

区块链世界里不能信什么？

原创 张开翔 [FISCO BCOS开源社区](#) 2019-03-01



点击上方蓝字，成为社区一分子



张开翔

FISCO BCOS 首席架构师
联盟链老司机


— AUTHOR — 作者 —

大家好，我是张开翔。

上一篇分享了“[信任区块链时究竟在信任什么？](#)”（还没看的童鞋，点击标题可直达），这次换个角度，漫步月之暗面，谈谈在区块链系统和业务设计时，不信任什么。

先讲结论：几乎什么都不能信！





Don't trust, just verify.

Steven D. Levitt

建立Don't Trust,Just Verify的理念，才是通往区块链世界的正确态度。

——By我随口说的

1

不信任其他节点

////////////////////

区块链节点和其他节点会建立P2P通信，共同组成网络，传递区块、交易、共识信令等各种信息。其他节点可能是由不同的机构、不同的人持有，持有节点的人可能是善意，也可能是恶意。

即使在善意假设时，节点运行存活的健康度也会受运维水平和资源影响，比如处于一个不稳定的网络里，会偶尔挂掉，会抽风乱发消息，或者硬盘满等原因导致数据存储失败，以及出现其他可能的故障。

在恶意假设时，要预设其他节点可能会骗自己或伤害自己，比如传递过来错误的协议包，或者用诡异的指令寻找漏洞进行攻击，或者发起高频垃圾请求，频繁连接然后断开，又或者海量连接占用资源等。

所以节点应该是把自己看成在黑暗丛林里孤身求生存的猎人，必须有“独立自主”、“自给自足”的态度，摆出“不相信其他任何节点”的姿势保护自己。在节点准入时，需要采用证书技术来认证节点身份；在连接控制上，拒绝有异常的连接；采用频率控制对连接次数、请求量等做限制；在协议包格式和指令正确性等方面做验证。自己发出去的信息，不应暴露自己的私有信息，也不期望其他节点一定会给出立刻和正确的响应，必须采用异步处理和校验容错的设计。

..... FISCO BCOS

2

节点和客户端互相不信任

//////////

客户端，指在区块链网络外，向区块链发起请求的模块，如业务使用的java sdk、钱包客户端等。客户端和节点通过网络端口通信。

如果客户端掌握在不受控的人手里，有可能会向节点发起大量的请求，或发送一堆垃圾信息，使节点疲于应对，甚至巧妙地构建漏洞攻击信息，试图越权访问，窃取信息或使节点出错。

同时，从客户端的角度看，节点有可能不响应或响应缓慢，或者返回错误的数据，包括格式错误、状态错误、表示收妥但其实不处理等，甚至别有用心的人会设置一个“假”节点和客户端通信，欺骗客户端。节点做出这些与期望不符的反应，可能使客户端运行出错，功能受损。

为提升节点和客户端的互信，可以为双方分配数字证书，必须通过证书进行双向握手，客户端经过私钥签名才能对节点发起交易类请求，节点应对客户端进行权限控制，拒绝高危的接口调用，不要轻易开放节点管理接口、系统配置接口等。双方对每次通信的数据格式、数据有效性都进行严密校验。

双方在交互时也应该进行频率控制，异步处理，对每一个交互进行结果校验，不能预设对方正确处理，必须获取交易回执和处理结果进行确认。

当认为只和一个节点通信并不能保证安全时，客户端可以采用“f+1查询”的思路，尽可能多地和几个节点通信。如果当前链的共识安全模型是“3f+1”，那么，如果从f+1个节点读到的信息是一致的，结果是可以确认的。

3

不信任区块高度

区块高度是一个非常关键的信息，代表整个链当前的状态。向区块链发送交易、节点间进行共识、对区块和状态的校验等操作都会依赖区块高度。

某个节点在断网或处理速度缓慢时，其区块高度有可能落后于整个链，又或者某个节点恶意伪造数据时，其高度又可能超过整个链。在链出现分叉时，如某一个分叉上的区块高度被另一个分叉超越，落后的分叉就会变得毫无意义。即使在正常的情况下，节点依旧有可能间歇性地落后于整个链一到几个区块，然后在一定时间内才可能追上最新高度。

如在PBFT共识模型里，总数2/3以上节点在同一个高度时，全链就有机会达成共识继续出块。余下的1/3的节点有可能和参与共识的节点高度不同，这时意味着从这个节点读取到的数据，并不是全网最新的数据，只能代表链在该高度时的一个快照。

业务逻辑可以把区块高度做为一个参考值，基于高度做一些判定逻辑，在确定性共识（如PBFT）的链上，采用f+1查询等方法确认链的最新高度，在可能分叉的链上，需要参考“6个区块确认”的逻辑，审慎选取可信的区块高度。

4

不信任交易数据

////

交易（Transaction）代表一方向另一方发起了一个事务请求，交易可能导致资产的转移、改变帐户状态或系统配置，区块链系统通过共识后确认交易，使相关的事务生效。

交易必须带上发送者的数字签名，交易里所有数据字段都必须包含在签名里，未经签名的字段存在被伪造的可能，不予采信。

交易数据在网络上广播时，可以被其他人读取，如交易数据里包含隐私数据，发送者则必须对数据进行脱敏或加密保护。

交易可能因为网络原因被重发，或者被其他人保存下来刻意再次发送，造成交易的“重放”，所以区块链系统必须对交易进行防重，避免出现“双花”。

..... FISCO BCOS

5

不信任状态数据

////

区块链的状态（State）数据是由智能合约运行后生成的，理想情况下，每个节点的合约引擎一致、输入一致、规则一致，那么输出的状态就应该一致。但不同的节点可能安装了不同的软件版本，或者合约引擎的沙盒机制不够严密引入了不确定性因素，甚至被侵入、篡改，或者存在其他莫名其妙的bug，都可能导致合约运行输出结果不一致，那么一致性和事务性就无法得到保障。

状态的校验是成本很高的事情，典型的校验方法是使用MPT（Merkle Patricia Tree）树，把所有

状态都塞到树里管理起来。MPT树可以把所有的状态归结为一个Merkleroot Hash，节点之间在共识过程中确认交易运行后生成的状态树Merkleroot，确保状态一致。

这棵树结构复杂，数据量大，消耗不少的计算和存储资源，很容易就成为了性能瓶颈。所以对状态的校验需要有更快、更简单，且又稳妥的方案，如结合版本验证、增量Hash验证等算法，辅以数据缓存，可减少重复计算和优化IO次数，能在保证一致性、正确性的同时，有效地提升验证效率。



6

不信任私钥持有者



采用私钥对交易以及其他关键操作进行签名，再使用公钥验签，是区块链上最基础的验证逻辑。只要私钥被正确使用，这个逻辑是安全的。

但私钥仅仅是一段数据，只依赖私钥则用户是匿名的。在联盟链面对的场景里，需要使用许可型的身份，首先通过KYC、尽调、权威认证等现实世界的验证方式确认身份，然后将身份和公钥绑定并公示，或者结合PKI体系的数字证书发放公私钥，这样私钥对应的身份是可知、可信、可控的。

私钥可能会因丢失、泄漏而被他人盗用，或者因被遗忘导致资产损失。所以在私钥的保存上，需要考虑采用周全的保护方案，如加密存储、TEE环境、密码卡、USBkey、软硬加密机等方案。在私钥的管理上，则需要考虑密钥丢失后如何安全的重置、找回。

加强版的私钥使用思路有几个，比如使用多签、门限签名等方式，每次交易时必须用多个私钥进行签名，私钥可以保管在不同的地方，安全性高，但技术方案和使用体验复杂。

还有一种是交易私钥和管理私钥分离。交易私钥用于管理资产，管理私钥用于管理个人资料，交易私钥可以被管理私钥重置，管理私钥本身则通过门限、分片等算法，分开存储保管，以备重置

或找回。



7

不信任其他链

在跨链的场景里，每条链有自己的资产、共识，链之间的安全模型变得非常复杂，比如一条链上的记账者串通造假，或者链出现了分叉、区块高度回滚，这时如果链外的其他模块和链有不够严谨的交互，都会造成数据不一致或资产损失。

如果不同的链采用的还是不一样的平台架构，那么在工程上会更加复杂。

跨链、侧链目前依旧是业界在研究和逐步实现的课题，主要目的是解决链和链之间的通信，进行资产锁定和资产交换，保证整个过程的全局一致性、交易事务性，以及抗欺诈。从A链往B链转移一个资产，必须要确保A链上的资产被锁定或销毁，且B链上一定能增加对应的一笔资产，在双方可能分别出现分叉、回滚的时间窗里，要有机制确保双向的资产安全。

在现有跨链的方案里，存在中继、链间HUB等方式，这些系统的设计本身也要达到高度可信可靠的标准，安全等级应不低于甚至高于所对接的链，同样也应采用多中心、群体共识的体系设计，整体复杂度可算是链的N次方了。



8

不信任网络层

区块链节点需要和其他节点发生通信，所以必须在网络上暴露自己的通信端口，如果通过公网通信，那么相当于在公网上暴露了自己，很容易遭到类似渗透、DDOS这样的网络攻击。节点必须在网络层保护自己，包括在网关上设置IP黑白名单、设置端口策略、进行DDOS流量防护，且对网络流量、网络状态进行监测，如果突发网络流量或连接数暴增，说不定，就是被人当肉鸡或者正在脱库进行时了。

非必要端口，切忌对公网开放，如用于做管理监控的RPC端口，只能对机构内部开放，在进行网络策略设定之前，一定要慎之又慎。

..... FISCO BCOS

9

不信任代码

“Code is law”确实是一句响亮的口号，但是在程序员头发掉光之前，他写的代码都可能有bug，只是看写bug快还是修bug快而已。

无论是底层的代码还是智能合约代码，都可能存在技术性 or 逻辑性的坑，但凡代码产生的数据和指令行为，都需要另一段代码对其进行严格地校验，代码本身也需要进行静态和动态扫描，包括采用形式化证明等技术进行全面地审核验证，以检测可能的逻辑错误、安全漏洞或是否有信息泄露。前段时间有一份公布到github上的某酒店系统的代码，居然包括了mysql的连接用户名密码，且数据库端口居然是向公网开放的，这种坑简直不可想象。

开放出去的开源代码，固然可以被人审查、反馈以提升安全性，也可能被人翻找漏洞、随意修改，甚至恶意埋雷。但总的来说，开源还是利大于弊。在开源社区中，开发者会向项目提交PR（Pull Request）。审核PR是很关键也很繁重的工作，值得安排专家并分配大量时间去做审核。有开源项目的老司机透露，其项目核心模块的PR的审核时间长达经年，否则“加了个功能引入

两个bug”那真是得不偿失，更别说如果被植入漏洞埋雷了。

10

不信任记账者

共识的流程大致可以抽象为，选出记账者，记账者发布区块，其他节点校验和确认。公链里记账可以用“挖矿”的方式进行（如比特币），矿工用大量的算力代价为它自己的诚信背书，又或者是用大量的资产权益抵押获得记账权（Pos和DPos等共识）。在联盟链常用的PBFT/Raft等算法里，记账者列表可以是随机或轮换产生，记账者给出提案，其他投票人多步提交，收集投票。按少数服从多数的原则，一般是2/3以上共识节点同意，共识才能达成。

从系统可用性角度看，记账者有可能出错、崩溃，或者运行缓慢，影响整个链的出块。又或者记账者可以只收录手续费高的交易，抛弃一些交易，导致有些交易总是不能达成。有的记账者还可以凭借算力或暗箱运作，进行“预挖”或者“扣块攻击”，破坏博弈关系.....

记账者故障或作恶，超越了共识的安全阈值的话，将直接伤害整条链的价值基础。根据不同的记账模式，记账者需要设计不同的容错、校验、抗欺诈算法，执行激励和惩罚机制，在运行过程中定期检查记账者的健康度，对于无力记账或者作恶的记账节点，全网不接受他们的记账结果，并对其进行惩戒，甚至是踢出网络。

.....

罗列起来还有很多，包括合约、证书、同步等等，每一个模块都有自己的功用和风险点，简直罄

竹难书。总之，区块链做为分布式的多方协作的体系，接入了形形色色参与者，整个体系绝不是单个开发者或运营者所能单点把控，“善意推测”在这个领域已经不尽适用，整个世界步步惊心，处处冷箭，只能通过周密的算法和繁杂的流程维系共识和安全，简而言之，没有经过验证的信息，一个字节都不能相信。

比起单一环境里的软件设计，区块链领域的设计思路确实存在颠覆性，开发者要从“做功能，只容错，不防骗”的思维模式里跳出来，带着“怀疑一切”的态度进行设计。

开发者在面向区块链领域时，不能只是思考怎么实现一个功能，而更要去思考整个流程会不会有出错，会不会被人篡改数据、发掘漏洞、攻击系统、欺诈其他参与者。要换位思考自己所实现的功能，会被别人用什么方式使用，在不同的环境会有什么表现，可能造成什么后果。任何收到的信息，任何流程输入、输出，都必须经过严格地校验才能采信，开发者能做到这一点，才算是打开了区块链新世界的大门，才能在连续剧里至少活到第二集。

分布式算法、对称非对称加密、HASH、证书、安全和隐私等技术在区块链领域大行其道，都是为了在保护信息的同时，给信息加上一层又一层证明和可验证因子，这使得整个系统变得复杂、繁琐，但这是值得的，因为这样才能共同验证，构建“安全”和“信任”。

以上，写给准备跳坑，或已经在坑里的程序员。共勉。



FISCO BCOS的代码完全开源且免费

下载地址↓↓↓（可戳 [阅读原文](#) 直接打开）

<https://github.com/fisco-bcos>



注：文章配图来源于网络渠道，特此鸣谢相关平台及创作者。

[阅读原文](#)