浅谈自动化漏洞利用的发展

◆ 刘 凯

摘要：近年来，软件漏洞数量持续增加，严重威胁着计算机信息系统的安全。为了保障信息系统安全，安全分析人员需对软件程序进行手工漏洞挖掘，构造漏洞利用，对漏洞危害作出评估，进而采取相应防护措施。而面对数量繁多的软件漏洞，仅仅依靠人工分析已经难以应对。自动化漏洞利用应运而生，该技术可以显著提高软件安全分析的效率，为人工分析提高良好辅助。论文围绕自动化漏洞利用技术，对该技术进行概述，同时选取典型研究成果，重点对其发展现状进行分析探讨，最后对未来发展趋势进行展望。

关键词：软件安全；漏洞挖掘；漏洞利用；自动化

一、自动化漏洞利用概述

软件漏洞存在于计算机信息系统的软件程序中，是在代码设计或实现过程中产生的一类容易对系统造成危害的缺陷。攻击者往往可以利用软件漏洞，实现对计算机信息系统的攻击，比如近年来爆发的勒索病毒程序 WannaCry，曾利用系统软件中存在的漏洞，大量传播，严重威胁了企业、教育行业的用户数据安全，造成了巨大损失。

自动化漏洞利用是指，在无安全分析人员干预的基础上，依靠程序自动化挖掘软件中隐藏的漏洞，并自动分析漏洞，生成可以获得计算机控制权限的利用数据或代码，最终利用该漏洞实现软件非预期的功能，一般来说，自动化漏洞利用生成的利用代码目标是获取计算机信息系统的运行权限，比如拿到系统 Shell。自动化漏洞利用的过程主要包括以下两个阶段：

一是，漏洞挖掘。漏洞挖掘阶段是对源代码或可执行的二进制程序进行分析，采用静态分析技术（反汇编技术、控制流图分析技术等）和动态分析技术（模糊测试、符号执行、污点分析技术等）结合的方式发现软件中的漏洞、确定漏洞在程序中的位置。

二是，漏洞利用生成。漏洞利用生成则是根据上一阶段获得的漏洞现场信息（内存上下文信息和寄存器信息等），对其加以分析，提取关键寄存器值，构造内存布局，并根据系统内存防护机制采取相应的绕过方法，最终生成能够利用挖掘出的软件漏洞获取目标系统控制权限的代码（exploit）。

二、自动化漏洞利用发展

为了提高软件漏洞安全分析的效率，国内外的学者和研究团队针对自动化漏洞利用开展了大量的研究工作，下面按时间顺序对六项重要的研究成果进行介绍和分析。

2008 年，美国卡内基梅隆大学的 David Brumley 等人首先提出了基于补丁比对的自动化漏洞利用方案 APEG[1]，该方案通过补丁程序寻找到导致程序崩溃的位置，并且根据补丁程序中的崩溃过滤条件，为待测程序生成非预期输入，以此触发程序漏洞。该方案具有较强的实际操作性，但是该方案过度依赖于补丁程序中的过滤条件，所以对不含过滤条件的补丁，无法进行漏洞利用。同时，该方案构造的利用无法完成控制流劫持攻击，对漏洞的利用程度有限。

为了解决漏洞利用过程中对补丁程序依赖的弊端，Avgerinos 等人在 2011 年，第一次系统提出了基于源码的自动化漏洞利用方案 AEG[2]，该方案将自动化漏洞利用转化为形式化的验证问题，使用带路径优先策略的前置条件符号执行技术收集路径约束并使用 STP 求解器进行求解，能够发现软件程序中的栈溢出漏洞和格式化字符串漏洞，可以生成绕过 NX 栈不可执行保护的漏洞利用，对程序实现 return-to-libc 方式的控制流劫持。其缺点是依赖程序源代码进行分析，且主要针对栈溢出等有限范围的漏洞。

2012 年，Cha 等人设计了自动化漏洞利用方案 Mayhem[3]，该方案以二进制程序为分析对象，摆脱了源代码限制，扩大了安全分析的适用范围。该方案在 AEG 方案基础上，使用动态符号执行技术检测漏洞，利用基于可满足性模理论（SMT）的约束求解器求解同时满足路径可达条件和漏洞可利用条件的漏洞利用。通过基于索引的内存模型，缓解了漏洞挖掘过程中的路径爆炸问题，在实际应用中表现较好，但该方案仅完成了对部分库函数的建模，使其无法有效地处理复杂程序。

2013 年，Wang 等人提出自动化漏洞利用方案 PolyAEG[4]，通过污点分析技术动态监控、获取程序中全部可能的控制流劫持点，通过构造不同的跳转指令链和对收集的路径约束进行求解，生成多样化的漏洞利用样本。实验中该方案针对单个控制流劫持漏洞最多生成了 4724 个漏洞利用样本，证明了其具有良好的利用构造能力。但是，该方案生成的漏洞利用在绕过内存保护机制方面存在一定的局限性，同时该方案依赖程序中存在的指令信息，动态生成漏洞利用的能力有所欠缺。

与上述控制流劫持利用不同，2015 年 H.Hu 等人提出了面向数据流分析的自动化漏洞利用方案 FlowStitch[5]，该方案在不改变程序控制流的前提下，针对目标二进制程序中的已知内存错误进行分析，对正常数据流中的关键变量内容进行篡改，生成的漏洞利用可以实现对目标主机的权限提升和信息泄露等功能。通过验证实验，该方案对 8 个真实漏洞程序生成了 19 个漏洞利用样本，可以绕过 NX 栈不可执行保护以及细粒度控制流完整性等系统保护机制。同时，该方案也存在一定局限性，要求目标程序含有已知的内存错误，限制了

其应用范围。

针对程序 crash 现场无法利用的问题，2018 年，Yan Wang 等人提出了自动化漏洞利用方案 Revery[6]。该方案以异常对象的内存布局为导向，采用模糊测试技术来探索漏洞可利用的替代路径，并利用关键内存操作指令定位漏洞点，使用污点分析技术在替代路径中寻找可利用状态。确定拼接点，完成路径拼接后，通过符号执行技术求解路径约束、漏洞利用约束等条件，生成漏洞利用。该方案选取 CTF 信息安全比赛的题目作为测试用例，实验证明该方案可以自动生成针对内存读错误、堆错误的利用样本。同时，该方案的局限性主要体现在，无法自动生成绕过 ASLR 内存地址随机化保护机制的利用样本，无法自动化构造内存布局等。

三、展望

针对自动化漏洞利用研究呈现的发展趋势和面临的问题或挑战，对其进行深入调研和总结，该技术的下一步发展将会围绕以下两方面：

（一）自动化攻防

近年来网络空间斗争态势日趋尖锐和复杂，网络攻击时刻威胁着国民经济安全，网络空间安全已成为国家安全战略的重要组成部分。而软件作为信息系统功能的重要载体，其中存在的漏洞是绝大多数信息安全问题的根源，对抗双方均会围绕软件漏洞展开博弈。

自动化漏洞利用技术的出现和发展正满足了自动化攻防的迫切需求，自动挖掘软件或系统中的 0day 漏洞，自动生成漏洞利用代码，实施网络对抗，同时随着技术的成熟，自动化漏洞利用会结合自动生成补丁，对乙方软件和系统中的漏洞威胁进行修复，实现积极防御，增强信息系统的安全性。正是由于自动化漏洞利用技术的高度适用性，围绕该技术的研究在网络自动化攻防中具有广阔前景。

（二）智能化发展

随着机器学习、深度学习等技术在语音识别、图像处理、自然语言处理等领域的成熟和应用，安全研究人员开始采用逻辑回归、加权前缀树等机器学习算法进行日志分析、垃圾邮件分类处理等基础工作，以提高处理准确性。而在当前自动化研究中，漏洞挖掘的精度和效率同样有待提升，主要体现在模糊测试输入用例的生成具有盲目性，符号执行存在路径爆炸、约束求解复杂等问题。

因此，在今后的软件安全分析研究中，一个可能的发展趋势是自动化漏洞利用会紧密结合机器学习等技术，探索漏洞挖掘与漏洞利用生成的新方式，尤其是在自动化漏洞利用中的漏洞挖掘阶段，构造路径覆盖率高或具有脆弱性导向的测试输入，能提高漏洞挖掘的效率和针对性，利用神经网络模型可以对海量待测程序进行分析、学习，并利用生成模型指导生成更高质量的测试输入样本，缓解盲目性、路径爆炸问题，结合梯度下降算法处理约束求解的复杂性问题。总之，上述机器学习技术或算法的应用，会推动自动化漏洞利用向着智能化方向发展，进一步提高软件安全分析的效率，有利于更好地保障计算机信息系统的安全。

四、结语

经过上述对自动化漏洞利用研究发展现状的分析，可以看出国内外研究者针对自动化漏洞利用研究已取得一些成果，可以实现从发现漏洞、分析漏洞，到设计漏洞利用方式，再到生成漏洞利用的全过程自动化，目前可以检测软件程序中的常见漏洞（栈溢出漏洞、格式化字符串漏洞、整数溢出漏洞等），并针对漏洞生成较多样化的利用代码，已经实现了从依赖程序源代码到独立分析二进制程序的跨越。安全分析人员正是依据生成的漏洞利用对软件和系统造成的危害程度，对漏洞威胁进行评估，进而采取相应级别的防护措施，保障系统安全。因此，目前研究在一定程度上有助于程序漏洞检测和漏洞威胁评估，可以为人工安全分析提供良好的辅助。

但是，目前自动化漏洞利用技术仅限于分析处理小型软件或代码片段。软件复杂性的增加、漏洞类型和数量的发展变化以及多种内存保护机制的部署，均给漏洞挖掘与漏洞利用生成带来了挑战，自动化漏洞利用离实际应用还存在一定距离，该方向研究仍将处于不断探索和发展中。

参考文献

[1] Brumley D, Poosankam P, Song D, et al. Automatic patch-based exploit generation is possible: Techniques and implications [C]//2008 IEEE Symposium on Security and Privacy. IEEE, 2008: 143-157.

[2] Avgerinos, Thanassis, Cha, Sang Kil, Rebert, Alexandre, et al. Automatic exploit generation[J]. Communications of the ACM, 57(2):74-84.

[3] Cha S K, Avgerinos T, Rebert A, et al. Unleashing mayhem on binary code[C]//2012 IEEE Symposium on Security and Privacy. IEEE, 2012: 380-394.

[4] Minghua Wang, Purui Su, Qi Li, et al. Automatic Polymorphic Exploit Generation for Software Vulnerabilities [C]// International Conference on Security and Privacy in Communication Systems. Springer International Publishing, 2013.

[5] Hu H, Chua Z L, Adrian S, et al. Automatic generation of data-oriented exploits[C]//24th USENIX Security Symposium (USENIX Security 15). 2015: 177-192.

[6] Wang Y, Zhang C, Xiang X, et al. Revery: From proof-of-concept to exploitable[C] // Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. 2018: 1914-1927.

（作者单位：中国海洋大学信息科学与工程学院）