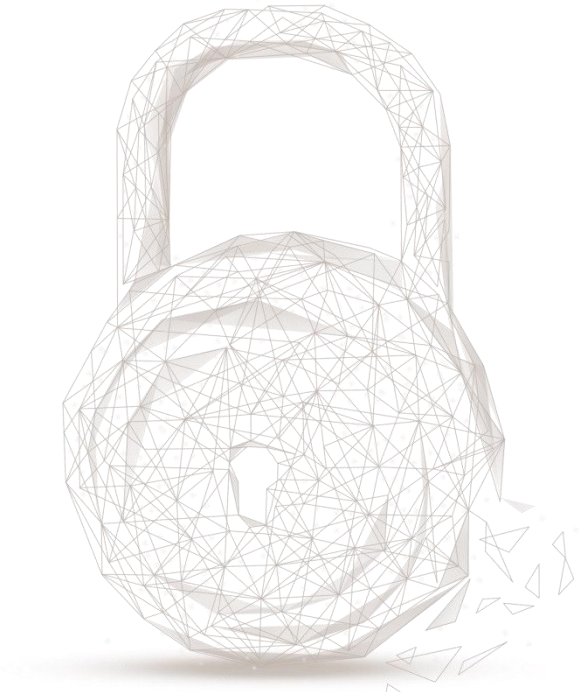
成都链安
BEOSIN

智能合约安全审计报告



审计编号：202104081552

报告查询名称：Apple

审计项目合约地址：

AppleSwapGov	0xcE0593AdfCb45EE0D9558f102dff2969a891DAE2
PairFeeDistribution	0x63E692467bFED95fcef58a1078D95E5A380A97B3
AppleSwapPair	0x245ad2d2d3f0810AA780eC000033d87C1Db4a1a9
AppleSwapFactory	0x2bCeb8B34715e168f2f6136Bb9Cd82B942f6cD13
AppleSwapRouter	0x1868202acdC5d6a3C38d52E72445599D5B7dd92f
AppleSwapBuyback	0xc840Cba0cA6dEe2Ce231DbBf0ADB79f971A0f994
AppleExchange	0x5964aC18376572e729EBe95eBe8982c34Af6F6F7
AppleRewardPool	0xf1De3918a4feE210daB101b131C3434fe98981aF
AppleDAOPool	0xEdcC81f3094308869082D790cAD79A30A0233253

合约审计开始日期：2021.04.01

合约审计完成日期：2021.04.08

审计结果：通过

审计团队：成都链安科技有限公司

审计类型及结果：

序号	审计类型	审计子项	审计结果
1	代码规范审计	编译器版本安全审计	通过
		弃用项审计	通过
		冗余代码审计	通过
		require/assert 使用审计	通过
		gas 消耗审计	通过
2	通用漏洞审计	整型溢出审计	通过
		重入攻击审计	通过
		伪随机数生成审计	通过
		交易顺序依赖审计	通过
		拒绝服务攻击审计	通过
		函数调用权限审计	通过
		call/delegatecall 安全审计	通过
		返回值安全审计	通过
		tx.origin 使用安全审计	通过
		重放攻击审计	通过

		变量覆盖审计	通过
3	业务审计	业务逻辑审计	通过
		业务实现审计	通过

备注：审计意见及建议请见代码注释。

免责声明：本次审计仅针对本报告载明的审计类型及结果表中给定的审计类型范围进行审计，其他未知安全漏洞不在本次审计责任范围之内。成都链安科技仅根据本报告出具前已经存在或发生的攻击或漏洞出具本报告，对于出具以后存在或发生的新的攻击或漏洞，成都链安科技无法判断其对智能合约安全状况可能的影响，亦不对此承担责任。本报告所作的安全审计分析及其他内容，仅基于合约提供者在本报告出具前已向成都链安科技提供的文件和资料，且该部分文件和资料不存在任何缺失、被篡改、删减或隐瞒的前提下作出的；如提供的文件和资料存在信息缺失、被篡改、删减、隐瞒或反映的情况与实际情况不符等情况或提供文件和资料在本报告出具后发生任何变动的，成都链安科技对由此而导致的损失和不利影响不承担任何责任。成都链安科技出具的本审计报告系根据合约提供者提供的文件和资料依靠成都链安科技现掌握的技术而作出的，由于任何机构均存在技术的局限性，成都链安科技作出的本审计报告仍存在无法完整检测出全部风险的可能性，成都链安科技对由此产生的损失不承担任何责任。

本声明最终解释权归成都链安科技所有。

审计结果说明：

本公司采用形式化验证、静态分析、动态分析、典型案例测试和人工审核的方式对Apple项目智能合约代码规范性、安全性以及业务逻辑三个方面进行多维度全面的安全审计。**经审计，Apple项目智能合约通过所有检测项，合约审计结果为通过，合约可正常使用。**以下为本合约详细审计信息。

代码规范审计

1. 编译器版本安全审计

- **安全建议：**建议固定编译器版本。
- **审计结果：通过**

2. 弃用项审计

Solidity智能合约开发语言处于快速迭代中，部分关键字已被新版本的编译器弃用，如throw、years等，为了消除其可能导致的隐患，合约开发者不应该使用当前编译器版本已弃用的关键字。

- **安全建议：无**
- **审计结果：通过**

3. 冗余代码审计

智能合约中的冗余代码会降低代码可读性，并可能需要消耗更多的gas用于合约部署，建议消除冗余代码。在Apple项目中存在少量代码冗余。

- **安全建议：建议删除冗余代码。**
- **修复结果：已修复。**
- **审计结果：通过**

4. require/assert 使用审计

Solidity使用状态恢复异常来处理错误。这种机制将会撤消对当前调用(及其所有子调用)中的状态所做的所有更改，并向调用者标记错误。函数assert和require可用于检查条件并在条件不满足时抛出异常。assert函数只能用于测试内部错误，并检查非变量。require函数用于确认条件有效性，例如输入变量，或合约状态变量是否满足条件，或验证外部合约调用的返回值。

- **安全建议：无**
- **审计结果：通过**

5. gas 消耗审计

以太坊虚拟机执行合约代码需要消耗gas，当gas不足时，代码执行会抛出out of gas异常，并撤销所有状态变更。合约开发者需要控制代码的gas消耗，避免因为gas不足导致函数执行一直失败。

- **安全建议：无**
- **审计结果：通过**

通用漏洞审计

1. 整型溢出审计

整型溢出是很多语言都存在的安全问题，它们在智能合约中尤其危险。Solidity最多能处理256位的数字(2**256-1)，最大数字增加1会溢出得到0。同样，当数字为uint类型时，0减去1会下溢得到最大数字值。溢出情况会导致不正确的结果，特别是如果其可能的结果未被预期，可能会影响程序的可靠性和安全性。

- **安全建议：无**
- **审计结果：通过**

2. 重入攻击审计

重入漏洞是最典型的以太坊智能合约漏洞。该漏洞原因是Solidity中的call.value()函数在被用来发送ETH的时候会消耗它接收到的所有gas，当调用call.value()函数发送TRX的逻辑顺序存在错误时，就会存在重入攻击的风险。

- **安全建议：无**
- **审计结果：通过**

3. 伪随机数生成审计

智能合约中可能会使用到随机数，在solidity下常见的是用block区块信息作为随机因子生成，但是这样使用是不安全的，区块信息是可以被矿工控制或被攻击者在交易时获取到，这类随机数在一定程度上是可预测或可碰撞的，比较典型的例子就是fomo3d的airdrop随机数可以被碰撞。

- **安全建议：无**
- **审计结果：通过**

4. 交易顺序依赖审计

在以太坊的交易打包执行过程中，面对相同难度的交易时，矿工往往会选择gas费用高的优先打包，因此用户可以指定更高的gas费用，使自己的交易优先被打包执行。

- **安全建议：无**
- **审计结果：通过**

5. 拒绝服务攻击审计

拒绝服务攻击，即Denial of Service，可以使目标无法提供正常的服务。在以太坊智能合约中也会存在此类问题，由于智能合约的不可更改性，该类攻击可能使得合约永远无法恢复正常工作状态。导致智能合约拒绝服务的原因有很多种，包括在作为交易接收方时的恶意revert、代码设计缺陷导致gas耗尽等等。

- **安全建议：无**
- **审计结果：通过**

6. 函数调用权限审计

智能合约如果存在高权限功能，如：铸币、自毁、change owner等，需要对函数调用做权限限制，避免权限泄露导致的安全问题。

- **安全建议：无**
- **审计结果：通过**

7. call/delegatecall安全审计

Solidity中提供了call/delegatecall函数来进行函数调用，如果使用不当，会造成call注入漏洞，例如call的参数如果可控，则可以控制本合约进行越权操作或调用其他合约的危险函数。

- **安全建议：无**
- **审计结果：通过**

8. 返回值安全审计

在Solidity中存在transfer()、send()、call.value()等方法中，transfer转账失败交易会回滚，而send和call.value转账失败会return false，如果未对返回做正确判断，则可能会执行到未预期的逻辑;另外在ERC20 Token的transfer/transferFrom功能实现中，也要避免转账失败return false的情况，以免造成假充值漏洞。

- **安全建议：无**
- **审计结果：通过**

9. tx.origin使用安全审计

在智能合约的复杂调用中，tx.origin表示交易的初始创建者地址，如果使用tx.origin进行权限判断，可能会出现错误;另外，如果合约需要判断调用方是否为合约地址时则需要使用tx.origin，不能使用extcodesize。

- **安全建议：无**
- **审计结果：通过**

10. 重放攻击审计

重放攻击是指如果两份合约使用了相同的代码实现，并且身份鉴权在传参中，当用户在向一份合约中执行一笔交易，交易信息可以被复制并且向另一份合约重放执行该笔交易。

- **安全建议：无**
- **审计结果：通过**

11. 变量覆盖审计

以太坊存在着复杂的变量类型，例如结构体、动态数组等，如果使用不当，对其赋值后，可能导致覆盖已有状态变量的值，造成合约执行逻辑异常。

- **安全建议：无**
- **审计结果：通过**

业务审计

1. 做市引擎+订单薄

- **描述：**AppleSwapPair_deploy合约实现了做市引擎与订单薄的功能，支持自动化做市和链上撮合。合约向AppleSwapPair_deploy合约提供了addLimitOrder用于添加限价订单，addMarketOrder用于添加市价订单。用户可调用removeOrder撤销单个未完成订单，也可以调用removeOrders批量撤销未完成订单。此外，本合约还提供了对应的流动性池以及订单状态查询接口，用于外部系统获取链上数据。
- **相关函数：addLimitOrder、addMarketOrder、removeOrder、removeOrders、getPrices、calcStockAndMoney、getOrderList**

- **安全建议：无**
- **审计结果：通过**

2. **交易对管理**

- **描述：**AppleSwapFactory_deploy_main合约存储了一些关键参数并实现了交易池的创建功能。AppleSwapFactory_deploy_main存储了交易池的交易手续费比例FeeBPS、交易池盈利的代币的受益人地址feeTo_1、feeTo_2等及交易池逻辑地址pairLogic，其中FeeBPS和pairLogic只能由治理合约地址更改，feeTo由初始化时设置的feeToSetter地址进行修改。另外，任何人可以调用createPair方法创建交易池代理合约AppleSwapPairProxy。
- **相关函数：createPair、setFeeTo、setFeeToSetter、setPairLogic、setFeeBPS**
- **安全建议：无**
- **审计结果：通过**

3. **回购功能**

- **描述：**AppleSwapBuyback_deploy合约实现了回购。所有交易池中因注入和提取流动性所产生的流动性代币（LP）都可打入AppleSwapBuyback_deploy合约。任何人可调用removeLiquidity方法在对应交易池将LP代币代币转化为对应交易池的代币。另外合约还定义了mainToken数组用以保存支持的代币类型，任何人可调用swapForMainToken方法将合约中的代币在对应交易池将代币转化为支持的代币类型。任何人都可以调用swapForApplesAndBurn方法将合约中支持的代币通过对应交易池转化为Apple并销毁。AppleSwapToken_deploy合约的owner可以更改支持的代币类型。
- **相关函数：addMainToken、removeMainToken、removeLiquidity、swapForMainToken、swapForApplesAndBurn**
- **安全建议：无**
- **审计结果：通过**

4. **治理**

- **描述：**AppleSwapGov_deploy合约实现了治理相关功能。AppleSwapGov_deploy合约的owner可以调用submitFundsProposal、submitParamProposal及submitUpgradeProposal方法发起使用治理合约Apple余额、更改交易池手续费及更改交易池逻辑地址的提案。任何持有Apple超过总量0.1%的地址都可以调用submitTextProposal进行文本提案，但必须在上个提案结束一天之后。每个提案持续三天，且只能同时存在一个提案。任何用户都可以通过调用vote方法并在AppleSwapGov_deploy中锁定Apple对提案表示“同意“或”反对“（提案未结束前可改票）。截至投票结束，获得Apple更多的选项即为提案结果，若提案失败，提案发起者将被扣除部分Apple进行销毁。若提案通过，任何人都可以调用tally方法执行提案。用户在未参加当前提案投票时，可调用withdrawApples取出锁定的Apple。

- 相关函数：submitFundsProposal、submitParamProposal、submitUpgradeProposal、submitTextProposal、vote、tally、withdrawApples
- 安全建议：无
- 审计结果：通过

5. 路由功能

- 描述：AppleSwapRouter_deploy合约实现了限价单、市价单及注入和移除流动性的功能，且支持跨交易池的市价单。任何人可以调用swapToken方法进行单个交易池或跨交易池的兑换，可调用limitOrder创建限价单，可调用addLiquidity向交易池注入流动性（若不存在此流动池则创建），可调用removeLiquidity去除交易池中的流动性。
- 相关函数：limitOrder、swapToken、addLiquidity、removeLiquidity
- 安全建议：无
- 审计结果：通过

6. LP代币挖矿

- 描述：ApplePool合约实现了抵押LP代币挖矿Apple的功能。用户通过deposit函数将流动性代币抵押到ApplePool合约中进行挖矿，具体收益率根据owner添加对应pool信息时设定的ApplePerBlock参数而定。在抵押期满后，用户可以通过调用withdraw函数提取抵押并结清所有奖励。
- 相关函数：addPool、updatePool、reclaimAppleStakingReward、deposit、withdraw
- 安全建议：无
- 审计结果：通过

7. 均分手续费

- 描述：PairFeeDistribution_deploy合约实现了均分手续费的功能。每一个AppleSwapPair_deploy合约在扣除手续费时，都将把部分手续费自动发送至本合约。合约的owner可以通过调用addInvestors函数来添加参与分红的成员，通过调用removeInvestors函数来移除成员。成员从加入开始均分产生的手续费（LP代币）。成员可通过调用withdrawfee函数领取指定池子的手续费，也可通过batchwithdrawfee函数批量领取多个池子的手续费分成。
- 相关函数：addpair、addInvestors、removeInvestors、updateInvestorPairPerShare、withdrawfee、batchwithdrawfee
- 安全建议：

1）AppleSwapFactory_deploy_main合约中可以更改PairFeeDistribution_deploy合约地址，若owner权限丢失，可能会导致用户无法分到手续费收益。

- 修复结果：忽略

- **审计结果：通过**

8. **交易挖矿**

- **描述：**ExchangePool合约实现了交易挖矿的功能。用户在进行代币兑换时，如果兑换的交易对中包含指定的交易对，将触发ExchangePool合约的Exchange函数，将获得部分手续费价值的Apple代币，用户通过调用ExchangePool合约中的withdraw函数取出Apple代币奖励。
- **相关函数：Exchange、withdraw**
- **安全建议：无**
- **审计结果：通过**

9. **锁仓挖矿**

- **描述：**DAOPool合约实现了锁仓挖矿的功能。用户可以通过调用deposit函数将代币抵押到DAOPool合约中获取奖励，本金将被锁定。用户可以随时调用reclaimStakingReward取出获得的奖励，但只有锁仓时间到期后才可取出本金。
- **相关函数：reclaimStakingReward、withdraw、deposit、pendingReward**
- **安全建议：无**
- **审计结果：通过**

结论

Beosin(成都链安)对 Apple 项目智能合约的设计和代码实现进行了详细的审计。审核过程中发现的所有问题均已通知项目组，并得到项目组的快速反馈和修复，所有发现的问题均已得到妥善解决，或已与项目方就如何处理问题达成协议。Apple 项目智能约的总体审计结果是**通过**。

成都链安
BEOSIN

官方网址

https://lianantech.com

电子邮箱

vaas@lianantech.com

微信公众号