策略

受众： 架构师，应用和智能合约开发者，管理员

本主题将包含：

什么是策略

从根本上来说，策略是一组规则，用来定义如何做出决策和实现特定结果。为此，策略一般描述了**谁**和**什么**，比如一个人对**资产**访问或者权限。我们可以看到，在我们的日常生活中策略也在保护我们的资产数据，比如汽车租金、健康、我们的房子等。

例如，购买保险时，保险策略定义了条件、项目、限制和期限。该策略经过了策略持有者和保险公司的一致同意，定义了各方的权利和责任。

保险策略用于风险管理，在 Hyperledger Fabric 中，策略是基础设施的管理机制。Fabric 策略表示成员如何同意或者拒绝网络、通道或者智能合约的变更。策略在网络最初配置的时候由联盟成员一致同意，但是在网络演化的过程中可以进行修改。例如，他们定义了从通道中添加或者删除成员的标准，改变区块格式或者指定需要给智能合约背书的组织数量。所有这些定义谁可以干什么的行为都在策略中描述。简单来说，你在 Fabric 网络中的所有想做的事情，都要受到策略的控制。

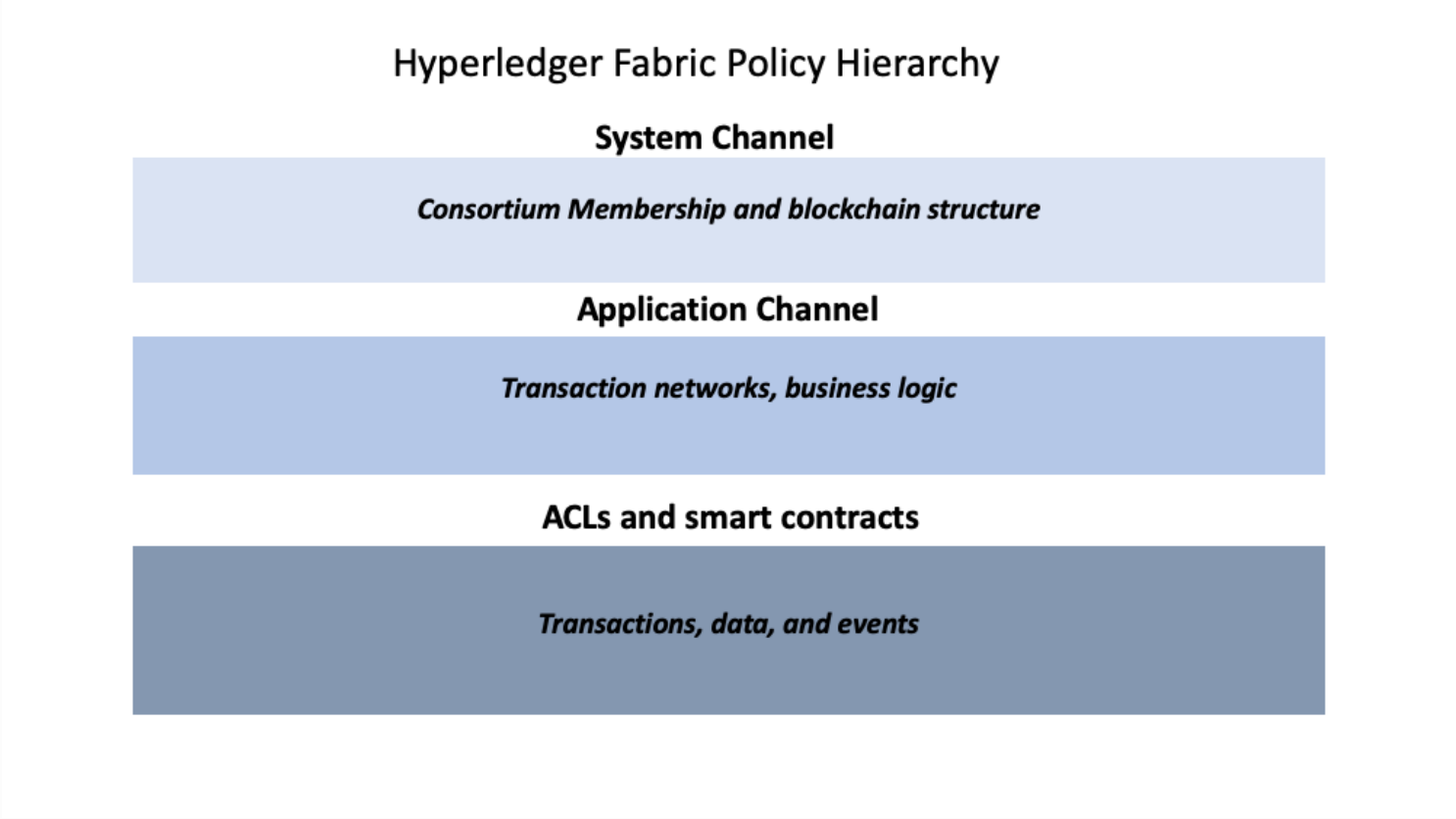
为什么需要策略

策略是使 Hyperledger Fabric 不同于其他区块链系统（比如 Ethereum 或者 Bitcoin）的内容之一。在其他系统中，交易可以在网络中的任意节点生成和验证。治理网络的策略可以在任何时间及时修复，并且只可以使用和治理代码相同的方式进行变更。因为 Fabric 是授权区块链，用户由底层基础设施识别，所以用户可以在启动前决定网络的治理方式，以及改变正在运行的网络的治理方式。

策略决定了那些组织可以访问或者更新 Fabric 网络，并且提供了强制执行这些决策的机制。策略包含了有权访问给定资源的组织列表，比如一个用户或者系统链码。他们同样指定了需要多少组织同意更新资源的提案，比如通道或者智能合约。一旦策略被写入，他们就会评估交易和提案中的签名，并验证签名是否满足网络治理规则。

Fabric是如何实现策略的

策略实现在 Fabric 网络的不同层次。每个策略域都管理着网络操作的不同方面。



Fabric 策略层级图。

系统通道配置

每个网络都从排序服务**系统通道**开始。网络中必须有至少一个排序服务的排序系统通道，它是第一个被创建的通道。该通道也包含着谁是排序服务（排序服务组织）以及在网络中交易（联盟组织）的成员。

排序系统通道配置区块中的策略治理着排序服务使用的共识，并定义了新区块如何被创建。系统通道也治理着联盟中的哪些成员可以创建新通道。

应用通道配置

应用 *通道* 用于向联盟中的组织间提供私有通信机制。

应用通道中的策略治理着从通道中添加和删除成员的能力。应用通道也治理着使用 Fabric 链码生命周期在链码定义和提交到通道前需要哪些组织同意。当系统通道初始创建时，它默认继承了排序系统通道的所有排序服务参数。同时，这些参数（包括治理它们的策略）可以被每个通道自定义。

权限控制列表（ACL）

网络管理员可能对 Fabric 中 ACL 的使用更感兴趣，ACL 通过将资源和已有策略相关联的方式提供了资源访问配置的能力。“资源”可以是系统链码中的方法（例如，“qscc”中的“GetBlockByNumber”）或者其他资源（例如，谁可以获取区块事件）。ACL 参考应用通道配置中定义的策略并将它们扩展到了其他资源的控制。Fabric ACL 的默认集合在 `configtx.yaml` 文件的 `Application: &ApplicationDefaults` 部分，但是它们可以也应该在生产环境中被重写。`configtx.yaml` 中定义的资源列表是 Fabric 当前定义的所有内部资源的完整集合。

该文件中，ACL 以如下格式表示：

```
# ACL policy for chaincode to chaincode invocation
peer/ChaincodeToChaincode: /Channel/Application/Writers
```

`peer/ChaincodeToChaincode` 表示该资源是被保护的，相关的交易必须符合 `/Channel/Application/Writers` 引用侧策略才能被认为是有效的。

关于 ACL 更深入的信息，请参考操作指南中的 ACL 主题。

智能合约背书策略

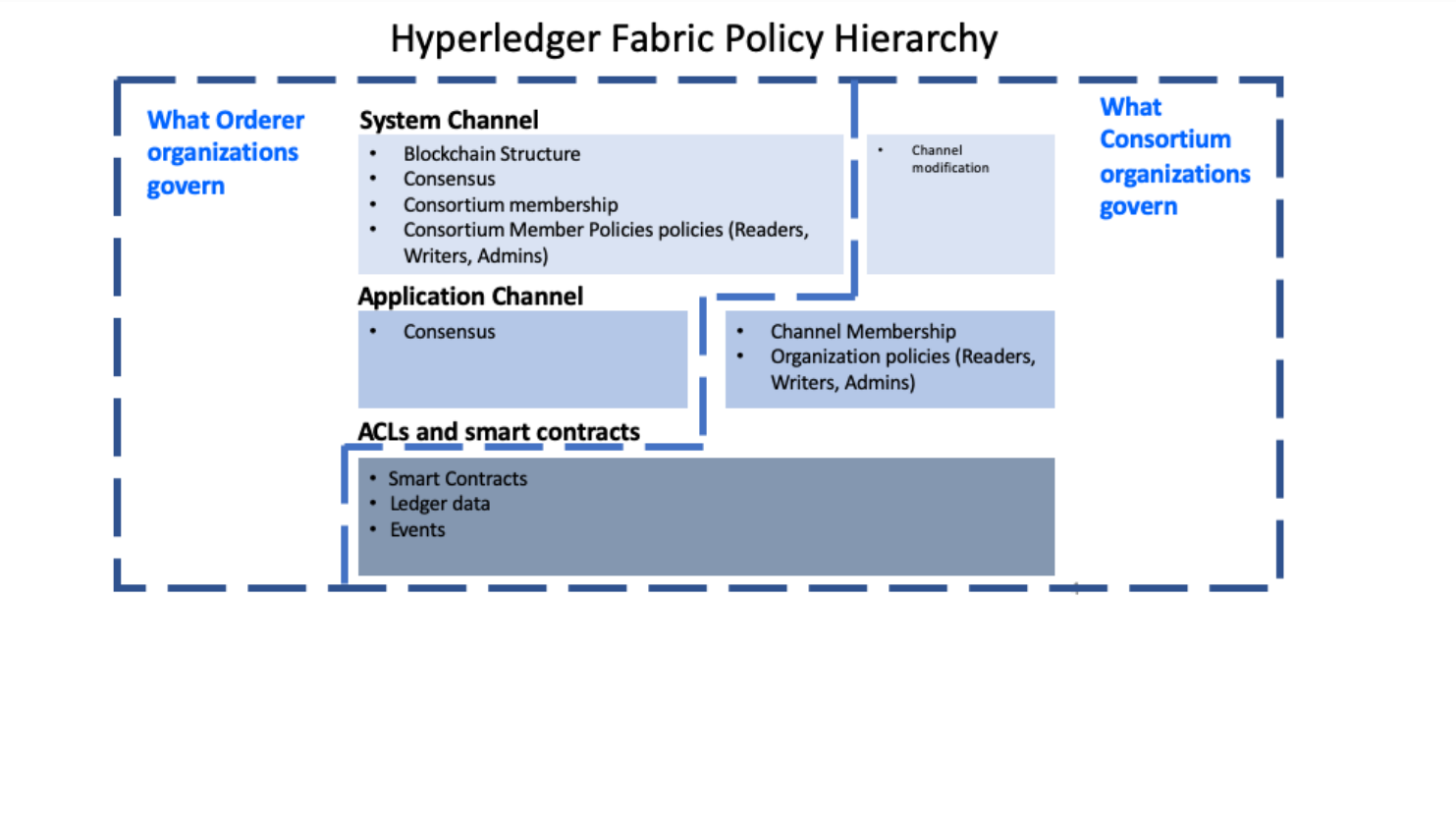
链码包中的每一个智能合约都有一个背书策略，该策略指明了需要通道中多少不同组织的成员根据指定智能合约执行和验证交易才能使一笔交易有效。因此，背书策略定义了必须“背书”（批准）提案执行的组织（的 Peer 节点）。

修改策略

还有一个对 Fabric 的策略工作有重要作用的策略类型，`修改（Modification）策略`。修改策略指明了需要签名所有配置 *更新* 的一组身份。它是定义如何更新策略的策略。因此，每个通道配置元素都包含这一个治理它的变更的策略的引用。

策略作用域

虽然 Fabric 的策略很灵活地配置以适应网络需要，但是策略的结构天然地隔离了由不同排序服务组织或者不同联盟成员治理的域。下边的图中，你可以看到默认策略是如何实现对 Fabric 策略域的控制的。



排序组织和联名组织治理的策略域的详细视图。

一个完整的 Fabric 网络可以由许多不同职能的组织组成。通过支持排序服务创建者建立初始规则和联盟成员的方式，域提供了向不同组织扩展不同的优先级和角色的能力。还支持联盟中的组织创建私有应用通道、治理他们自己的商业逻辑以及限制网络中数据的访问权限。

系统通道配置和每个应用通道配置部分提供了排序组织对哪些组织是联盟成员、区块如何分发到通道以及排序服务节点使用的共识机制的控制。

系统通道配置为联盟成员提供了创建通道的能力。应用通道和 ACL 是联盟组织用来从通道中添加或删除成员以及限制通道中智能合约和数据访问的机制。

在Fabric中如何写策略

如果你想修改 Fabric 的任何东西，和资源相关的策略都描述了**谁**需要批准它，可以是来自个人的一个显式签名，也可以是组的一个隐式签名。在保险领域，一个明确的签名可以是业主保险代理集团中的一员。而一个隐含的签名类似于需要业主保险代理集团中的大多数管理成员批准。这很重要，因为集团中的成员可以在不更新策略的情况下变动。在 Hyperledger Fabric 中，策略中明确的签名使用 `Signature` 语法，隐含的签名使用 `ImplicitMeta` 语法。

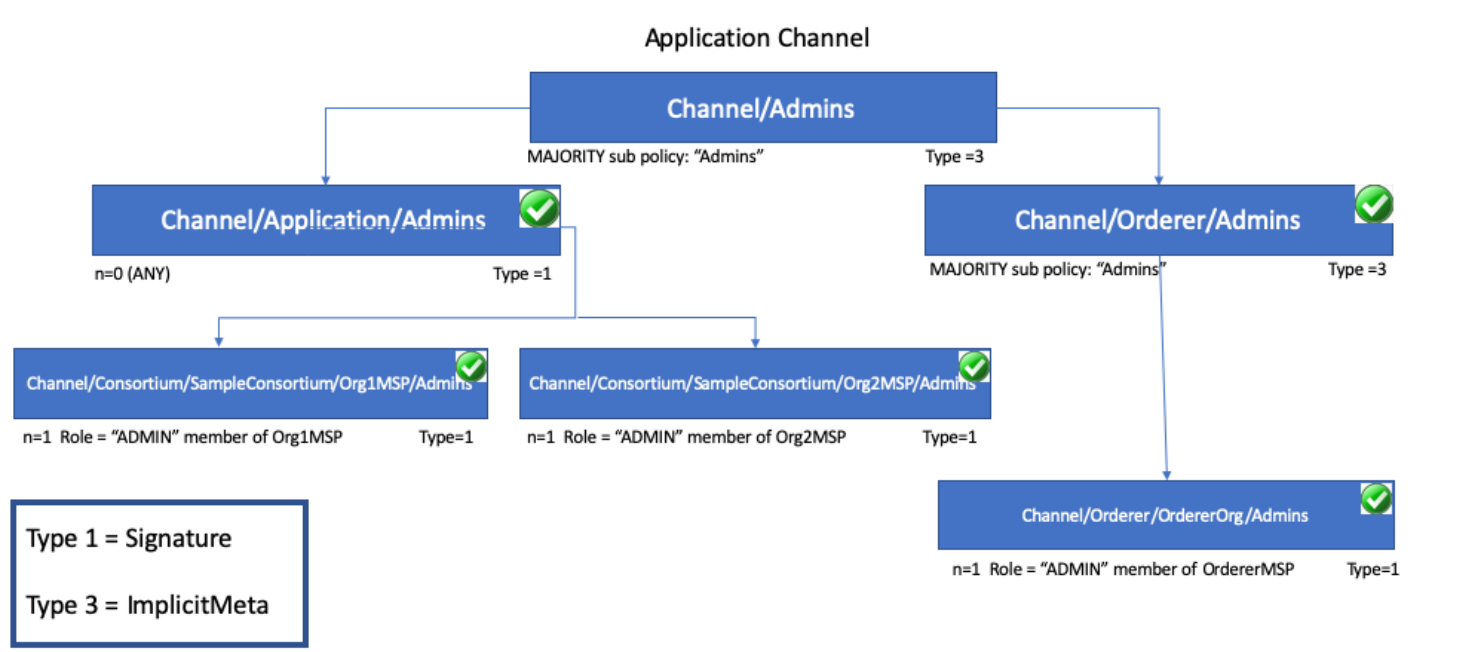
签名策略

`Signature` 策略定义了要满足策略就必须签名的特定用户类型，比如 `Org1.Peer OR Org2.Peer`。策略是很强大的，应为它可以构造复杂的规则，比如“组织 A 和 2 个其他管理员，或者 6 个组织的管理员中的 5 个”。语法支持 `AND`、 `OR` 和 `NOutOf` 的任意组合。例如，一个策略可以简单表达为使用 `AND (Org1, Org2)` ，表示满足该策略就同时需要 Org1 中的一个成员和 Org2 中的一个成员的签名。

隐元（ImplicitMeta）策略

`隐元`策略只在通道配置上下文中有效，通道配置在配置树策略中是基于分层的层次结构。隐元策略聚合了由签名策略最终定义的配置树深层的结果。它们是`隐藏的`，因为它们基于通道配置中的当前组织隐式构建，它们是`元信息`，因为它们的评测不依赖于特定 MSP 规范，而是依赖于配置树中它们的其他子策略。

下边的图例说明了应用通道分层的策略结构，并演示了`隐元`通道配置管理策略（称为 `/Channel/Admins`）是如何处理的，也就是说，当满足配置层级中它的 `Admins` 子策略时，就代表也满足了其子策略的子策略条件。



正如你在上图看到的，`隐元`策略，Type = 3，使用了一种不同的语法`"<ANY|ALL|MAJORITY> <SubPolicyName>"`，例如：

```
`MAJORITY sub policy: Admins`
```

上边的图表展示了一个在配置树中所有`Admins`策略都引用了的`Admin`子策略。你可以创建你自己的子策略并随意命名，并且可以定义在你的每一个组织中。

正如上边提到的，`隐元`策略比如`MAJORITY Admins`的主要优势在于当你向通道添加新组织的时候，你不必更新通道策略。因此`隐元`策略就像联盟成员变更一样灵活。联盟中成员的新增或者退出只要联盟成员一致同意即可，不需要更新策略。重申一下，`隐元`策略最终处理的是如图所示的配置树中它们之下的`签名`子策略。

你也可以定义一个应用级别的隐策略来进行跨组织操作，例如在通道中，需要 ANY （任意）、 ALL （全部）或者 MAJORITY （大多数）组织来满足。这个格式有更好、更自然的默认值，因此组织可以决定有效背书的含义。

你可以通过在组织定义中引入`NodeOUs`来实现进一步的粒度和控制。OU （Organization Units，组织单元）定义在 Fabric CA 客户端配置文件中，当创建身份的时候就会与之关联。在 Fabric 中，`NodeOUs`提供为数字证书层级分类的功能。例如，一个指定了`NodeOUs`的组织可以让一个 'Peer' 签名合法背书，或者组织也可以简单设置为任何成员都可以签名。

示例：通道配置策略

背书策略的理解要从`configtx.yaml`开始，`configtx.yaml`里边定义了通道策略。我们可以查看 BYFN （first-network） 教程中的`configtx.yaml`来查看这两种策略语法类型的示例。请导航至 fabric-

samples/first-network 目录查看 BYFN 中的 configtx.yaml 文件。

文件的第一部分（Organizations）定义了网络中的组织。在每个组织的定义中设置了默认策略，`Readers, Writers, Admins, and Endorsement`，但是你可以任意定义策略命名。每个策略都有一个 `Type` 和 `Rule`，`Type` 描述了策略的表达式类型（`Signature` 或 `ImplicitMeta`）。

下边的 BYFN 示例展示了组织 `Org1` 在系统通道中的定义，其中策略的 `Type` 是 `Signature` 背书策略规则定义为 `"OR('Org1MSP.peer')"`，表示需要 `Org1MSP` 成员中的 peer 来签名。正是这些签名策略形成了隐元策略指向的子策略。

▶ **Click here to see an example of an organization defined with signature policies**

The next example shows the `ImplicitMeta` policy type used in the `Application` section of the `configtx.yaml`. These set of policies lie on the `/Channel/Application/` path. If you use the default set of Fabric ACLs, these policies define the behavior of many important features of application channels, such as who can query the channel ledger, invoke a chaincode, or update a channel config. These policies point to the sub-policies defined for each organization. The Org1 defined in the section above contains `Reader`, `Writer`, and `Admin` sub-policies that are evaluated by the `Reader`, `Writer`, and `Admin` `ImplicitMeta` policies in the `Application` section. Because the test network is built with the default policies, you can use the example Org1 to query the channel ledger, invoke a chaincode, and approve channel updates for any test network channel that you create.

▶ **Click here to see an example of ImplicitMeta policies**

Fabric链码生命周期

Fabric 2.0 发布版本中，介绍了一个新的链码生命周期过程，这是一个在网络中更民主的治理链码的过程。新的过程允许多个组织在链码应用到通道之前如何操作进行投票。这个很重要，因为这是新生命周期过程和策略的融合，策略是在过程中指定的决定着网络的安全性。关于该流程的更多细节在 操作者的链码 教程中，但是为了本主题的目的，你应该理解策略在流程中的使用。新的流程指定策略包含两步，当链码被组织成员**批准**的时候，以及当它被**提交**到通道后。

`configtx.yaml` 文件中 `Application` 部分包含了默认的链码生命周期背书策略。在生产环境中你应该为你的用例自定义这个。

```
################################################################################
#
#   SECTION: Application
#
#   - This section defines the values to encode into a config transaction or
#   genesis block for application related parameters
#
################################################################################
Application: &ApplicationDefaults

    # Organizations is the list of orgs which are defined as participants on
    # the application side of the network
    Organizations:

    # Policies defines the set of policies at this level of the config tree
    # For Application policies, their canonical path is
    #   /Channel/Application/<PolicyName>
    Policies:
```

```
        Readers:
            Type: ImplicitMeta
            Rule: "ANY Readers"
        Writers:
            Type: ImplicitMeta
            Rule: "ANY Writers"
        Admins:
            Type: ImplicitMeta
            Rule: "MAJORITY Admins"
        LifecycleEndorsement:
            Type: ImplicitMeta
            Rule: "MAJORITY Endorsement"
        Endorsement:
            Type: ImplicitMeta
            Rule: "MAJORITY Endorsement"
```

- `LifecycleEndorsement` 策略控制需要谁 *批准链码定义* 。
- `Endorsement` 策略是 *链码的默认背书策略* 。更多细节请继续阅读。

链码背书策略

当使用 Fabric 链码生命周期**链码**被批准并提交到通道时会指定一个背书策略（这个背书策略会覆盖与该链码相关的所有状态）。背书策略可以引用通道配置中的背书策略或者明确指定签名策略。

如果在批准阶段没有明确指明背书策略，就默认使用 `Endorsement` 策略 `"MAJORITY Endorsement"`，意味着要想使交易生效就需要大多数不同通道成员（组织）的执行并验证交易。默认策略允许加入通道的组织自动加入链码背书策略。如果你不想使用默认背书策略，你可以使用签名策略格式来指定更复杂的背书策略（这样就需要链码先被通道中的一个组织签名，然后让其他组织签名）。

签名策略也允许你包含`主角（principals）`，这是匹配角色和身份的一种简单方式。主角类似用户 ID 或者 组 ID，但是更广泛，因为它们可以包含更大范围演员身份的属性，比如演员的组织、组织单元、角色，甚至演员指定的身份。我们讨论的主角是决定他们权限的属性。主角被描述为 'MSP.ROLE'，`MSP` 表示需要的 MSP ID（组织），`ROLE` 表示一下四种可接受的角色之一：Member、 Admin、 Client 和 Peer。角色在用户使用 CA 登记（enroll）的时候与之关联。你可以在 Fabric CA 中自定义可用的角色列表。

一些有效的主角：

- 'Org0.Admin': Org0 MSP 的一个管理员
- 'Org1.Member': Org1 MSP 的一个成员
- 'Org1.Client': Org1 MSP 的一个客户端
- 'Org1.Peer': Org1 MSP 的一个 Peer 节点
- 'OrdererOrg.Orderer': OrdererOrg MSP 的一个排序节点

有一些场景可能需要一些特殊的状态（特殊的键-值对，或这其他的）有不同的背书策略。**基于状态的背书**可以指定与默认链码级别背书策略不同的键的背书策略。

如何写一个背书策略的更多信息请参考操作指南中的 背书策略 主题。

注意： 不同版本 Fabric 的策略有所不同：

- 在 Fabric 2.0 之前，链码的背书策略可以在链码实例化或者使用链码生命周期命令时更新。如果没有在实例化时指明，默认背书策略是“通道中组织的任意成员”。例如。在有 “Org1” 和 “Org2” 的通道中，将有一个 “OR(‘Org1.member’, ‘Org2.member’)” 的默认策略。
- 从 Fabric 2.0 开始，Fabric 提出了一个新的链码生命周期过程，允许多组织同意在链码应用到通道之前如何操作。新的过程需要组织同意链码定义的参数，比如名字、版本以及链码背书策略。

覆盖策略定义

Hyperledger Fabric 包含了用于快速入门、开发、测试去快来你的默认策略，但是在生产环境中需要自定义。你应该留意 `configtx.yaml` 文件中的默认策略。通道配置策略可以使用任意单词扩展，不仅仅是 `configtx.yaml` 中的 `Readers、 Writers、 Admins`。当你编辑 `configtx.yaml` 文件重写了排序系统通道或这指定通道的默认策略并提交了配置更新，就会覆盖排序系统和应用通道的默认策略。

更多信息请查阅更新通道配置。