智能合约安全审计报告

1 前言

慢雾安全团队于 2020 年 12 月 07 日，收到 LINA 团队对 LINA Finance 项目安全审计评估的申请，根据项目特点慢雾安全团队制定如下审计方案。

慢雾安全团队将采用“白盒为主，黑灰为辅”的策略，以最贴近真实攻击的方式，对项目进行安全审计。

慢雾科技 DApp 项目测试方法：

黑盒测试	站在外部从攻击者角度进行安全测试。
灰盒测试	通过脚本工具对代码模块进行安全测试，观察内部运行状态，挖掘弱点。
白盒测试	基于项目的源代码，进行脆弱性分析和漏洞挖掘。

慢雾科技 DApp 漏洞风险等级：

严重漏洞	严重漏洞会对项目的安全造成重大影响，强烈建议修复严重漏洞。
高危漏洞	高危漏洞会影响项目的正常运行，强烈建议修复高危漏洞。
中危漏洞	中危漏洞会影响项目的运行，建议修复中危漏洞。
低危漏洞	低危漏洞可能在特定场景中会影响项目的业务操作，建议项目方自行评估和考虑这些问题是否需要修复。
弱点	理论上存在安全隐患，但工程上极难复现。
增强建议	编码或架构存在更好的实践方法。

2 审计方法

慢雾安全团队智能合约安全审计流程包含两个步骤：

- 使用开源或内部自动化分析的工具对合约代码中常见的安全漏洞进行扫描和测试。

- 人工审计代码的安全问题，通过人工分析合约代码，发现代码中潜在的安全问题。

如下是合约代码审计过程中我们会重点审查的漏洞列表:

（其他未知安全漏洞不包含在本次审计责任范围）

- 重入攻击
- 重放攻击
- 重排攻击
- 短地址攻击
- 拒绝服务攻击
- 交易顺序依赖
- 条件竞争攻击
- 权限控制攻击
- 整数上溢/下溢攻击
- 时间戳依赖攻击
- Gas 使用，Gas 限制和循环
- 冗余的回调函数
- 不安全的接口使用
- 函数状态变量的显式可见性
- 逻辑缺陷
- 未声明的存储指针
- 算术精度误差
- tx.origin 身份验证
- 假充值漏洞
- 变量覆盖

3 项目背景

3.1 项目简介

我们审计了 LINA Finance 的智能合约代码，如下是相关的文件信息：

审计版本文件信息：

SHA256(linear-only-buildr.zip)

c213495ff245579fe42f9580c8ee85db716dc067872c538f009a390864bbdbc7

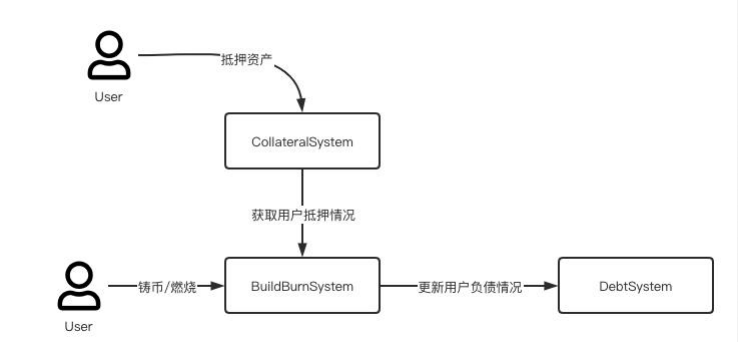
3.2 项目架构

```
contracts
├── IAsset.sol
├── IERC20.sol
├── ISharePool.sol
├── LinearFinanceToken.sol
├── LnAccessControl.sol
├── LnAddressCache.sol
├── LnAdmin.sol
├── LnAsset.sol
├── LnAssetSystem.sol
├── LnBuildBurnSystem.sol
├── LnChainLinkPrices.sol
├── LnCollateralSystem.sol
├── LnConfig.sol
├── LnDebtSystem.sol
├── LnDefaultPrices.sol
├── LnEndAdmin.sol
├── LnErc20Handler.sol
├── LnFundVault.sol
├── LnLinearStaking.sol
├── LnObserver.sol
├── LnOperatorModifier.sol
├── LnPrices.sol
├── LnProxyERC20.sol
```

```
├── LnProxyImpl.sol
├── LnRewardLocker.sol
├── LnSharePool.sol
├── LnSimpleStaking.sol
├── LnTokenLocker.sol
├── LnTokenStorage.sol
├── LnTokenStorageLock.sol
├── Migrations.sol
└── SafeDecimalMath.sol
```

3.3 合约架构

LINA Finance 项目合约主要分成三个部分，分别为 CollateralSystem 合约、DebtSystem 合约及 BuildBurnSystem 合约。其中 CollateralSystem 合约用于用户抵押 LINAToken。DebtSystem 合约记录用户的借贷状况。BuildBurnSystem 合约用于生成和燃烧合成代币。项目整体架构如下：



4 项目代码概览

4.1 合约概览

我们对合约主要合约进行了函数可见性分析，结果如下：

LnBuildBurnSystem			
Function	Visibility	Mutaibility	Modifier

constructor	Public	can modify state	LnAdmin
setPaused	External	can modify state	onlyAdmin
updateAddressCache	Public	can modify state	onlyAdmin
SetLusdTokenAddress	Public	can modify state	onlyAdmin
MaxCanBuildAsset	Public	-	-
BuildAsset	Public	can modify state	whenNotPaused
BuildMaxAsset	External	can modify state	whenNotPaused
_burnAsset	Internal	can modify state	-
BurnAsset	External	can modify state	whenNotPaused
BurnAssetToTarget	External	can modify state	whenNotPaused

LnAsset			
Function	Visibility	Mutaibility	Modifier
keyName	External	-	-
constructor	Public	-	LnErc20Handler
updateAddressCache	Public	-	onlyAdmin
_mint	Private	can modify state	-
mint	External	can modify state	OnlyIssueAssetRole
burn	External	can modify state	OnlyBurnAssetRole
_burn	Private	can modify state	-

setTotalSupplyAggre	External	can modify state	onlyObserver

LnErc20Handler			
Function	Visibility	Mutaibility	Modifier
constructor	Public	can modify state	LnAdmin LnProxyImpl
allowance	Public	-	-
balanceOf	External	-	-
setTokenStorage	External	can modify state	optionalProxy_onlyAdmin
_internalTransfer	Internal	can modify state	-
_transferByProxy	Internal	can modify state	-
_transferFromByProxy	Internal	can modify state	-
_beforeTokenTransfer	Internal	can modify state	-
transfer	External	can modify state	optionalProxy
transferFrom	External	can modify state	optionalProxy
approve	Public	can modify state	optionalProxy
addressToBytes32	Internal	-	-
emitTransfer	Internal	can modify state	-
emitApproval	Internal	can modify state	-
emitTokenStorageUpdat	Internal	can modify state	-

ed			

LnAssetSystem			
Function	Visibility	Mutaibility	Modifier
constructor	Public	can modify state	LnAddressStorage
addAsset	External	can modify state	onlyAdmin
removeAsset	External	can modify state	onlyAdmin
assetNumber	External	-	-
totalAssetsInUsd	Public	-	-
getAssetAddresses	External	-	-

LnDebtSystem			
Function	Visibility	Mutaibility	Modifier
constructor	Public	can modify state	LnAdmin
updateAddressCache	Public	can modify state	onlyAdmin
SetLastCloseFeePeriodAt	External	can modify state	OnlyDebtSystemRole
_pushDebtFactor	Private	can modify state	-
PushDebtFactor	External	can modify state	OnlyDebtSystemRole

_updateUserDebt	Private	can modify state	-
UpdateUserDebt	External	can modify state	OnlyDebtSystemRole
UpdateDebt	External	can modify state	OnlyDebtSystemRole
GetUserDebtData	External	-	-
_lastSystemDebtFactor	Private	-	-
LastSystemDebtFactor	External	-	-
GetUserCurrentDebtProportion	Public	-	-
GetUserDebtBalanceInUsd	External	-	-

LnCollateralSystem			
Function	Visibility	Mutaibility	Modifier
constructor	Public	can modify state	LnAdmin
setPaused	External	can modify state	onlyAdmin
updateAddressCache	Public	can modify state	onlyAdmin
updateTokenInfo	Private	can modify state	-
UpdateTokenInfo	External	can modify state	onlyAdmin
UpdateTokenInfos	External	can modify state	onlyAdmin

GetSystemTotalCollateralInUsd	Public	-	-
GetUserTotalCollateralInUsd	Public	-	-
GetUserCollateral	External	-	-
GetUserCollaterals	External	-	-
Collateral	External	can modify state	whenNotPaused
IsSatisfyTargetRatio	Public	-	-
MaxRedeemableInUsd	Public	-	-
MaxRedeemable	Public	-	-
RedeemMax	External	can modify state	whenNotPaused
_Redeem	Internal	can modify state	-
Redeem	Public	can modify state	whenNotPaused
constructor	External	payable	whenNotPaused
_CollateralEth	Internal	can modify state	-
CollateralEth	External	payable	whenNotPaused
RedeemETH	External	can modify state	whenNotPaused

4.2 代码审计

4.2.1 低危漏洞

4.2.1.1 代币总量不一致

tokenStorage 用于存储代币信息，但是没有存储代币的 total_Supply，导致代理合约改变了 total_supply 可能会不一致。

代码位置：lnAsset.sol 第 22 行

```
    constructor( bytes32 _key, address payable _proxy, LnTokenStorage _tokenStorage, string memory _name, string memory _symbol,
        uint _totalSupply, uint8 _decimals, address _admin) public
        LnErc20Handler(_proxy, _tokenStorage,_name,_symbol, _totalSupply, _decimals, _admin ) {

        mKeyName = _key;
        tokenStorage = _tokenStorage;
        name = _name;
        symbol = _symbol;
        totalSupply = _totalSupply;
        totalSupplyAggre = _totalSupply;
        decimals = _decimals;
    }
```

修复建议：将 total_supply 变量存储在 tokenStorage 合约逻辑中。

修复状态：未修复。经与项目方沟通过后，确定此问题在首次部署时，由于 totalSupply 为 0，所以该问题暂时不影响。此问题后续升级合约时再处理。

4.2.1.2 合约之间地址可能不同步

LnCollateralSystem, LnDebtSystem, LnBuildBurn 合约都实现了 updateAddressCache 函数，当地址更新时，无法保证每个合约之间互相同步，建议当一个合约的 address cache 更新时，同时更新其他合约

的 address cache。

代码位置：LnBuildBurnSystem.sol 第 46 行，LnCollateralSystem.sol 第 67 行，LnDebtSystem.sol 第 47 行

```
function updateAddressCache( LnAddressStorage _addressStorage ) onlyAdmin public override
    {
        priceGetter =      LnPrices( _addressStorage.getAddressWithRequire( "LnPrices",       "LnPrices address not valid" ) );
        debtSystem = LnDebtSystem( _addressStorage.getAddressWithRequire( "LnDebtSystem", "LnDebtSystem address not valid" ) );
        assetSys =   LnAssetSystem( _addressStorage.getAddressWithRequire( "LnAssetSystem","LnAssetSystem address not valid" ) );
        address payable collateralAddress = payable(_addressStorage.getAddressWithRequire( "LnCollateralSystem","LnCollateralSystem address not valid" ));
        collaterSys = LnCollateralSystem( collateralAddress );
        mConfig =          LnConfig( _addressStorage.getAddressWithRequire( "LnConfig",       "LnConfig address not valid" ) );

        emit updateCachedAddress( "LnPrices",               address(priceGetter) );
        emit updateCachedAddress( "LnDebtSystem",           address(debtSystem) );
        emit updateCachedAddress( "LnAssetSystem",          address(assetSys) );
        emit updateCachedAddress( "LnCollateralSystem", address(collaterSys) );
        emit updateCachedAddress( "LnConfig",               address(mConfig) );
    }
```

修复建议：建立用于更新地址的合约，统一对合约地址进行更新。

修复状态：未修复，经与项目方沟通后，项目方确认首次部署时并不受影响。而后续项目方可能会加一个 helper contract，同时调用几个合约来确保原子性。或者也可以让系统暂停运作，再开始升级。

4.2.2 增强建议

4.2.1.2 LnBuildBurnSystem / LnCollateralSystem 合约未检查相关地址是否存在

LnCollateralSystem / LnCollateralSystem 合约使用 updateAddressCache 函数设置外部合约地址，但是合约中很多用到这些地址的地方没有先校验一遍外部地址已经设置完成，导致合约函数无法成功调用，如 LnCollateralSystem / LnCollateralSystem 合约的 GetSystemTotalCollateralInUsd /GetUserTotalCollateralInUsd/GetUserCollateral 函数时，未检查对应的预言机地址是否已经设置，导致函数无法调用。

```
    function GetSystemTotalCollateralInUsd() public view returns (uint256 rTotal) {
        for (uint256 i=0; i< tokenSymbol.length; i++) {
            bytes32 currency = tokenSymbol[i];
            if (tokenInfos[currency].totalCollateral > 0) { // this check for avoid calling getPrice when collateral is zero
                if (Currency_LINA == currency) {
                    uint256 totallina = tokenInfos[currency].totalCollateral.add(mRewardLocker.totalNeedToReward());
                    rTotal = rTotal.add( totallina.multiplyDecimal(priceGetter.getPrice(currency)) );
                } else {
                    rTotal =
rTotal.add( tokenInfos[currency].totalCollateral.multiplyDecimal(priceGetter.getPrice(currency)) );
                }
            }
        }

        if (address(this).balance > 0) {
            rTotal = rTotal.add(address(this).balance.multiplyDecimal(priceGetter.getPrice(Currency_ETH)));
        }
    }
```

修复建议：在 updateAddressCache 设置一个状态变量 initialize，外部地址设置成功后设置为 true，需要用到外部地址的函数，需要先校验 initialize 是否为 true，避免地址未正确设置导致的调用异常。

修复状态：未修复，经与项目方确认后，项目方确认首次部署时人工确保地址都已正确加载，此问题于后续

迭代修复。

5 审计结果

5.1 结论

审计结果: 低风险

审计编号: 0X002012150002

审计日期: 2020 年 12 月 15 日

审计团队 ：慢雾安全团队

总结: 本次审计中发现了 2 个安全问题，其中包含 2 个低危问题，同时给出了 1 个提升建议。经与项目方沟通后，所有的问题或风险均在可接受范围内。

6 声明

慢雾仅就本报告出具前已经发生或存在的事实出具本报告，并就此承担相应责任。对于出具以后发生或存在的事实，慢雾无法判断其智能合约安全状况，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，仅基于信息提供者截至本报告出具时向慢雾提供的文件和资料(简称“已提供资料”)。慢雾假设：已提供资料不存在缺失、被篡改、删减或隐瞒的情形。如已提供资料信息缺失、被篡改、删减、隐瞒或反映的情况与实际情况不符的，慢雾对由此而导致的损失和不利影响不承担任何责任。

Official Website
www.slowmist.com

E-mail
team@slowmist.com

Twitter
@SlowMist_Team

WeChat Official Account