软硬件平台，将更多的语言种类包括进来，且结合基于源 代码和二进制的分析方法开发出新的工具，以便能处理多种语言混合的情况。 (3)不同类型的应用中存在的安全漏洞模式也大不相同。如何针对更多应用 领域的程序，总结和整理其经常出现的漏洞类型和安全性约束，是开发新方法或改进上述方法适用范围的有效途径。