传统的集中式&quot云存储&quot，以及&quot分布式存储网络IPFS区块链&quot可以解决区块链天生难以存储大文件的问题。。本文源于京东金融区块链实验室技术专家李冠南分享的《基于Fabric的存储扩展实践》主题演讲。在实践过程中，他提出了三个设计方案，希望能给你一些启发。

以下是李冠男&#039；s分享，编译。

众所周知，区块链的故事始于2008年中本聪出版《一种点对点的电子现金系统》，随后比特币诞生。后来以太坊出现了。随着市场的热潮，人们发现了区块链技术本身的应用价值。各种项目层出不穷，但其中许多场景都要求区块链具备文件存储能力。

一般来说，大家&#039；的第一反应是，我能在链上存储所有的数据吗？但是现有的主流区块链，比特币就不用说了。在以太网上存储数据非常昂贵。按照燃气数据5Gwei计算，存储1MB数据需要3.76ETH。HyperledgerFabric是一个联盟链，所以在上面持久化数据是可以的。但是现在有一个写死的限制。默认数据小于99M。如果大于99m，其代码需要重新编译。

所以可以看出，把所有的数据都上传是不明智的，没有必要像存储交易数据一样，在每个节点上存储几百兆的数据文件。。所以通常的做法是把文件存储在外链上，把文件的散列存储在外链上，这样文件仍然是集中存储的，比如传统的&quot云存储&quot。

什么是集中式云存储

？

传统云存储允许用户将自己的数据上传到云中。用户上传后，服务提供商将把数据保存在他们的数据中心。这样，无论何时何地，用户想要访问这些信息，只需要向数据中心发送一个请求。，数据中心向用户发送数据。

集中式&quot云存储&quot有以下问题：

典型问题是数据中心都是大型服务器，需要温度控制和严格维护，成本高，延迟大。因为数据中心通常离用户不是很近。有人说CDN可以用，但问题是它的隐私政策是服务商设计的，他们还是有办法接触和分享用户&#039；个人资料，毕竟不透明。而且除了作恶的可能，只要有人为介入。，很可能会出现意想不到的错误。比如员工误删一个数据库的事情并不少见，GitLab事件至今让人记忆犹新&#039；的记忆.采用一个&quot分布式存储网络&quot，所以我们需要采用分布式存储网络。

这项技术并不新鲜。它有很多优点，比如文件大小不限，存储容量不限，成本低，不受地域限制，去中心化。如果应用了特征技术，可以保证其内容不被篡改。

如果&quot分布式存储网络&quot和&quot区块链&quot合并后，能否解决区块链自然难以存储大文件的问题？

我选择了IPFS技术，中文叫星际文件系统。

分布式存储IPFS——What什么？怎么会？什么是

IPFS？

ipfs是一个分布式存储网络；

ipfs是一种对等超媒体协议。目的是让现有网络更快、更安全、更开放；

IPFS有一个疯狂的目标：取代HTTP协议，为每个人建立一个更好的网络；

那么它有什么特点呢？

首先，存储在IPFS上的文件是分散存储的。也就是说，每个文件和所有文件块都有一个唯一的指纹，这个指纹就是一个加密的哈希值。其次，IPFS网络可以自动删除重复文件，并跟踪每个文件的版本历史。。而且每个网络节点只需要存储自己感兴趣的内容和一些索引信息，用来查找谁存储了什么。在查找文件时，可以使用Hash来查询IPFS网络中的哪些节点存储了什么内容。最后每个文件都可以通过分散式命名系统IPNS得到一个友好的名字，而不是一连串的散列。

IPFS技术堆栈分为几层：

当然，最底层是网络，然后是路由、交换层、特定结构层、Merkledag、命名系统，最顶层是应用。与区块链一样，IPFS是技术大师，它借鉴了许多相关技术。下面三层用于传输数据。上面两层是定义数据，最上面一层当然是使用数据。

在介绍完基本工具后，我将正式说明我的尝试。

基于Fabric的存储扩展实践

首先，我定义了三个目标：

。

1。希望Fabric能暴露一组接口，让外面可以用IPFS；通过织物；

2。希望Chaincode在必要的时候也可以直接使用IPFS的功能；

3。希望在这个过程中尽量少修改现有的Fabric代码。换句话说，我希望这是一种系统能力，而不是具体的应用层手段。

目标就在这里，最直接的方法就是利用现有的go-sdk作为粘合剂。，联合布料和IPFS。

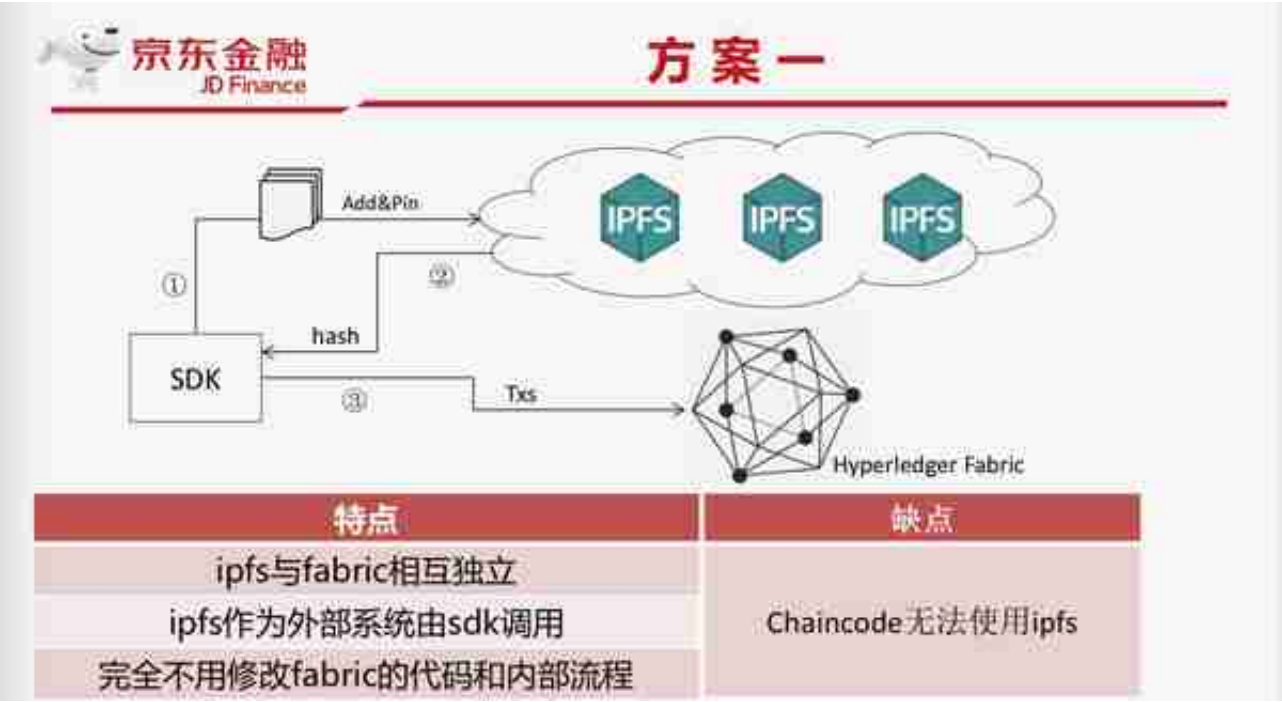
最初的想法是重新开发go-sdk，并封装与IPFS交互的逻辑。事实上，在使用过程中，sdk首先要求归还IPFS。，然后将返回的结果作为事务内容写入Fabric，这与当前的大多数用法相同。

另外，IPFS有几个特点需要注意：首先，它有自己的垃圾收集机制。当你写一个节点的数据时，只有在一个叫做&quotpin&quot能保证不被回收吗？其次，除非发出请求，否则不同节点之间的同步数据不会自动备份。

基于这些思路，第一个方案诞生了：

方案一

。

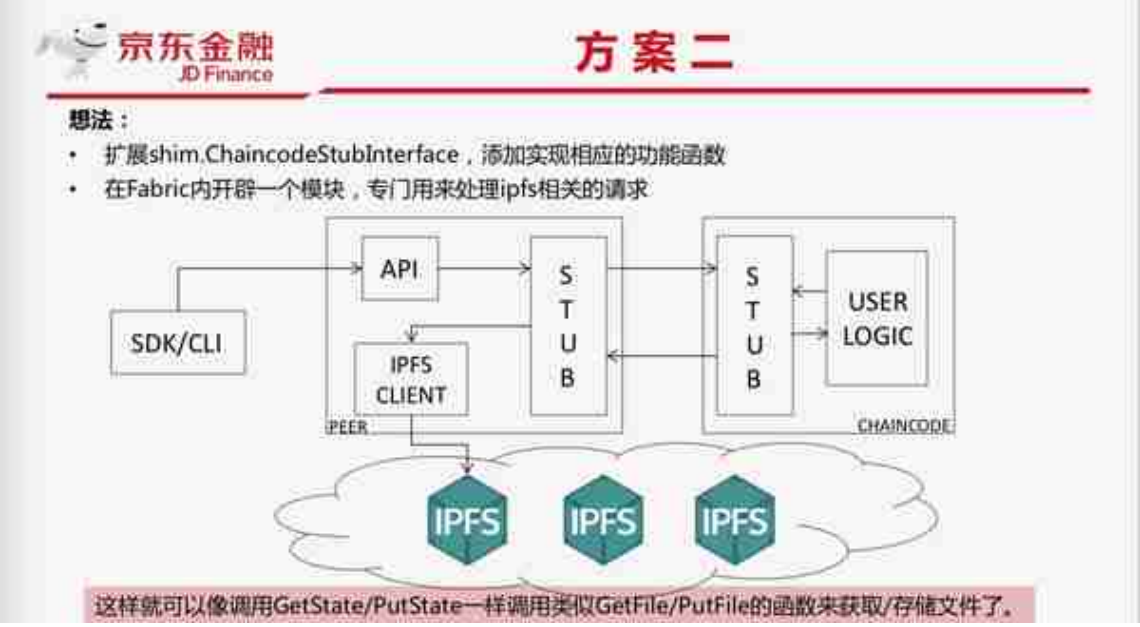


It&#039；这很简单。通过sdk进行改造，用sdk运营IPFS网络。向该文件添加一个pin并获得一个散列返回，然后将散列值封装到一个普通事务中并提交给结构网络。

这个方案有几个明显的特点：第一，IPFS和Fabric相互独立，没有直接的交互作用。IPFS被sdk称为外部系统，其优点是无需修改Fabric代码和内部流程。但缺点是将来，我们将直接使用IPFS。

Let&#039；让我们看看织物的链码是如何工作的。大家都知道，Chaincode运行在docker容器中，它通过GRPC与peer通信。

通常情况下，Chaincode需要先注册，注册后进行初始化，然后才能运行存储信息等操作，之后就是一些判断操作。但是这种交互实际上有一个中间层，通过Shim组件进行交互。Shim中的ChaincodeStubInterface定义了可以在chaincode中使用的函数。于是我萌生了第二个想法。如果我想让chaincode使用IPFS，我将扩展shim来实现它。



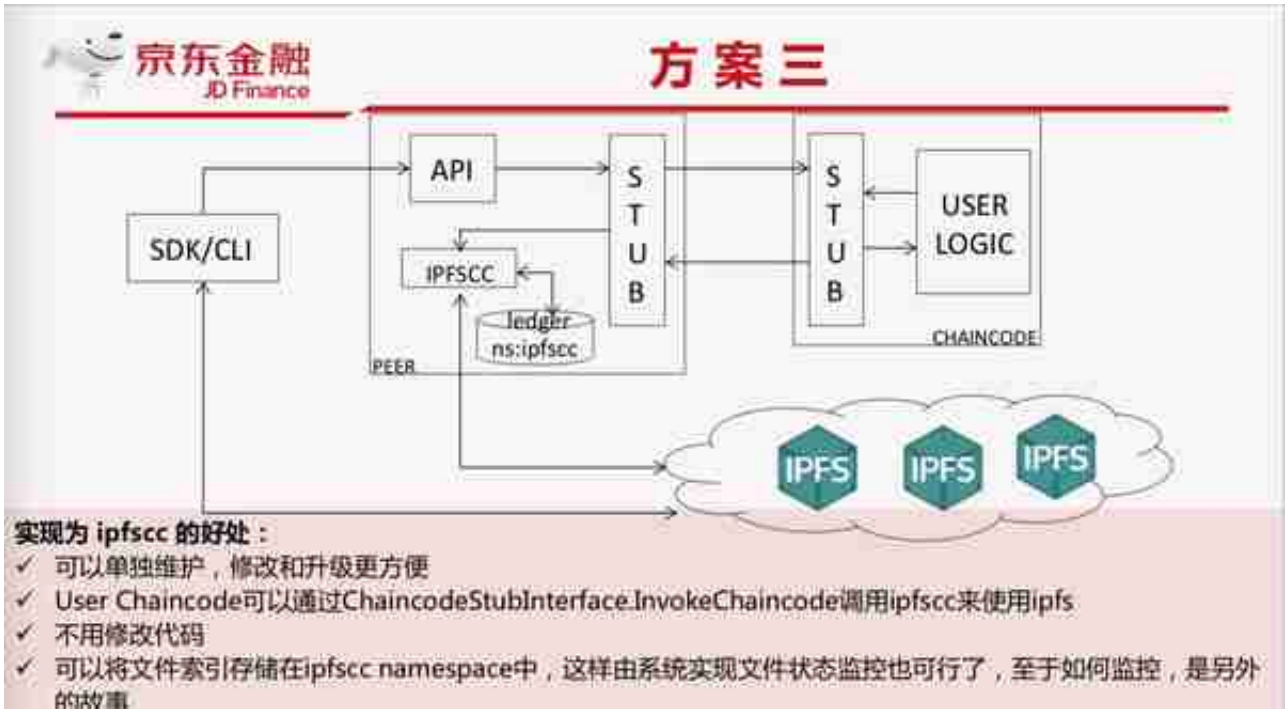
方案二[XY002][XY001]该方案的思路是扩展垫片。ChaincodeStubInterface，并添加相应的函数。仅仅添加功能是远远不够的，还需要在Fabric中打开一个模块来处理与IPFS相关的请求。这样就可以像调用GetState/PutState一样调用GetFile/PutFile这样的函数来获取/存储文件。

在这个方案中，peer是Chaincode的代理。，你可以直接与IPFS网络互动。整个思路其实很简单，但是在真正的实现过程中，我发现并不容易，因为Fabric的很多代码都需要修改，仍然需要应用这个方案来主动维护文件的状态。。IPFS网络节点不

会主动备份这些文件。与第一种方案相比，实现了链码可以直接使用IPFS。虽然缺点很多，但似乎方向应该是对的。

这个时候我才注意到。Fabric1.2有一个很好的特性，也得益于它的架构设计，所有设计都是可插拔的。Fabric1.2中的系统链代码会自动部署到每个新创建的通道中，不同的通道是隔离的。。所以让&#039；s跳出来看看，有哪些方法可以让chaincode使用IPFS，也就是让chaincode调用外部？能否利用Fabric系统的可插拔特性来实现这个功能？在ChaincodeStubInterface看到一个叫InvokeChaincode的函数。这个功能很好，它可以让一个Chaincode调用另一个Chaincode，这让我想起了第三种方案。

方案三



我可以将与IPFS交互的逻辑完全打包到一个系统代码中，然后将它与可插拔功能集成在一起。然后就可以使用InvokeChaincode的函数来获得相应的能力。而且好处多多。第一，可以单独维护。修改升级非常方便。用户Chaincode可以通过我刚才提到的函数调用IPFScc来使用IPFS，它并没有&#039；不需要修改代码，它可以将文件索引存储在IPFScc命名空间中。系统监控文件状态也是可行的。

我其实可以更灵活一些，把第一个方案中的方法整合起来，让sdk开发一套接口，这样就可以和IPFS网交互，也可以把得到的hash上传到账本上。

还有刚才提到的最后一个问题。上传文件后，如果你没有&#039；t操作它，它不

会主动去备份。如何解决这个问题？最后，我使用IPFS集群，它有自己的共识机制。，并能自动备份IPFS网络中的文件，并能灵活配置备份次数。

这样的话，过程就比较清晰了。。我可以通过Sdk/CLI/chaincode上传文件可以根据需要在对等端临时生成文件并上传到IPFS网络，并获取返回的hash，然后使用IPFS-clusterpin文件。成功后，将文件信息和相关IPFS指标信息写入IPFScc的账本。在这种情况下，实际上将打开应用程序空间。

以上是我的实践尝试，这个方案并不完美。