浅析Libra背后的区块链技术

2019-11-20

前段时间国家领导人曾公开表示鼓励区块链技术的研究，称要把区块链作为核心技术自主创新的突破口。Libra的发行计划是区块链发展史上一座重要的里程碑，本文从合约语言、数据库协议、逻辑数据模型、数据结构、共识协议等方面简要介绍Libra区块链的技术方案。

2019年5月，Facebook首次确认推出加密货币的意向，“全球币”、“脸书币”的消息不胫而走。6月18日，Facebook正式宣布将推出名为Libra的加密货币，预计2020年上半年针对性发布。Facebook宣称，Libra建立在安全、可靠、可扩展的区块链上，采用链外资产抵押的模式，锚定一篮子法定货币作为资产担保，由独立的Libra协会治理。Facebook在全球拥有27亿用户，Libra的愿景是建立一套简单、无国界的货币，为数十亿人提供金融服务。

Facebook正式宣布Libra后，同步上线了Libra的官网、白皮书和测试网络等内容。白皮书中提到，Libra网络希望未来以公有链的形式运作，但由于目前没有成熟的技术能在公有链上支撑大规模的交易，Libra将以联盟链的形式起步，计划三到五年内展开由联盟链过渡为公有链的研究。

Libra官网公布了三篇论文详细说明Libra使用的技术方案，这三篇论文分别是《Libra区块链》、《Move：一种具有可编程资源的语言》和《Libra区块链中的状态机复制》。Libra为满足高度安全、足够灵活、吞吐量极高等要求，设计了新的编程语言Move，选择BTF共识机制，采用广泛使用的Merkle Tree作为数据结构。本文简要介绍Libra区块链的相关技术。

Move

当现实世界的资产进入Libra储备，系统会创建相应的数字资产Libra货币，这些数字资产在不同账户之间流通，当现实中的资产离开Libra储备，对应的数字资产也随之销毁。Fackbook设计了新的编程语言Move用于对Libra中的数字资产进行管理，Libra在Move中将数字资产表示为资源（resource）。

Move是带有类型的字节码语言，资源是Move的类型之一，程序在执行前会先经过字节码验证器检验，然后由解释器执行。Move中的资源仅支持copy和move两种操作，可以理解为copy移出资源，move移入资源，也就是说资源和普通的变量类型不同，资源的值可以赋给普通变量，但资源本身只能在地址间移动，不能复制或丢弃。如果在程序中使用了违反规则的copy和move操作，比如copy一次move两次，程序将无法通过字节码验证器，因为在第一次move后原先的资源就已经不可访问了。

使用Move可以编写自定义的交易逻辑和智能合约，相比现有的合约语言要更加强大。比特币脚本提供了简洁优雅的设计用于表达花费比特币的策略，但是比特币脚本不支持自定义数据类型和程序，不是图灵完备的。以太坊虚拟机倒是支持流程控制、自定义数据结构等特性，但太过自由的合约让程序的漏洞随之增多，发生过多起安全事件。Move的静态类型系统为数字资产的安全性提供了保障。

为了配合静态验证工具的验证，Move在设计上采取了一些措施：没有动态调度，让验证工具更容易分析程序；限制可变性，每一次值的变化都要通过引用传递，临时变量必须在单个脚本中创建和销毁，字节码验证器使用类似Rust的”borrow checking”机制保证同一时间变量只有一个可变引用；模块化，验证工具可以从模块层面对程序进行验证而不需要关心具体实现细节，等等。Move的这些特性都使得静态验证工具更加高效可靠。

```
public main(payee: address, amount: u64) {
  let coin: 0x0.Currency.Coin = 0x0.Currency.withdraw_from_sender(copy(amount));
  0x0.Currency.deposit(copy(payee), move(coin));
}
```

这是一段交易脚本的示例程序，是Move语言的中间表示（IR），IR更适合程序员阅读和编写。

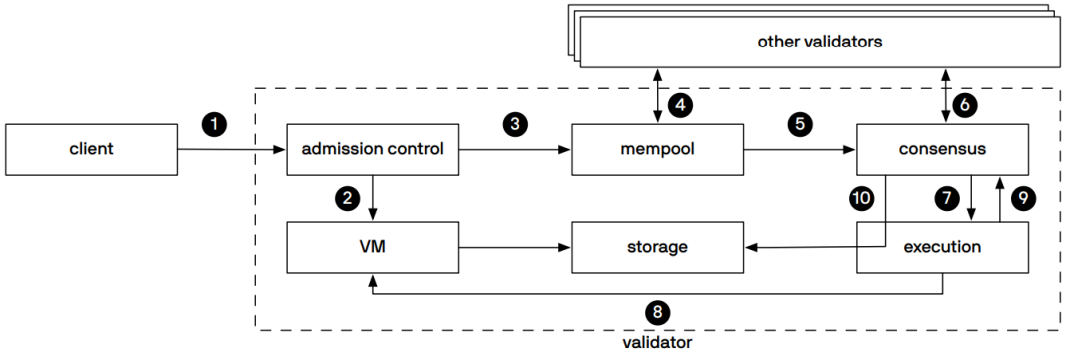
程序实现了一个转移资源的函数，main方法是脚本的入口，包含两个入参：目标地址和金额。程序先从0x0地址Currency模块中移出amount个资源暂存到coin变量，然后将coin的资源移动到payee的地址上。

交易脚本是为Move提供灵活性的一个方面，另一方面来自安全的模块化设计。交易脚本让交易逻辑更加自由，模块化设计则让保证了脚本程序的多样化。模块的类型是`module`，主要包含Move程序，一个模块可以包含任意个资源，也就是声明另个或多个资源类型的变量，modules/resources/procedures相当于面向对象语言中的classes/object/methods，不同的是Move中并没有self、this之类的概念。

Libra协议

Libra区块链是一个需要经过密码学认证的分布式数据库，用于储存可编程资源，比如Libra货币就是可编程资源，在Move中表现为资源。Libra协议中有两种实体类型，验证节点（validators）共同参与维护数据库，客户端（client）通常发起向数据库的请求。Libra协议会在执行过程中选举出leader接收客户端的请求，然后leader将请求同步到其他验证节点执行，其他验证节点执行结束后把结果返回给leader，leader再把请求的最终结果返回客户端。

Libra的交易会经过很多步骤，包括验签、运行先导程序、验证交易脚本和模块程序的正确性、发布模块、执行交易脚本、运行结尾程序等。为了使合约交易的计算能力可计量，Libra吸收了以太坊中Gas的概念，消耗Gas作为交易的费用。

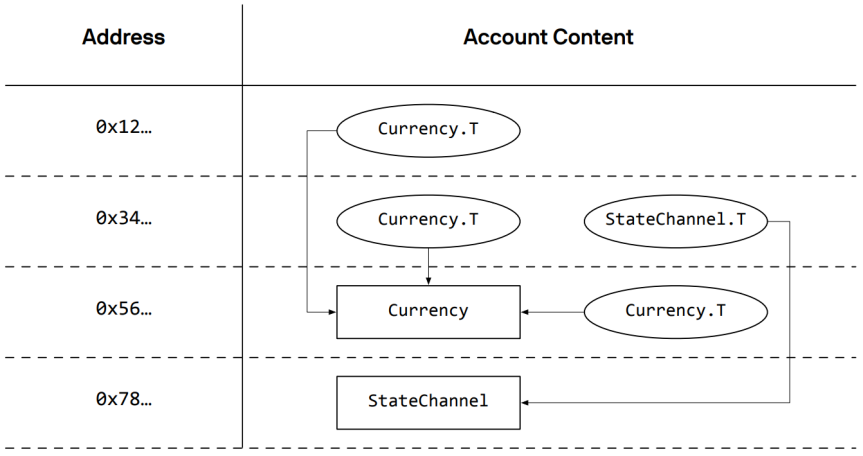


这张图详细展示了交易请求在Libra的网络组件中流转的过程，客户端发起请求到权限控制层，权限验证后将请求数据转给虚拟机进行预处理，同时数据也会进入内存池中，内存池负责将请求同步到其他节点，共识协议在请求同步的过程中发挥作用，节点同步结束后虚拟机执行真正的交易程序，程序执行完毕对结果持久化，基本流程结束。

逻辑数据模型

Libra区块链上所有的数据都保存在有版本号标识的数据库中，版本号是64位无符号整数。每个版本的数据库都包含一个元组 (T, O, S)，T代表交易，O代表交易的输出，S代表账本的状态。当我们说执行了一个Apply操作，表示为Apply(S, T) -> (O, S)，意思是在S状态下执行了T交易，产生了O输出并且账本的状态变为S。

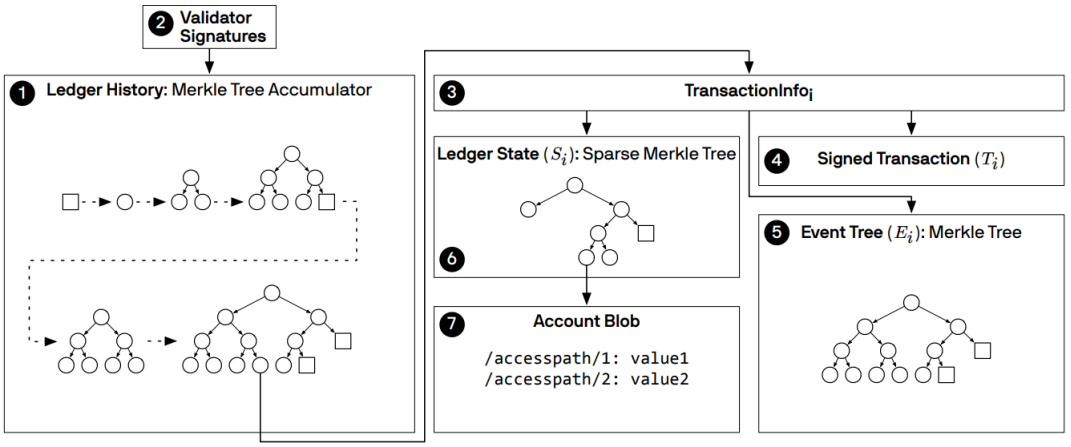
账户是资源的拥有者，可以使用账户内的资源进行交易。账户地址是一个256位的值，创建新账户需要一个验证/签名的键值对（vk, sk），新的账户地址a由vk经过公钥加密计算得到，a = H(vk)。具体来说Libra使用SHA3-256实例化哈希函数，使用wards25519椭圆曲线做变量的EdDSA公钥进行数字签名。交易过程中由已经存在的账户调用create_account(a) 指令即可生成新账户。



上图所示有四个以0x为前缀的账户地址，矩形框表示模块，椭圆形表示资源，箭头表示依赖关系。图中0x12账户中的Currency.T在Currency模块中声明，Currency模块的代码储存在0x56地址上。同理，0x34的StatChannel.T声明自0x78的StateChannel模块。当客户端想要访问0x12下的Currency.T，请求资源的路径应写作0x12/resources/0x56.Currency.T。

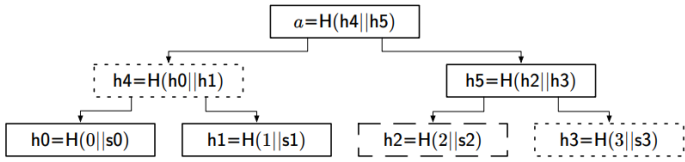
数据结构

Libra交易区块中包含各节点签名的数据，交易前会对签名数据进行校验，根据这一集体签名客户端可以相信请求的数据库版本是完整有效的，也因此客户端可以请求任意节点甚至是第三方数据库副本进行查询。Libra协议中的数据结构主要基于默克尔树。



如图所示，账本历史数据的根哈希用来验证系统的完整状态，账本数据由默克尔树累加形成，虚线表示数据累积的过程。账本历史数据的每一个节点都包含交易签名、事件树和账本状态，事件树也是基于默克尔树，账本状态则是基于稀疏默克尔树，账本状态的每个叶子节点都包含有账户数据。

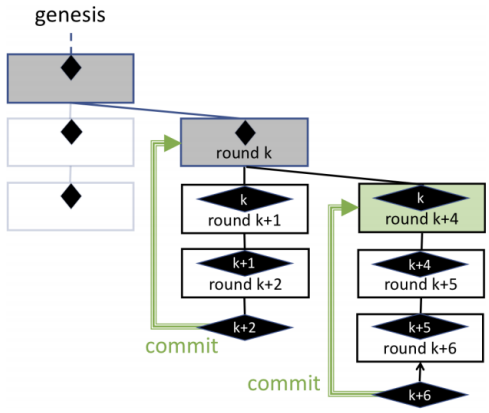
Libra协议中验证节点V会对数据D的根哈希a进行校验。例如不受信任的节点在获取到数据D后使用函数f对数据进行运算，希望得到结果r，同时还需要一个用于验证函数结果正确性的数据π，协议会要求节点把(a, f, r, π)都传到验证节点V处进行验证，如果f(D) = r则通过验证。



在上图中，数据D = {0:s0, 1:s1, 2:s2, 3:s3}。假设f是获取第三项数据的函数，也就是要获取h2的数据，期望结果f(D) = h2，此时h2就是r，r = h2，用于验证计算结果正确性的数据π = [h3, h4]，根哈希a = H(h4||h5) = H(h4 || H(H(2 || r) || h3))，验证节点将执行Verify(a, f, r, π)对计算结果进行验证。

共识协议

Libra选择使用的是拜占庭容错共识，实现了一种HotStuff共识的变体LibraBFT，简称LBFT。LBFT协议的主要作用是让提交的块在同一个序列上，或者说避免分叉。每三次提交为一轮操作，每一次提交验证节点都会投票选举出下一轮的leader，同时这些投票的集合形成一个法定证书（QC），每一轮的第一个块记为preferred_round，下一个块写入时对preferred_round的QC进行验证，也就是preferred_round后的第一、二、三个块都与preferred_round校验，第四个块将是第二轮的preferred_round，依次更迭。



如图所示，k是preferred_round，如果在k处出现了分叉，并且有2f + 1个验证节点投票给了k，k+1将接在k的后面，k+2依次写入，k左侧的分叉失效。这时假如k+3的leader超时了，k+4成为新的leader并写入在preferred_round（k）的后面，会引起新的分叉。如果k+4获得2f+1个投票，k+5会按照规则写入在k+4后面。

在k+4分叉后，当超时的k+3再次被选为leader并重新提交，验证节点会对k+3的preferred_round（k）的QC进行校验，校验通过，k+3写入在了k+2的后面，这符合规则。再然后，下一个leader（k+6）的preferred_round实际上是k+4，准备提交块到k+3后面时发现QC对不上，k及其后的k+1、k+2、k+3都会被删除，k+4后的链成为主链。

LBFT基本上是对链式HotStuff的实现，并没有太多创新，Libra团队更多的是在共识协议中做出一种选择，对BFT的选择是好是坏还存在争议，需要时间来验证，LBFT算是Libra从联盟链到公有链过渡前的方案，预计至少支持100个节点，上限大概是1000个左右。和HotStuff一样，LBFT最多容忍三分之一的不诚实节点。

小结

Libra协议目前还处于比较早期的阶段，性能上并不算惊艳，支持每秒1000笔交易，每次提交的交易确认时间大概是10秒钟。Libra协议在设计上很多也是兼顾了性能，比如每三次提交进行一

次共识，每一轮操作内不需要等待就可以进行投票，这减少了客户端和验证节点之间的网络延时；考虑到并行和分片的思想，稀疏默克尔树的使用使得账户的身份数据可以跨数据库进行验证，也支持并行更新等等。

Libra选择了众多成熟的技术构建Libra系统，使用内存安全的Rust编写核心程序、使用容易验证的Merkle Tree作数据结构、基于Chained HotStuff实现共识协议等等。Move是Libra在技术上最大的亮点之一，在语言层面保证了数字资产的安全性，Move本身是一种字节码语言，难以阅读，所以提供了Move的中间表示IR，用于编写交易脚本和智能合约。Libra作为一种在金融领域的创新实验，基于区块链提出了世界货币的愿景，其社会意义可能要远大于在技术创新上所带来的意义。Libra协会目前拥有16个成员组织，涵盖支付业、电信业、区块链业、风险投资业等领域，已经在世界范围引起广泛关注。

Libra预计2020年上半年针对性发布，让我们拭目以待！