HyperService 进展

(190313-190320)

1 进度

1.1 VES

1.1.1 sessionSetupPrepare(json op_intents)

说明: 准备创建 session.
格式化 Op-intent, 生成 $\mathcal{G}_{\mathcal{T}}$, 最后生成 session. 并且返回一个 Cert(session_content; $\text{Sig}_{\mathcal{V}}$)(异步以后取消这个内容，直接对其他 dApp 发送 $\text{Cert}_{\mathcal{V}}$ 请求确认).

1.1.2 sessionSetupUpdate(integer session_id ,Certificate ack)

说明: 投票 session.
ack = [user, $\text{Cert}_{\mathcal{V},\mathcal{D}_{\text{u}}}$ *or* None]

1.1.3 sessionSetupFinished(integer session_id)

说明: 结束sessionSetup.
向 NSB 添加信息 , 并 stake funds. 其他 dApp 发送 $\text{Atte}_{\mathcal{V},\mathcal{D}} = \text{Cert}(\text{session\_content}; \text{Sig}_{\mathcal{V}}, \{\text{Sig}_{\mathcal{D}_{\text{u}}}\})$.
PS:$\text{Atte}_{\mathcal{V},\mathcal{D}}$ 的验证方法是:
1.verifySign(Sig_{V}, session_content, $\mathcal{V}$)
2.verifySign($\text{Sig}_{\mathcal{D}_{\text{u}}}$, Sig_{V}, $\mathcal{D}_{\text{u}}$)

1.1.4 transact(integer session_id)

TODO

1.2 ISC

1.2.1 CreateContract(address[] owner,uint[] funds_require, ? tx_intents)

tx_intent 最理想是json[], 但这不可能, 具体形式待定.

1.3 dApp

说明: 现在可以进行 ethereum 签名, 收发交易.

1.4 其他模块

1.4.1 Class uip.OpIntent

说明: 可扩展的输入检查.
使用`createopintents(json op_intents)`就行了.

1.4.2 Class uip.TransactionIntents

说明: 依据已经格式化的 Opintents 和 Dependencies 初始化 Transaction intents 和 $\mathcal{G}_{\mathcal{T}}$(Dependency Graph), 输出是 [Topological_Sorted($V_{\mathcal{G}_{\mathcal{T}}}$), $\mathsf{dependencies}_{\mathcal{G}_{\mathcal{T}}}$]. 其中 Topological_Sorted($V_{\mathcal{G}_{\mathcal{T}}}$) 是`Transaction[]`.

1.4.3 Class eth.Transaction

说明: 使用`self.jsonize()`返回 ethereum 可执行的`object Transaction`.

1.4.4 Class eth.tools.AbiEncoder

1.4.5 Class eth.tools.AbiDecoder

1.4.6 Class eth.tools.SignVerify

2 遇到的问题

2.1 ISC 的输入问题

solidity 只支持一维数组 (string, bytes) 或者固定的二维数组 (bytes[789][]).

2.2 ISC 的监听问题

contract 如果不在同一个 code, 那么并不支持在链上直接交互. 现在由 VES 暂代 ISC 进行 insurance claim.

2.3 contract invoke 的 data encode

要么用户直接提供已经 encode 过的 function-data. 要么必须要一个 parament_type_list 提供所有变量的类型描述. 这个类型描述有两个作用，一是生成函数签名 (例如 `isOwner(address)` 的函数签名)，二是 encode data.

2.4 contract invoke 的返回值问题

storage position 可以解决问题，这个需要用户提供吗? 也就是 contract invocation 的 op_intent 里需要增加一个 storage position 域.

2.5 VES 签名问题

用什么私钥去签?(VES 是人?).

3 现在的 Op-intent 设定

```
Key_Attribute_All = ('name', 'op_type')
Key_Attribute_Payment = ('amount', 'src', 'dst')
Key_Attribute_ContractInvocation = ('invoker', 'contract_domain', 'func')
Option_Attribute_Payment = ('unit',)
Option_Attribute_ContractInvocation = (
    'parameters',
    'parameters_description'
)
Op_Type = ('Payment', 'ContractInvocation')
Chain_Default_Unit = {
    'Ethereum': 'wei'
}
```