以太坊智能合约漏洞及检测技术综述

杨　晶　朱卫星*　杨忠举　王梅娟　朱益威

（陆军工程大学　江苏　南京　210007）

【摘　要】智能合约是区块链的核心技术之一，拥有去中心化，不可篡改等特点。然而，也正因其特点，导致智能合约的安全性问题频发，给区块链生态以及现实经济带来了巨大的损失。本文首先从程序语言、EVM 和区块链三个层面分析典型的以太坊智能合约漏洞，接着针对上述漏洞综述模糊测试、符号执行、形式化验证、污点分析以及深度学习五种漏洞检测技术，最后阐述未来为保障智能合约安全运行所需要的研究工作。本文研究以太坊智能合约漏洞及检测技术，有助于为相关应用者做出现实参考以及提高以太坊智能合约的安全可靠性。

【关键词】区块链；以太坊；智能合约；漏洞分析；漏洞检测

智能合约[1]与传统合约不同，通过数字形式掌控现实资产，是提供验证并执行的计算机交易协议，一旦满足智能合约的执行条件，协议会自动执行且无需第三方参与。智能合约概念于二十世纪末被提出，由于当时缺乏可信的执行环境无法在现实行业得到有效应用，直至区块链的诞生为其创造了一个理想的执行环境。就目前而言，智能合约在领域应用过程中带来重大价值的同时也存在潜在的安全漏洞风险，极易招致各方面的恶意攻击，进而产生安全问题，因此并未得到普及。例如 2022 年，跨链通讯协议 Nomad 遭到了大规模的黑客攻击，项目方损失达 1.9 亿美元；2021 年区块链安全生态报告显示，80％ DAPP 安全事故缘于智能合约漏洞，至少有 76 亿美元加密资产在这些安全事故中损失；2020 年智能合约游戏 FarmEOS 造成近百万美元损失等等。以太坊是较为流行的区块链平台，其智能合约漏洞及其检测技术成为相关学者近年的研究热点。

一、以太坊智能合约漏洞

智能合约是依在附区块链上的一种程序，与其他程序不同，智能合约有很多特殊性能，也正因此给智能合约带来了一些独特的威胁攻击方式。这些攻击方式，有的源于智能合约自身的，有的源于开发人员，以下从三个方面综述十种典型的以太坊智能合约漏洞。

（一）程序语言层面

1. 重入漏洞

按顺序执行和原子性是合约程序具有的特性。按顺序执行是指程序的执行是按次序逐个执行；原子性即程序可拆分为小段。因此，在一段程序函数中，尤其以非递归函数为代表，会在一段程序指令执行结束以后进行下一段程序的执行。同时，以太坊智能合约有外部调用的特点，而调用的函数具有重入性，即多个函数同时调用同一个函数，这就给了攻击恶意合约可乘之机。例如，合约经常出现给外部账户转账的行为，进行此操作需外部调用 fallback()，当恶意合约重复执行被调用合约的代码时，可能会产生被调用合约的以太币损失。为避免此漏洞，可以替换易出现漏洞的函数，例如可以使用 send()、transfer()；除此以外，还可以从代码根源入手，在编写时尽量避免写入外部调用的程序。

2. 溢出漏洞

程序编写完成后，程序会存放于存储空间如内存中，但由于字节长度，内存空间等大小限制，致使变量在被分配空间时有一定的长度限制，若运算结果超过其被分配的空间范围，就会发生整数溢出的情况。用户使用智能合约时，往往会忽视运算结果的安全检测，导致整数溢出概率增大，其结果也可能导致智能合约运行异常。整数溢出曾造成巨大的经济损失，如美链公司发布的 token 上线交易时出现漏洞，代币价值瞬间归为零。由于其不可篡改性，即使被攻击者发现问题也无法解决。为避免此漏洞，可以在用户层面对运算结果进行验证并通过技术人员层面使用 SafeMath 安全库[2]处理算术逻辑。

作者简介：杨晶（1999-），女，河南漯河，本科，研究方向：指挥信息系统工程；通讯作者：朱卫星（1978-），男，浙江长兴，陆军工程大学指挥控制工程学院副教授，博士，研究方向：军事需求、软件工程、作战实验与大数据。

3. 拒绝服务漏洞

智能合约系统具有一定的逻辑顺序性，其拒绝服务问题是指当合约中原来的逻辑顺序被打破，以太坊网络上的信息耗尽，导致合约在一段时间内不能顺利执行或正常进行。原因大致有三种：第一种是攻击者使用智能合约的编码方式，入侵智能合约系统；第二种是由于在编程中某个变量掌握了每个代码的所有权，控制权垄断；第三种是攻击者通过一系列操控使 gas 耗尽，合约中的交易中断。为避免此漏洞，可以严格限制 selfdestruct() 的权限，设置完善合理的访问控制策略。当目标地址是合约时，充分考虑合约的特性也尤为重要。

4. 未校验返回值漏洞

在智能合约中，若不检查代码的执行情况，可能会导致未校验返回值漏洞的发生。智能合约往往会调用其他合约程序段，当出现异常时，合约没有统一的方法处理。如转账操作中 transfer() 可提示异常，而低级函数如 send() 没有此功能，而是继续执行剩余代码或只返回 false。因此，不对函数的执行结果校验可能导致程序执行的控制流与预期结果相悖，给攻击者可乘之机。为避免此漏洞，可减少使用底层调用函数，或在每次调用函数后进行返回值检查，及时处理问题。

5. 类型匹配异常漏洞

类型匹配异常通常分为两种，一种是函数自身匹配异常，如在智能合约中变量名与变量不匹配，而在执行过程中不显示异常；另一种是调用函数不匹配，若智能合约与其调用合约中的函数或参数不匹配，则程序会默认执行 fallback()，异常匹配会给程序的执行带来极大的隐患。为避免此漏洞，开发人员需加大对程序的人工审查力度。

6.delegatecall 漏洞

delegatecall() 是一种特殊调用，允许被调函数在原有的上下文环境中运行，当调用的函数是输入函数时，攻击者可输入任意值控制程序进程；当调用的函数中存在 suicide 等恶意函数时可以自毁合约，典型实例为以太币冻结。因此，合约必须谨慎调用 delegatecall()。

（二）EVM 层面

1. 短地址漏洞

在 EVM 层面中，不同合约中交易信息的输入有不同的字节位数限制，若输入的信息字节长度小于该类规范中规定的字节长度，以太坊会在末尾自动补 0，直至达到规定的位数，这相当于将原来的信息扩大了一定的倍数。攻击者能够利用此漏洞在输入信息时不输入规定的位数，利用其自动补其的功能扩大原有的金额，导致以太币大量损失。为避免此漏洞，严格校验用户输入的地址，以及在前端禁止掉用恶意的短地址。

2.Tx.Origin 漏洞

Tx.Origin 是 solidity 中的全局变量，返回值为授权用户的地址。若智能合约使用该变量，则攻击者可通过该变量从根源处着手获取授权账户的信息或调用原授权用户的内部函数以达到自己的需求，对合约的安全性造成威胁。为避免此漏洞，可以使用 require() 以防止一些中间合约调用本合约。除此之外，还可以在开发者层面尽量避免使用 tx.origin，而使用 msg.sender() 替代。

（三）区块链层面

1. 交易顺序漏洞

部分智能合约有一定的执行次序，一旦执行次序出现问题合约可能不会正常执行，而决定交易次序的是在区块链网站上的矿工（即区块链网络中的节点或用户），不难理解若矿工故意更改合约的交易次序将会导致合约错误执行。为避免此漏洞，利用哈希存储提交信息，通过提交的哈希值信息判断两个交易的哈希值是否匹配。

2. 时间戳漏洞

智能合约的执行通过当前区块时间戳决定，恶意矿工能够改变时间戳进而改变合约的执行。恶意矿工可绕过时间戳的设计限制在一定取值范围内操纵，最终影响合约正常进行。为避免此漏洞，可以使用更改合约状态的方法执行合约。

二、以太坊智能合约漏洞检测技术分类

上一小节论述了十种典型的以太坊智能合约漏洞，事实上以太坊智能合约漏洞远不止此十种，使用检测工具进行智能合约漏洞检测尤为重要。模糊测试、符号执行、形式化验证、污点分析、深度学习这五种漏洞检测技术都取得了很好的效果，以下是此五种检测技术的方法及特点。

（一）模糊测试

模糊测试指使用大量样例在软件上执行，监视其异常情况以找到软件缺陷的一门应用研究技术。这种技术较为便捷，利用生成的测试样例在合约上实施，检查合约中是否存在漏洞。生成样例是整个过程的关键，一般包括基于生成和基于变异两个方法。基于生成是指利用某种方法或技术进行基础合成，适用于待测系统要求较明确以及技术条件严格

的情形；基于变异指在原始输入的基础上依据程序反馈结果进行对原始系统的更改，从而通过突变产生新样例。ContractFuzzer、sFuzz 和 AFL 是三种典型的基于模糊的以太坊智能合约漏洞检测方法。

（二）符号执行

符号执行指在程序中将变量符号化作为输入进而执行程序的一种方式。在一般的程序中输入为一个定值，因而在程序执行过程中有一个确定的路径选择。而利用符号执行的方式因其输入是符号化无法确定其具体值，故遇到条件判断时会进行所有的路径并且记录所有有解的分支。因此，符号执行的一个优点则是对程序的分析更加的全面。符号执行适用于代码路径少，长度短的代码，智能合约也恰恰符合这一特性。根据符号执行的所产生的典型智能合约漏洞检测工具有 Oynete[3]、Maian[4]、Secruify[5]。

（三）形式化验证

形式化验证可以抽象为一种建模形式，是指将智能合约中的代码用逻辑语言的形式表达，经过一定的逻辑推理和证明，检测智能合约中的功能规范和源代码等安全文档是否符合预期结果。此处列举一种基于状态空间的智能合约形式化验证系统，共分为七个模块：代码编译，基本块划分，控制流图生成，逻辑规则，源代码形式化，执行与验证。该种方法可减少人工参与度，提高了自动化能力和效率，且可以反推导出漏洞类型的标志作为状态空间分析的输入，提高模型执行效率。典型的形式化验证的方法有 SPIN，F* framework[6]。

（四）污点分析

污点分析指对于来自非正规渠道的数据能否不经无害处理直接进入污点汇聚点。若可以直接进入则证明系统中产生了危险数据操作，反之证明系统是安全的。识别污点源、污点传播分析和无害处理是污点分析的三个过程。首先识别并标记污点数据，接着被标记的污点源经过显示流分析或隐式流分析的方式，探测污点如何传播到其实现自身功能的语句中，最后经无害处理模块使污点数据去除其危害性。

（五）深度学习

深度学习是一种模式分析方法的统称，也是人工智能领域的一个新的研究方向。基于深度学习的以太坊智能合约漏洞检测技术可以分为预处理，模型训练，漏洞检测三个步骤，使用典型的实例对其阐述。预处理过程即反编译和反汇编过程，由于智能合约操作码难以分析，因此需要使用反编译工具将字节码转变为汇编代码，并将汇编代码汇编成高级语言代码。模型训练过程首先通过将智能合约表征为控制流程图方式获取合约的主要功能点和函数调用，接着提取该功能点将需要进行漏洞检测的漏洞类型和漏洞位置输入到深度学习算法中，最后将其组合成智能合约程序切片。通过 One-shot、word2vec 等分词及向量化表示方法将智能合约程序切片转化为向量形式作为深度学习模型的输入，进而通过构建合适的模型进行训练以及测试。深度学习法能够较好的利用大数据的优势，依据大规模训练数据逐步提高检测性能；除此之外，还能够从数据中直接提取特征减小错误率。

三、总结与展望

智能合约安全问题是近几年的研究热点，目前 90% 左右的智能合约都存在漏洞，严重制约了区块链技术的发展。本文从三个层面梳理了 10 种典型的以太坊智能合约漏洞并提出了相关安全防范方法，接着，由漏洞问题引出 5 种典型的以太坊智能合约漏洞检测技术以及相应检测工具的讨论。就目前而言，以太坊智能合约漏洞检测工具普遍存在检测效率不高的问题，因此，如何设计安全系数、自动化以及兼容性更高的检测工具从而减少性能带来的损失尤为重要。与此同时，还需要更新以太坊智能合约漏洞库使其更加完整，创建新型漏洞挖掘工具并建立的评价方法以保障未来以太坊智能合约安全运行。

参考文献：

[1]Szabo.N.Formalizing and Securing Relationships on Public Networks[J].First Monday,1997,2(9):1-21.

[2]SafeMath. 2020. https://docs.statechannels.org/contract-api/natspec/SafeMath.

[3]Luu L, Chu D H, Olickel , et al. Making smart contracts smarter. In Proc. of the 2016 ACM SIGSAC Conference on Computer and Communications Security[J]. 2016.

[4]Nikolić I, Kolluri A, Sergey I, et al. Finding the greedy, prodigal, and suicidal contracts at scale[C]// Proceedings of the 34th annual computer security applications conference. 2018: 653-663.

[5]Tsankov P, Dan A, Drachsler-Cohen D, et al. Securify: Practical security analysis of smart contracts[C]// Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. 2018: 67-82.

[6]Grishchenko I, Maffei M, Schneidewind C. A semantic framework for the security analysis of ethereum smart contracts[C]//International Conference on Principles of Security and Trust. Springer, Cham, 2018: 243-269.