使用configtx.yaml创建通道配置

通过构建指定通道的初始配置的通道创建交易来创建通道。**通道配置**存储在账本中，并管理所有添加到通道的后续块。通道配置指定哪些组织是通道成员，可以在通道上添加新块的排序节点，以及管理通道更新的策略。可以通过通道配置更新来更新存储在通道创世块中的初始通道配置。如果足够多的组织批准通道更新，则在将新通道配置块提交到通道后，将对其进行管理。

虽然可以手动构建通道创建交易文件，但使用`configtx.yaml`文件和configtxgen工具可以更轻松地创建通道。`configtx.yaml`文件包含以易于理解和编辑的格式构建通道配置所需的信息。`configtxgen`工具读取`configtx.yaml`文件中的信息，并将其写入Fabric可以读取的protobuf格式。

概览

您可以使用本教程来学习如何使用`configtx.yaml`文件来构建存储在创世块中的初始通道配置。本教程将讨论由文件的每个部分构建的通道配置部分。

- Organizations
- Capabilities
- Application
- Orderer
- Channel
- Profiles

由于文件的不同部分共同创建用于管理通道的策略，因此我们将在独立的教程中讨论通道策略。

在创建通道教程的基础上，我们将使用`configtx.yaml`文件作为示例来部署Fabric测试网络。在本地计算机上打开命令终端，然后导航到本地Fabric示例中的`test-network`目录：

```
cd fabric-samples/test-network
```

测试网络使用的`configtx.yaml`文件位于`configtx`文件夹中。在文本编辑器中打开该文件。在本教程的每一节中，您都可以参考该文件。您可以在Fabric示例配置中找到`configtx.yaml`文件的更详细版本。

Organizations

通道配置中包含的最重要信息是作为通道成员的组织。每个组织都由MSP ID和通道MSP标识。通道MSP存储在通道配置中，并包含用于标识组织的节点，应用程序和管理员的证书。`configtx.yaml`文件的**Organizations**部分用于为通道的每个成员创建通道MSP和随附的MSP ID。

测试网络使用的`configtx.yaml`文件包含三个组织。可以添加到应用程序通道的两个组织是Peer组织Org1和Org2。OrdererOrg是一个Orderer组织，是排序服务的管理员。因为最佳做法是使用不同的证书颁发机构来部署Peer节点和Orderer节点，所以即使组织实际上是由同一公司运营，也通常将其称为Peer组织或Orderer组织。

您可以在下面看到`configtx.yaml`的一部分，该部分定义了测试网络的Org1：

```
- &Org1
    # DefaultOrg defines the organization which is used in the sampleconfig
    # of the fabric.git development environment
    Name: Org1MSP

    # ID to load the MSP definition as
    ID: Org1MSP

    MSPDir: ../organizations/peerOrganizations/org1.example.com/msp

    # Policies defines the set of policies at this level of the config tree
    # For organization policies, their canonical path is usually
    #   /Channel/<Application|Orderer>/<OrgName>/<PolicyName>
    Policies:
        Readers:
            Type: Signature
            Rule: "OR('Org1MSP.admin', 'Org1MSP.peer', 'Org1MSP.client')"
        Writers:
            Type: Signature
            Rule: "OR('Org1MSP.admin', 'Org1MSP.client')"
        Admins:
            Type: Signature
            Rule: "OR('Org1MSP.admin')"
        Endorsement:
            Type: Signature
            Rule: "OR('Org1MSP.peer')"

    # leave this flag set to true.
    AnchorPeers:
        # AnchorPeers defines the location of peers which can be used
        # for cross org gossip communication.  Note, this value is only
        # encoded in the genesis block in the Application section context
        - Host: peer0.org1.example.com
          Port: 7051
```

- `Name`字段是用于标识组织的非正式名称。
- `ID`字段是组织的MSP ID。MSP ID充当组织的唯一标识符，并且由通道策略引用，并包含在提交给通道的交易中。
- `MSPDir`是组织创建的MSP文件夹的路径。`configtxgen`工具将使用此MSP文件夹来创建通道MSP。该MSP文件夹需要包含以下信息，这些信息将被传输到通道MSP并存储在通道配置中：
  - 一个CA根证书，为组织建立信任根。CA根证书用于验证应用程序，节点或管理员是否属于通道成员。
  - 来自TLS CA的根证书，该证书颁发了Peer节点或Orderer节点的TLS证书。TLS根证书用于通过Gossip协议标识组织。
  - 如果启用了Node OU，则MSP文件夹需要包含一个`config.yaml`文件，该文件根据x509证书的OU标识管理员，节点和客户端。
  - 如果未启用Node OU，则MSP需要包含一个admincerts文件夹，其中包含组织管理员身份的签名证书。

用于创建通道MSP的MSP文件夹仅包含公共证书。如此一来，您可以在本地生成MSP文件夹，然后将MSP发送到创建通道的组织。

- `Policies`部分用于定义一组引用通道成员的签名策略。我们将在讨论通道策略时更详细地讨论这些策略。
- `AnchorPeers`字段列出了组织的锚节点。为了利用诸如私有数据和服务发现之类的功能，锚节点是必需的。建议组织选择至少一个锚节点。虽然组织可以使用`configtxgen`工具首次选择其通道上的锚节

点，但建议每个组织都使用`configtxlator`工具来设置锚节点来更新通道配置。因此，该字段不是必须的。

Capabilities

Fabric通道可以由运行不同版本的Hyperledger Fabric的Orderer节点和Peer节点加入。通道功能通过仅启用某些功能，允许运行不同Fabric二进制文件的组织参与同一通道。例如，只要通道功能级别设置为V1_4_X或更低，则运行Fabric v1.4的组织和运行Fabric v2.x的组织可以加入同一通道。所有通道成员都无法使用Fabric v2.0中引入的功能。

在`configtx.yaml`文件中，您将看到三个功能组：

- **Application**功能可控制Peer节点使用的功能，例如Fabric链码生命周期，并设置可以由加入通道的Peer运行的Fabric二进制文件的最低版本。
- **Orderer**功能可控制Orderer节点使用的功能，例如Raft共识，并设置可通过Orderer属于通道共识者集合的节点运行的Fabric二进制文件的最低版本。
- **Channel**功能设置可以由Peer节点和Orderer节点运行的Fabric的最低版本。由于Fabric测试网络的所有Peer和Orderer节点都运行版本v2.x，因此每个功能组均设置为`V2_0`。因此，运行Fabric版本低于v2.0的节点不能加入测试网络。有关更多信息，请参见capabilities概念主题。

Application

Application部分定义了控制Peer组织如何与应用程序通道交互的策略。这些策略控制需要批准链码定义或给更新通道配置的请求签名的Peer组织的数量。这些策略还用于限制对通道资源的访问，例如写入通道账本或查询通道事件的能力。

测试网络使用Hyperledger Fabric提供的默认Application策略。如果您使用默认策略，则所有Peer组织都将能够读取数据并将数据写入账本。默认策略还要求大多数通道成员给通道配置更新签名，并且大多数通道成员需要批准链码定义，然后才能将链码部署到通道。本部分的内容在通道策略教程中进行了更详细的讨论。

Orderer

每个通道配置都在通道共识者集合中包括Orderer节点。共识者集合是一组排序节点，它们能够创建新的块并将其分发给加入该通道的Peer节点。在通道配置中存储作为共识者集合的成员的每个Orderer节点的端点信息。

测试网络使用`configtx.yaml`文件的**Orderer**部分来创建单节点Raft 排序服务。

- `OrdererType`字段用于选择Raft作为共识类型：

```
OrdererType: etcdraft
```

Raft 排序服务由可以参与共识过程的共识者列表定义。因为测试网络仅使用一个Orderer节点，所以共识者列表仅包含一个端点：

```
EtcdRaft:
    Consenters:
    - Host: orderer.example.com
      Port: 7050
      ClientTLSCert: ../organizations/ordererOrganizations/example.com/orderers/orderer.example.c
      ServerTLSCert: ../organizations/ordererOrganizations/example.com/orderers/orderer.example.c
    Addresses:
    - orderer.example.com:7050
```

共识者列表中的每个Orderer节点均由其端点地址以及其客户端和服务器TLS证书标识。如果要部署多节点排序服务，则需要提供主机名，端口和每个节点使用的TLS证书的路径。您还需要将每个排序节点的端点地址添加到`Addresses`列表中。

- 您可以使用`BatchTimeout`和`BatchSize`字段通过更改每个块的最大大小以及创建新块的频率来调整通道的延迟和吞吐量。
- `Policies`部分创建用于管理通道共识者集合的策略。测试网络使用Fabric提供的默认策略，该策略要求大多数Orderer管理员批准添加或删除Orderer节点，组织或对分块切割参数进行更新。

因为测试网络用于开发和测试，所以它使用由单个Orderer节点组成的排序服务。生产中部署的网络应使用多节点排序服务以确保安全性和可用性。要了解更多信息，请参阅配置和操作Raft排序服务。

Channel

通道部分定义了用于管理最高层级通道配置的策略。对于应用程序通道，这些策略控制哈希算法，用于创建新块的数据哈希结构以及通道功能级别。在系统通道中，这些策略还控制Peer组织的联盟的创建或删除。

测试网络使用Fabric提供的默认策略，该策略要求大多数排序服务管理员需要批准对系统通道中这些值的更新。在应用程序通道中，更改需要获得大多数Orderer组织和大多数通道成员的批准。大多数用户不需要更改这些值。

Profiles

`configtxgen`工具读取**Profiles**部分中的通道配置文件以构建通道配置。每个配置文件都使用YAML语法从文件的其他部分收集数据。`configtxgen`工具使用此配置为应用程序通道创建通道创建交易，或为系统通道写入通道创世块。要了解有关YAML语法的更多信息，Wikipedia提供了一个良好的入门指南。

测试网络使用的`configtx.yaml`包含两个通道配置文件`TwoOrgsOrdererGenesis`和`TwoOrgsChannel`：

TwoOrgsOrdererGenesis

`TwoOrgsOrdererGenesis`配置文件用于创建系统通道创世块：

```
TwoOrgsOrdererGenesis:
    <<: *ChannelDefaults
    Orderer:
        <<: *OrdererDefaults
        Organizations:
            - *OrdererOrg
        Capabilities:
```

```
            <<: *OrdererCapabilities
    Consortiums:
        SampleConsortium:
            Organizations:
                - *Org1
                - *Org2
```

系统通道定义了排序服务的节点以及排序服务管理员的组织集合。系统通道还包括一组属于区块链联盟的Peer组织。联盟中每个成员的通道MSP都包含在系统通道中，从而允许他们创建新的应用程序通道并将联盟成员添加到新通道中。

该配置文件创建一个名为`SampleConsortium`的联盟，该联盟在`configtx.yaml`文件中包含两个Peer组织Org1和Org2。配置文件的`Orderer`部分使用文件的**Orderer:**部分中定义的单节点Raft 排序服务。**Organizations:**部分中的OrdererOrg成为排序服务的唯一管理员。因为我们唯一的Orderer节点正在运行Fabric 2.x，所以我们可以将Orderer系统通道功能设置为`V2_0`。系统通道使用**Channel**部分中的默认策略，并启用`V2_0`作为通道功能级别。

TwoOrgsChannel

测试网络使用`TwoOrgsChannel`配置文件创建应用程序通道：

```
TwoOrgsChannel:
    Consortium: SampleConsortium
    <<: *ChannelDefaults
    Application:
        <<: *ApplicationDefaults
        Organizations:
            - *Org1
            - *Org2
        Capabilities:
            <<: *ApplicationCapabilities
```

排序服务将系统通道用作创建应用程序通道的模板。在系统通道中定义的排序服务的节点成为新通道的默认共识者集合，而排序服务的管理员则成为该通道的Orderer管理员。通道成员的通道MSP从系统通道转移到新通道。创建通道后，可以通过更新通道配置来添加或删除Orderer节点。您还可以更新通道配置以将其他组织添加为通道成员。

`TwoOrgsChannel`提供了测试网络系统通道托管的联盟名称`SampleConsortium`。因此，`TwoOrgsOrdererGenesis`配置文件中定义的排序服务成为通道共识者集合。在`Application`部分中，来自联盟的两个组织Org1和Org2均作为通道成员包括在内。通道使用`V2_0`作为应用程序功能，并使用**Application**部分中的默认策略来控制Peer组织如何与通道进行交互。应用程序通道还使用**Channel**部分中的默认策略，并启用`V2_0`作为通道功能级别。