Haven Protocol

代币安全审计报告

目录

1 前言(Executive Summary)

慢雾安全团队于 2020-08-14 日，收到 Haven Protocol 团队对 Haven Protocol 代币安全审计申请，根据双方约定和项目特点制定审计方案，并最终出具安全审计报告。

慢雾安全团队采用“黑盒+灰盒+白盒”的策略，以最贴近真实攻击的方式，对项目方进行完整的安全测试。

慢雾科技区块链系统测试方法：

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	基于源代码，对节点、SDK 等程序进行漏洞挖掘。

慢雾科技区块链风险等级：

严重漏洞	严重漏洞会对区块链的安全造成重大影响，强烈建议修复严重漏洞。
高危漏洞	高危漏洞会影响区块链的正常运行，强烈建议修复高危漏洞。
中危漏洞	中危漏洞会影响区块链的运行，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区块链的操作，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 项目背景(Context)

2.1 项目简介

项目官网：https://www.havenprotocol.com/

代币符号：XHV

项目源码：https://github.com/haven-protocol-org/haven-offshore

审计版本：

Commit: 3c7439fd0e142870b48728f920bc41c517e2d029

2.2 审计范围

本次安全审计的主要类型包括：

序号	审计子类	审计结果
1	代码合规性审计	通过
2	随机数生成算法安全审计	通过
3	密钥存储与内存安全审计	通过
4	密码学组件调用安全审计	通过
5	加密强度安全审计	通过
6	交易延展性安全审计	通过
7	交易重放审计	通过
8	代币“假充值”漏洞审计	通过
9	RPC“黑色情人节”漏洞审计	通过

（其他未知安全漏洞不包含在本次审计责任范围）

3 代码分析(Code Overview)

3.1 基础架构

Haven-offshore 基于开源的 monero release-v0.14 开发，主要功能变动如下：

(1). 基于价格预言机生成稳定币 xUSD；

(2). 环签名升级为 CLSAG 签名，具有更好的性能；

(3). 优化地址格式、增加 Haven 标识等；

3.2 代码合规性审计

Fork 开源公链源代码生成新的公链会存在重放攻击、节点地址池污染等问题，对此我们进行相关安全合规性评估。

(1) 节点地址池污染评估

- patches/src/cryptonote_config.h.patch

```
boost::uuids::uuid const NETWORK_ID = { {
- 0x12 ,0x30, 0xF1, 0x71 , 0x61, 0x04 , 0x41, 0x61, 0x17, 0x31, 0x00, 0x82, 0x16, 0xA1, 0xA1, 0x10
- } }; // Bender's nightmare
- std::string const GENESIS_TX =
"013c01ff0001ffffffffffff03029b2e4c0281c0b02e7c53291a94d1d0cbff8883f8024f5142ee494ffbbd08807121017
767aafcde9be00dcfd098715ebcf7f410daebc582fda69d24a28e9d0bc890d1";
+ 0x05 ,0x39, 0xF1, 0x70 , 0x61, 0x04 , 0x41, 0x60, 0x17, 0x32, 0x00, 0x81, 0x16, 0xA1, 0xA1, 0x10
+ } };
```

- src/p2p/net_node.inl

```
template<class t_payload_net_handler>
bool node_server<t_payload_net_handler>::do_handshake_with_peer(peerid_type& pi,
p2p_connection_context& context_, bool just_take_peerlist)
{
network_zone& zone = m_network_zones.at(context_.m_remote_address.get_zone());

typename COMMAND_HANDSHAKE::request arg;
```

```
typename COMMAND_HANDSHAKE::response rsp;
get_local_node_data(arg.node_data, zone);
m_payload_handler.get_payload_sync_data(arg.payload_data);

epee::simple_event ev;
std::atomic<bool> hsh_result(false);

bool r = epee::net_utils::async_invoke_remote_command2<typename
COMMAND_HANDSHAKE::response>(context_.m_connection_id, COMMAND_HANDSHAKE::ID, arg,
zone.m_net_server.get_config_object(),
[this, &pi, &ev, &hsh_result, &just_take_peerlist, &context_](int code, const typename
COMMAND_HANDSHAKE::response& rsp, p2p_connection_context& context)
{
epee::misc_utils::auto_scope_leave_caller scope_exit_handler =
epee::misc_utils::create_scope_leave_handler([&](){ev.raise();});

if(code < 0)
{
LOG_WARNING_CC(context, "COMMAND_HANDSHAKE invoke failed. (" << code << ", " <<
epee::levin::get_err_descr(code) << ")");
return;
}

if(rsp.node_data.network_id != m_network_id)
{
LOG_WARNING_CC(context, "COMMAND_HANDSHAKE Failed, wrong network! (" << rsp.node_data.network_id <<
"), closing connection.");
return;
}
```

Haven Protocol 与 Monero 节点的 network_id 不相同，不会造成节点间地址池互相污染的问题。

漏洞成因参考：https://mp.weixin.qq.com/s/UmricgYGUakAlZTb0ihqdw

(2) 价格预言机安全性评估

- patches/src/cryptonote_core/blockchain.cpp.patch

```
+bool Blockchain::get_pricing_record(offshore::pricing_record& pr, uint64_t timestamp) const
+{
+ LOG_PRINT_L1("Requesting pricing record from Oracle - time : " << timestamp);
```

```
+
+ epee::net_utils::http::http_simple_client http_client;
+ COMMAND_RPC_GET_PRICING_RECORD::request req = AUTO_VAL_INIT(req);
+ COMMAND_RPC_GET_PRICING_RECORD::response res = AUTO_VAL_INIT(res);
+
+ std::array<std::string, 3> oracle_urls = {{"oracle.havenprotocol.org:443",
"oracle2.havenprotocol.org:443", "oracle3.havenprotocol.org:443"}};
+ std::shuffle(oracle_urls.begin(), oracle_urls.end(),
std::default_random_engine(crypto::rand<unsigned>()));
```

价格预言机由 3 个官方节点组成，中心化权限较大

预言机代码位置：https://github.com/dweab/oracle-lite

(3) 预言机价格操控防范

xUSD 和 XHV 之间的兑换不需要对手盘，由于初期流动性不足，这里有一个价格操控问题，利用短时的拉盘或者砸盘实现的 xUSD 和 XHV 汇率操控，进而增发大量的 xUSD 或 XHV。

对此问题，Haven Protocol 做了相应的防范：第一点就是兑换的价格不是某个节点的交易所价格，目前使用的兑换价格是前面 720 个区块里记录的价格平均值。然后就是一个兑换还有一个费率，目前使用的费率是根据兑换解锁时间来确定，目前最快的解锁周期是 6 个小时，你要 6 个小时解锁的话，你需要支付你兑换总额的 20%作为兑换费率，24 小时解锁的 10%，48 小时的 5%，168 小时的 0.2%。后期计划设计更合理的费率机制，会根据当前流通量的大小，以及用户当时兑换对总量的影响来增加类似兑换滑点。

3.3 随机数生成算法安全审计

基于/dev/urandom 生成随机数，满足私钥随机数的安全性。

- src/crypto/crypto.cpp

```
secret_key crypto_ops::generate_keys(public_key &pub, secret_key &sec, const secret_key& recovery_key,
bool recover) {
ge_p3 point;

secret_key rng;

if (recover)
{
rng = recovery_key;
```

```
}
else
{
random_scalar(rng);
}
sec = rng;
sc_reduce32(&unwrap(sec)); // reduce in case second round of keys (sendkeys)

ge_scalarmult_base(&point, &unwrap(sec));
ge_p3_tobytes(&pub, &point);

return rng;
}
```

3.4 密钥存储与内存安全审计

未校验 keystore 密码强度，测试中可使用`123456`之类的弱口令，极易被猜解。

3.5 密码学组件调用安全审计

Haven Protocol 引入了 CLSAG 签名，与 Monero 协议中使用的当前环签名结构相比，CLSAG 签名更小，更快，并且具有严格的安全性。

算法论文：https://eprint.iacr.org/2019/654.pdf

相关代码：patches/src/ringct/rctSigs.cpp.patch

由于环签名是 Monero 协议的关键组成部分，因此 Monero 社区委托对 CLSAG 密码（算法，安全模型和证明）以及将要部署的实现代码进行正式的安全审核。

审计结果参考：https://web.getmonero.org/resources/research-lab/audits/clsag.pdf

慢雾安全团队基于已知的攻击方式对 CLSAG 及 bulletproof 算法应用进行安全评估，未发现导致代币增发和销毁的代码缺陷，没有发现安全问题。

3.6 加密强度安全审计

未使用 md5、sha1 等弱哈希函数。

3.7 交易延展性安全审计

交易所有字段都进行编码并计算哈希值，没有交易延展性漏洞。

- src/cryptonote_basic/cryptonote_basic.h

```
class transaction_prefix
{

public:
// tx information
size_t version;
uint64_t unlock_time; //number of block (or time), used as a limitation like: spend this tx not early
then block/time

std::vector<txin_v> vin;
std::vector<tx_out> vout;
//extra
std::vector<uint8_t> extra;

BEGIN_SERIALIZE()
VARINT_FIELD(version)
if(version == 0 || CURRENT_TRANSACTION_VERSION < version) return false;
VARINT_FIELD(unlock_time)
FIELD(vin)
FIELD(vout)
FIELD(extra)
END_SERIALIZE()
```

3.8 交易重放审计

基于 UTXO 模型，同链上没有重放问题，不同链上不存在相同的 UTXO，无法重放交易。

3.9 代币“假充值”漏洞审计

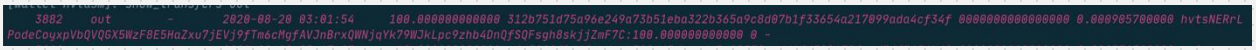
1. 使用 transfer 转账 XHV

transfer

hvtaPHpEsPFGkL7MVMoFKkeHkeo65hdJEgWdD4veGodmS1B58EZruaVKxXn5PDJDCuc6Pm99p

aTN13gxcydjsrFh17c9wGxC8U 100

转账记录



Test2 接收

2. 使用 offshore_transfer 转账 xUSD

offshore_transfer

hvtaPHpEsPFGkL7MVMoFKkeHkeo65hdJEgWdD4veGodmS1B58EZruaVKxXn5PDJDCuc6Pm99p

aTN13gxcydjsrFh17c9wGxC8U 100

支持多币种，包括{"XHV", "xAG", "xAU", "xAUD", "xBTC", "xCAD", "xCHF", "xCNY", "xEUR", "xGBP",

"xJPY", "xNOK", "xNZD", "xUSD"}

需要用查看密钥查看资金，支持多币种，**入账时需要注意区分代币符号**。

3.10 RPC“黑色情人节”漏洞审计

RPC 有 sign_transfer 功能，但节点默认只允许在本地开放 RPC 端口，避免了类似以太坊“黑色情人节”的远程转账漏洞。

漏洞参考：https://mp.weixin.qq.com/s/Kk2IsoQ1679Gda56Ec-zJg

4 审计结果(Result)

4.1 增强建议

- 价格预言机存中心化权限较高，建议分散权限以提高分布式安全性。

4.2 交易所安全小结

- 充值入账需要检测交易和收据结构中所有相关字段，并与账户总余额实时对账，出现异常时需要人工排查后再处理入账，防范“假充值攻击”。
- 主链存在多种代币资产，入账时需要注意区分代币符号。

4.3 结论

审计结果：通过

审计编号：BCA002008250001

审计日期：2020 年 08 月 25 日

审计团队：慢雾安全团队

综合结论：经反馈修正后，所有发现问题均已修复，综合评估 Haven Protocol 已无上述风险。

5 声明(Statement)

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，慢雾无法判断该项目安全状况，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。慢雾仅对该项目的安全情况进行约定内的安全审计并出具了本报告，慢雾不对该项目背景及其他情况进行负责。

慢雾科技
slow mist

官方网址

www.slowmist.com

电子邮箱

team@slowmist.com

微信公众号