Hyperledger Fabric

Frequently Asked Questions

常见问题

一、Endorsement

一、背书

Endorsement architecture:

背书架构：

Question: How many peers in the network need to endorse a transaction?

问题：网络中有多少对等方需要签署一个事务？

Answer: The number of peers required to endorse a transaction is driven by the endorsement policy that is specified at chaincode deployment time.

答：认可一个事务所需的对等方数量由在链码部署时指定的认可策略驱动。

Question: Does an application client need to connect to all peers?

问题：应用程序客户端是否需要连接到所有对等端？

Answer: Clients only need to connect to as many peers as are required by the endorsement policy for the chaincode.

答：客户机只需要连接到链码的认可策略所要求的尽可能多的对等端。

二、Security & Access Control

二、安全和访问控制

Question:

问题：

How do I ensure data privacy?

如何确保数据隐私？

Answer:

答：

There are various aspects to data privacy. First, you can segregate your network into channels, where each channel represents a subset of participants that are authorized to see the data for the chaincodes that are deployed to that channel.

数据隐私有很多方面。首先，您可以将您的网络隔离到通道中，其中每个通道代表一个子集，这些参与者被授权查看部署到该通道的链码的数据。

Second, you can use private-data to keep ledger data private from other organizations on the channel. A private data collection allows a defined subset of organizations on a channel the ability to endorse, commit, or query private data without having to create a separate channel. Other participants on the channel receive only a hash of the data. For more information refer to the Using Private Data in Fabric tutorial. Note that the key concepts topic also explains

when to use private data instead of a channel.
第二，您可以使用私有数据来保持来自渠道上其他组织的分类帐数据私有。私有数据收集允许在一个通道上定义的组织子集在不需要创建单独通道的情况下认可、提交或查询私有数据。通道上的其他参与者只接收一个散列数据。有关更多信息，请参阅“在结构中使用私有数据”教程。注意，关键概念主题还解释了何时使用私有数据而不是通道。
Third, as an alternative to Fabric hashing the data using private data, the client application can hash or encrypt the data before calling chaincode. If you hash the data then you will need to provide a means to share the source data. If you encrypt the data then you will need to provide a means to share the decryption keys.
第三，作为使用私有数据对数据进行结构散列的替代方法，客户机应用程序可以在调用 chaincode 之前散列或加密数据。如果散列数据，则需要提供共享源数据的方法。如果加密数据，则需要提供共享解密密钥的方法。
Fourth, you can restrict data access to certain roles in your organization, by building access control into the chaincode logic.
第四，通过将访问控制构建到链码逻辑中，您可以限制对组织中某些角色的数据访问。
Fifth, ledger data at rest can be encrypted via file system encryption on the peer, and data in-transit is encrypted via TLS.
第五，静态的分类账数据可以通过对等机上的文件系统加密进行加密，传输中的数据通过 TLS 进行加密。
Question:
问题：
Do the orderers see the transaction data?
订购者看到交易数据了吗？
Answer:
答：
No, the orderers only order transactions, they do not open the transactions. If you do not want the data to go through the orderers at all, then utilize the private data feature of Fabric. Alternatively, you can hash or encrypt the data in the client application before calling chaincode. If you encrypt the data then you will need to provide a means to share the decryption keys.
不，订购方只订购交易，他们不打开交易。如果您根本不希望数据通过订购者，那么使用结构的私有数据特性。或者，您可以在调用 chaincode 之前在客户机应用程序中散列或加密数据。如果加密数据，则需要提供共享解密密钥的方法。

三、Application-side Programming Model

三、应用端编程模型

Question:
问题：
How do application clients know the outcome of a transaction?
应用程序客户端如何知道事务的结果？
Answer:
答：

The transaction simulation results are returned to the client by the endorser in the proposal response. If there are multiple endorsers, the client can check that the responses are all the same, and submit the results and endorsements for ordering and commitment. Ultimately the committing peers will validate or invalidate the transaction, and the client becomes aware of the outcome via an event, that the SDK makes available to the application client.
交易模拟结果由背书人在提案响应中返回给客户。如果有多个背书人，客户可以检查回复是否一致，并提交结果和背书以供订购和承诺。最终，提交对等端将验证或使事务无效，并且客户机通过一个事件了解到结果，而该事件是 SDK 向应用程序客户机提供的。
Question:
问题：
How do I query the ledger data?
如何查询分类帐数据？
Answer:
答：
Within chaincode you can query based on keys. Keys can be queried by range, and composite keys can be modeled to enable equivalence queries against multiple parameters. For example a composite key of (owner,asset_id) can be used to query all assets owned by a certain entity. These key-based queries can be used for read-only queries against the ledger, as well as in transactions that update the ledger.
在 chaincode 中,您可以基于键进行查询。可以按范围查询键,并且可以对组合键进行建模,以便对多个参数进行等价查询。例如，可以使用（owner，asset_id）的复合键查询某个实体拥有的所有资产。这些基于键的查询可用于对分类帐以及更新分类帐的交易记录进行只读查询。
If you model asset data as JSON in chaincode and use CouchDB as the state database, you can also perform complex rich queries against the chaincode data values, using the CouchDB JSON query language within chaincode. The application client can perform read-only queries, but these responses are not typically submitted as part of transactions to the ordering service.
如果您在 chaincode 中将资产数据建模为 json，并使用 couchdb 作为状态数据库，那么您还可以使用 chaincode 中的 couchdb json 查询语言对 chaincode 数据值执行复杂的丰富查询。应用程序客户机可以执行只读查询，但这些响应通常不会作为事务的一部分提交给订购服务。
Question:
问题：
How do I query the historical data to understand data provenance?
如何查询历史数据以了解数据来源？
Answer:
答：
The chaincode API GetHistoryForKey() will return history of values for a key.
chaincode api getHistoryForkey（）将返回键值的历史记录。
Question:
问题：

How to guarantee the query result is correct, especially when the peer being queried may be recovering and catching up on block processing?
如何保证查询结果的正确性，特别是当被查询的对等机可能正在恢复和追赶块处理时？
Answer:
答：
The client can query multiple peers, compare their block heights, compare their query results, and favor the peers at the higher block heights.
客户端可以查询多个对等点，比较它们的块高度，比较它们的查询结果，并在更高的块高度上支持对等点。

四、Chaincode (Smart Contracts and Digital Assets)

四、链码（智能合约和数字资产）

Question:
问题：
Does Hyperledger Fabric support smart contract logic?
Hyperledger 结构是否支持智能合约逻辑？
Answer:
答：
Yes. We call this feature Chaincode. It is our interpretation of the smart contract method/algorithm, with additional features.
对。我们称这个特性为链码。这是我们对智能合约方法/算法的解释，具有附加功能。
A chaincode is programmatic code deployed on the network, where it is executed and validated by chain validators together during the consensus process. Developers can use chaincodes to develop business contracts, asset definitions, and collectively-managed decentralized applications.
chain code 是部署在网络上的编程代码,在协商一致的过程中由链验证器一起执行和验证。开发人员可以使用链代码来开发业务合同、资产定义和集中管理的分散应用程序。
Question:
问题：
How do I create a business contract?
如何创建业务合同？
Answer:
答：
There are generally two ways to develop business contracts: the first way is to code individual contracts into standalone instances of chaincode; the second way, and probably the more efficient way, is to use chaincode to create decentralized applications that manage the life cycle of one or multiple types of business contracts, and let end users instantiate instances of contracts within these applications.
通常有两种方法来开发业务合同：第一种方法是将单个合同编码为链码的独立实例；第二种方法，可能也是更有效的方法，是使用链码创建分散的应用程序，以管理一种或多种类型的业务合同的生命周期，并让最终用户在这些应用程序中实例化契约的实例。
Question:

问题：
How do I create assets?
如何创建资产？
Answer:
答：
Users can use chaincode (for business rules) and membership service (for digital tokens) to design assets, as well as the logic that manages them.
用户可以使用链式代码（用于业务规则）和成员服务（用于数字令牌）来设计资产，以及管理资产的逻辑。
There are two popular approaches to defining assets in most blockchain solutions: the stateless UTXO model, where account balances are encoded into past transaction records; and the account model, where account balances are kept in state storage space on the ledger.
在大多数区块链解决方案中，有两种常用的定义资产的方法：无状态 utxo 模型，其中账户余额编码为过去的交易记录;以及账户模型,其中账户余额保存在分类账的状态存储空间中。
Each approach carries its own benefits and drawbacks. This blockchain technology does not advocate either one over the other. Instead, one of our first requirements was to ensure that both approaches can be easily implemented.
每种方法都有其自身的优点和缺点。这种区块链技术既不提倡一种又不提倡另一种。相反，我们的首要要求之一是确保这两种方法都可以很容易地实现。
Question:
问题：
Which languages are supported for writing chaincode?
哪些语言支持编写链码？
Answer:
答：
Chaincode can be written in any programming language and executed in containers. Currently, Golang, node.js and java chaincode are supported.
链码可以用任何编程语言编写并在容器中执行。目前，Golang、NoDE.JS 和 Java 链码都得到了支持。
It is also possible to build Hyperledger Fabric applications using Hyperledger Composer.
也可以使用 HyperledgeComposer 构建 HyperledgeFabric 应用程序。
Question:
问题：
Does the Hyperledger Fabric have native currency?
超级账本织物有本国货币吗？
Answer:
答：
No. However, if you really need a native currency for your chain network, you can develop your own native currency with chaincode. One common attribute of native currency is that some amount will get transacted (the chaincode defining that currency will get called) every time a transaction is processed on its chain.
不需要。但是，如果您的链网络确实需要一种本国货币，您可以使用链码开发自己的本国货

币。原生货币的一个常见属性是，每次在其链上处理事务时，都会处理一些金额（定义该货币的链码将被调用）。

五、Differences in Most Recent Releases

五、最新版本的差异

Question: Where can I find what are the highlighted differences between releases?

问题：在哪里可以找到版本之间突出显示的差异？

Answer: The differences between any subsequent releases are provided together with the Releases.

答：任何后续版本之间的差异都随版本一起提供。

Question: Where to get help for the technical questions not answered above?

问题：对于上述未回答的技术问题，从何处获得帮助？

Answer: Please use StackOverflow.

答：请使用 stackoverflow。

六、Ordering Service

六、订购服务

Question: I have an ordering service up and running and want to switch consensus algorithms. How do I do that?

问题：我有一个正在运行的订购服务，我想切换共识算法。我该怎么做？

Answer: This is explicitly not supported.

答：这是明确不支持的。

Question: What is the orderer system channel?

问：订购方系统的渠道是什么？

Answer: The orderer system channel (sometimes called ordering system channel) is the channel the orderer is initially bootstrapped with. It is used to orchestrate channel creation. The orderer system channel defines consortia and the initial configuration for new channels. At channel creation time, the organization definition in the consortium, the /Channel group’s values and policies, as well as the /Channel/Orderer group’s values and policies, are all combined to form the new initial channel definition.

答：订购方系统通道（有时称为订购系统通道）是订购方最初引导使用的通道。它用于协调通道创建。订购方系统通道定义新通道的联合体和初始配置。在通道创建时，联合体中的组织定义、/通道组的值和策略以及/通道/订购方组的值和策略都将组合在一起，形成新的初始通道定义。

Question: If I update my application channel, should I update my orderer system channel?

问题：如果我更新我的应用程序频道，我应该更新我的订购者系统频道吗？

Answer: Once an application channel is created, it is managed independently of any other channel (including the orderer system channel). Depending on the modification, the change may or may not be desirable to port to other channels.

In general, MSP changes should be synchronized across all channels, while policy changes are more likely to be specific to a particular channel.
答：一旦创建了应用程序通道，它将独立于任何其他通道（包括订购方系统通道）进行管理。根据修改情况，可能需要或不需要更改以连接到其他通道。一般来说，MSP 更改应该在所有通道之间同步，而策略更改更可能是特定于特定通道的。
Question: Can I have an organization act both in an ordering and application role?
问题：我可以让一个组织同时扮演订购和应用角色吗？
Answer: Although this is possible, it is a highly discouraged configuration. By default the /Channel/Orderer/BlockValidation policy allows any valid certificate of the ordering organizations to sign blocks. If an organization is acting both in an ordering and application role, then this policy should be updated to restrict block signers to the subset of certificates authorized for ordering.
答：虽然这是可能的，但这是一种非常不鼓励的配置。默认情况下，/channel/order/blockvalidation 策略允许订购组织的任何有效证书对块进行签名。如果一个组织同时扮演排序和应用程序角色，那么应该更新此策略，以将块签名者限制为授权排序的证书的子集。
Question: I want to write a consensus implementation for Fabric. Where do I begin?
问题：我想为 Fabric 编写一个共识实现。我从哪里开始？
Answer: A consensus plugin needs to implement the Consenter and Chain interfaces defined in the consensus package. There are two plugins built against these interfaces already: solo and kafka. You can study them to take cues for your own implementation. The ordering service code can be found under the orderer package.

答：共识插件需要实现共识包中定义的同意者和链接口。针对这些接口已经构建了两个插件：Solo 和 Kafka。您可以研究它们，为自己的实现提供线索。订购服务代码可以在订购程序包下找到。
Question: I want to change my ordering service configurations, e.g. batch timeout, after I start the network, what should I do?
问题：我想更改我的订购服务配置，例如批量超时，在我启动网络后，我应该怎么做？
Answer: This falls under reconfiguring the network. Please consult the topic on configtxlator.
答：这属于重新配置网络。请参考 configtxlator 的主题。

1、Solo

1、Solo

Question: How can I deploy Solo in production?
问题：如何在生产中部署 Solo？
Answer: Solo is not intended for production. It is not, and will never be, fault tolerant.
答：Solo 不适合制作。它不是，也永远不会是容错的。

2、Kafka

2、Kafka

Question:
问题：
How do I remove a node from the ordering service?
如何从订购服务中删除节点？
Answer:
答：
This is a two step-process:
这是一个两步的过程：
Add the node’s certificate to the relevant orderer’s MSP CRL to prevent peers/clients from connecting to it.
将节点的证书添加到相关订购方的 MSP CRL，以防止对等方/客户端连接到该节点。
Prevent the node from connecting to the Kafka cluster by leveraging standard Kafka access control measures such as TLS CRLs, or firewalling.
通过利用标准的 kafka 访问控制措施（如 tls-crls 或防火墙），防止节点连接到 kafka 集群。
Question: I have never deployed a Kafka/ZK cluster before, and I want to use the Kafka-based ordering service. How do I proceed?
问题：我以前从未部署过 kafka/zk 集群，我想使用基于 kafka 的订购服务。如何继续？
Answer: The Hyperledger Fabric documentation assumes the reader generally has the operational expertise to setup, configure, and manage a Kafka cluster (see Caveat emptor). If you insist on proceeding without such expertise, you should complete, at a minimum, the first 6 steps of the Kafka Quickstart guide before experimenting with the Kafka-based ordering service. You can also consult this sample configuration file for a brief explanation of the sensible defaults for Kafka/ZooKeeper.
答：HyperledgeFabric 文档假定读者通常具有设置、配置和管理 Kafka 集群的操作专业知识（请参见 Caveat Emptor）。如果您坚持在没有此类专业知识的情况下继续操作，则在尝试基于 Kafka 的订购服务之前，至少应完成《Kafka 快速入门指南》的前 6 个步骤。您还可以参考这个示例配置文件，以获得对 Kafka/ZooKeeper 的合理默认值的简要说明。
Question: Where can I find a Docker composition for a network that uses the Kafka-based ordering service?
问题：在哪里可以找到使用基于 Kafka 的订购服务的网络的 Docker 组合？
Answer: Consult the end-to-end CLI example.
答：请参阅端到端的 CLI 示例。
Question: Why is there a ZooKeeper dependency in the Kafka-based ordering service?
问：为什么在基于卡夫卡的订购服务中存在 ZooKeeper 管理员依赖关系？
Answer: Kafka uses it internally for coordination between its brokers.
答：卡夫卡内部使用它来协调经纪人之间的关系。
Question: I’m trying to follow the BYFN example and get a “service

unavailable” error, what should I do?
问题：我试图遵循 byfn 示例，并得到一个“服务不可用”错误，我应该怎么做？
Answer: Check the ordering service’s logs. A “Rejecting deliver request because of consenter error” log message is usually indicative of a connection problem with the Kafka cluster. Ensure that the Kafka cluster is set up properly, and is reachable by the ordering service’s nodes.

答：查看订购服务日志。“由于同意者错误而拒绝传递请求”日志消息通常表示 Kafka 集群存在连接问题。确保 kafka 集群设置正确，并且可以通过订购服务的节点访问。

3、BFT

3、BFT

Question: When is a BFT version of the ordering service going to be available?
问：订购服务的 BFT 版本何时可用？
Answer: No date has been set. We are working towards a release during the 1.x cycle, i.e. it will come with a minor version upgrade in Fabric. Track FAB-33 for updates.
答：没有设定日期。我们正致力于在 1.x 周期内发布一个版本，也就是说，它将在结构中进行一个小版本升级。跟踪 FAB-33 以获取更新。