针对 EOS、NEO 等大公链平台的多个双花攻击漏洞的案例，360 区块链实验室总结出了多种造成数字货币双花攻击的多种原因，并提出了一种通用的安全减缓措施。各种大公链项目实际上都产生过能够产生双花攻击之类的严重安全问题，盗取加密货币对黑客来讲不是难事。

## 1 工作量证明和双花攻击

**工作量证明机制的原理**：网络中每一个区块都包含当前网络中的交易和上一个区块的区块头哈希。新区块产生，其区块头哈希必须满足工作量证明条件（需要进行大量的哈希计算）。整个网络将满足工作量证明的哈希链连接起来，从而形成区块链。除非攻击者重新完成全部的工作量证明，否则形成的交易记录将不可更改。最长的区块链不仅将作为被观察到的交易序列的证明，而且被看做是来自算力最大的群体的共识。

只要整个网络中大多数算力都没有打算合作起来对全网进行攻击，那么诚实的节点将会生成最长的、超过攻击者的链条，从而实现对双花攻击的抵抗。

**双花攻击**实际上是一个结果。如果一个攻击者 A 将同一个比特币同时支付给 B 和 C 两个用户，并且 B 和 C 两个用户都认可了这笔交易。那么我们说 A 将该比特币花了两次，A 实现了一次双花攻击。

针对工作量证明机制的双花攻击实际上有多种形式，包括**芬妮攻击、竞争攻击、Vector76** 攻击等。这些攻击实际上也得到了充分的关注和讨论，本文中不做赘述。实际上，实用的数字货币双花攻击还有很多种其他形式。下文中，我们将通过多个我们发现的多个安全漏洞，讨论多种数字货币双花攻击的多种原因，并提出一种高效减的缓措施。

## 2 双花攻击的新分类

智能合约平台，本质上是要在全网共享一个账本。这可以看成是一个分布式状态机复制问题。

在全网状态机复制的过程中，如果一旦因为某些原因产生了全网状态不一致，则我们可以认为全网产生了一个分叉。分叉被攻击者利用，可进一步实现双花攻击。

**双花攻击漏洞分成 3 类：**

1 验证不严格造成的双花攻击。

2 状态机 State\_n × Tx\_{n+1}→State\_{n+1}不一致执行造成的双花攻击。

3 共识机制造成的双花攻击。

状态机不一致执行造成的双花攻击，主要是由于智能合约虚拟机因为各种原因导致直接结果不一致，从而在整个网络中创造分叉，造成双花攻击。

共识机制漏洞可能产生整个网络的分叉，从而进一步造成双花攻击。人们常说的 51% 攻击，实际上就是 PoW 共识机制的分叉漏洞。

## 3 验证不严格造成的双花攻击

验证不严格造成的双花攻击，主要原因在于具体实现逻辑校验问题。这里我们介绍两个关于区块与交易绑定时校验不严格，从而产生双花攻击的漏洞。

如果攻击者能够打包交易与区块的绑定，则攻击者能通过造成全网的分叉从而实现双花攻击。

### 3.1 NEO虚拟机GetInvocationScript 双花攻击漏洞

区块链项目中，一个交易一般是由未签名的部分（UnsignedTx，交易要执行的内容）和签名的部分（交易的 witness）构成的。

攻击者利用这个性质，可以对 NEO 智能合约上的所有代币资产进行双花攻击

|  |
| --- |
| **其具体攻击场景如下：** |
| 步骤 1：攻击者构造智能合约交易 Tx\_1（未签名内容 UnsignedTx\_1, 验证脚本为 VerficationScript\_1）。在 UnsignedTx\_1 的合约执行中，合约判断自己的 VerficationScript 是否为 VerficationScript\_1。如果为 VerficationScript\_1，择发送代币给 A 用户。如果 VerficationScript 为空，则发送代币给 B 用户。  步骤 2：Tx\_1 被打包到区块 Block\_1 中。  步骤 3: 攻击者收到 Block\_1 后，将 Tx\_1 替换成 Tx\_2(Tx\_1 具有与 Tx\_1 相同的未签名内容 UnsignedTx\_1，但验证脚本为空) 从而形成 Block\_2。攻击者将 Block\_1 发送给 A 用户，将 Block\_2 发送给 B 用户。  步骤 4：当 A 用户收到 Block\_1 时，发现自己收到攻击者发送的代币。当 B 用户收到 Block\_2 时，也会发现自己收到了攻击者发送的代币。双花攻击完成。  可见，该漏洞的利用门槛非常低，且可以对 NEO 智能合约上的所有代币资产进行双花攻击。危害非常严重。 |

### 3.2 NEO MerlkeTree 绑定绕过造成交易双花攻击漏洞

智能合约交易与区块的绑定，通常通过 MerkleTree 来完成。如果攻击者能绕过该绑定，则能实现对任意交易的双花。

攻击者可以实现对任意 NEO 资产的双花攻击。其具体攻击场景如下：

**步骤 1**：假设正常的一个合法 Block\_1，包含的交易列表为【Tx\_1 Tx\_2 … Tx\_n】。攻击者收到 Block\_1 后，将交易列表替换为【Tx\_1 Tx\_2 … Tx\_n Tx\_】，形成 Block\_2。然后将 Block\_2 发布到网络中去。

**步骤 2**：一个普通节点收到 Block\_2 后，会对 Block\_2 的合法性进行校验。然而因为【Tx\_1 Tx\_2 … Tx\_n Tx\_】与【Tx\_1 Tx\_2 … Tx\_n】具有相同的 MerkleRoot。所以 Block\_2 能够通过区块合法性校验，从而进如区块持久化流程。NEO 本地取消了普通节点对合法区块中交易的验证（信任几个共识节点）。则 Tx\_n 交易可以被普通节点执行两次，双花攻击执行成功。

该漏洞的利用门槛非常低，且可以对 NEO 上的所有资产进行双花攻击。危害非常严重。

## 4 虚拟机不一致性执行

智能合约平台共识机制，本质上是将所有的交易【Tx\_1 Tx\_2 ……. Tx\_n】按顺序作用到初始 State\_0 上，使全网始终保持相同的状态。

### 4.1 EOS 虚拟机内存破坏 RCE 漏洞:

其本质而言，是在 State\_n × Tx\_{n+1} → State\_{n+1}过程中。攻击者能让 EOS 虚拟机完全脱离原本执行路径，执行任意指令，自然可以完成双花攻击。

攻击流程如下：

步骤 1：攻击者构造能够实现 RCE 的恶意智能合约，并将该合约发布到 EOS 网络中。

步骤 2：EOS 超级节点解析到该合约后，触发漏洞，执行攻击者自定义的任意指令。

步骤 3：攻击者实现双花攻击。

该漏洞的危害非常严重，且是第一次智能合约平台受到远程代码执行攻击事件。

### 4.2 EOS 虚拟机内存未初始化造成双花攻击：

在智能合约虚拟机中，内存未初始化漏洞有更直接的利用方式，可以直接造成双花攻击。

攻击者可以构造恶意合约，实现对 EOS 上任意合约资产的双花攻击。

**攻击流程如下：**

步骤 1： 攻击者构造恶意智能合约。合约中通过 grow\_memory 获得一块新的内存地址。

步骤 2：合约中读取该地址中的某个 bit 内容（此时该 bit 可能为 0，也可能为 1，依赖于合约执行机器的状态）。

步骤 3：合约判断该 bit 的内容，如果为 1 则发送代币给 A 用户，如果为 0 则发送代币给 B 用户，从而实现双花攻击。

### 4.3 EOS 虚拟机内存越界读造成双花攻击:

在传统的内存破坏中，内存越界读漏洞主要将会导致信息泄露，从而辅助我们绕过如 ASLR 之类的现代二进制程序的缓解措施，进一步与其他漏洞一起实现攻击。然而在智能合约虚拟机中，内存越界读漏洞有更直接的利用方式，可以直接造成双花攻击。

下面为一个我们发现的 EOS 内存越界读漏洞，我们可以利用该漏洞实现双花攻击。

攻击者可以利用该漏洞实现双花攻击。

**其攻击过程如下：**

**步骤 1：**攻击者构造恶意智能合约。合约中利用内存越界读漏洞，读取超越 WASM 内存基址的某个 bit）此时该 bit 可能为 0，也可能为 1，依赖于合约执行机器的状态）。

**步骤 2：**合约判断该 bit 的内容，如果为 1 则发送代币给 A 用户，如果为 0 则发送代币给 B 用户，从而实现双花攻击。

### 4.4 标准函数实现不一致造成双花攻击：

### 4.5 版本实现不一致造成双花攻击：

### 4.6 其他问题不一致性问题

## 5 共识机制造成的双花攻击

### 5.1 ONT vBFT VRF 随机数绕过漏洞

### 5.2 NEO dBFT 共识分叉

## 6 一种针对虚拟机执行不一致双花问题的高效减缓措施