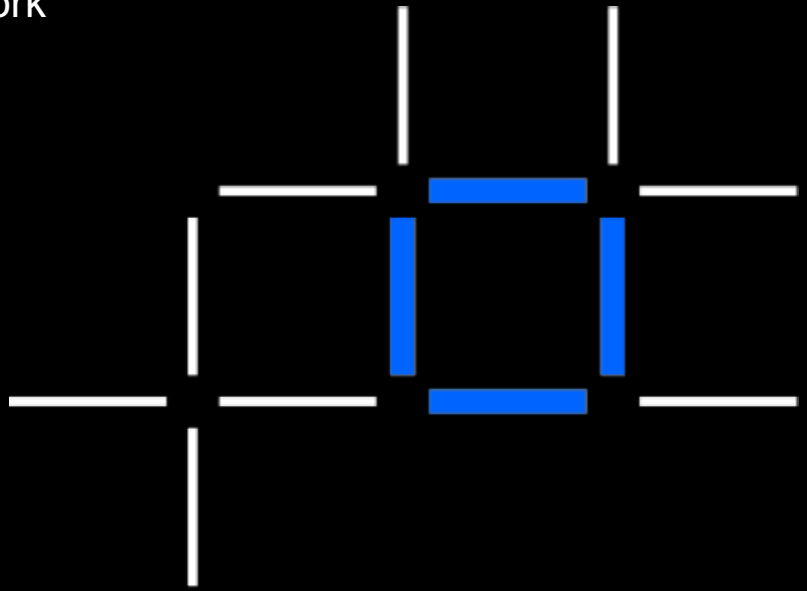
Blockchain Explored, Part 3

Permission and privacy in a Hyperledger Fabric Network



Barry Silliman
IBM Washington Systems Center
silliman@us.ibm.com

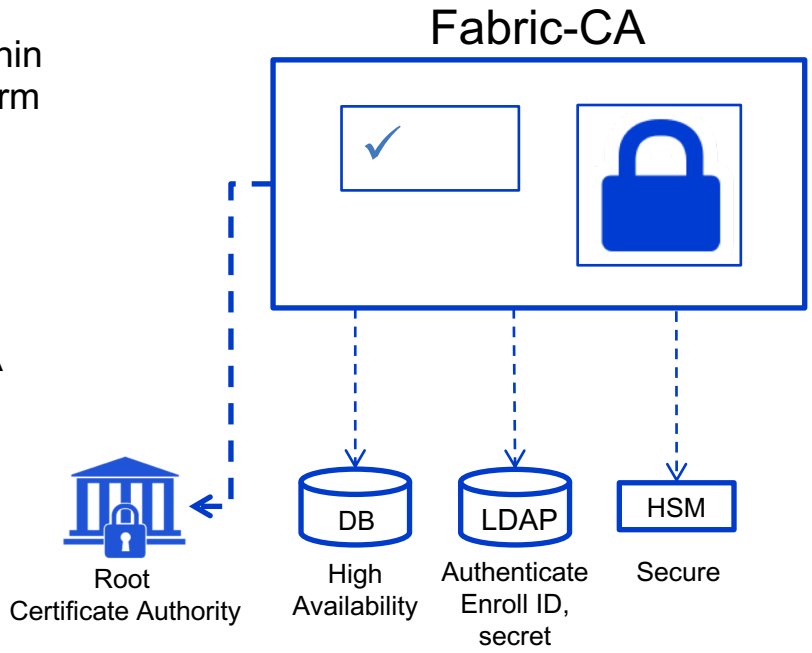
V4.07, 1 March 2019

IBM **Blockchain**

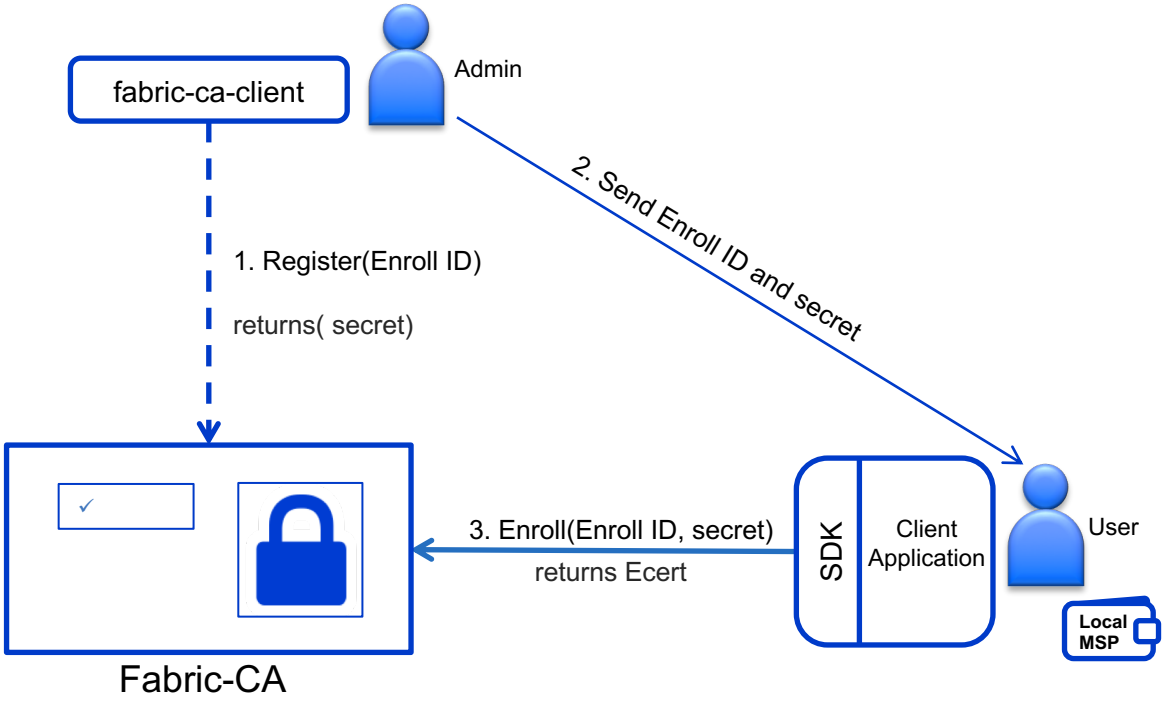
IBM

Fabric-CA

- Default (optional) Certificate Authority within Fabric network for issuing Ecerts (long-term identity)
- Supports clustering for HA characteristics
- Supports LDAP for user authentication
- Supports HSM for security
- Can be configured as an intermediate CA



New User Registration and Enrollment



Registration and Enrollment

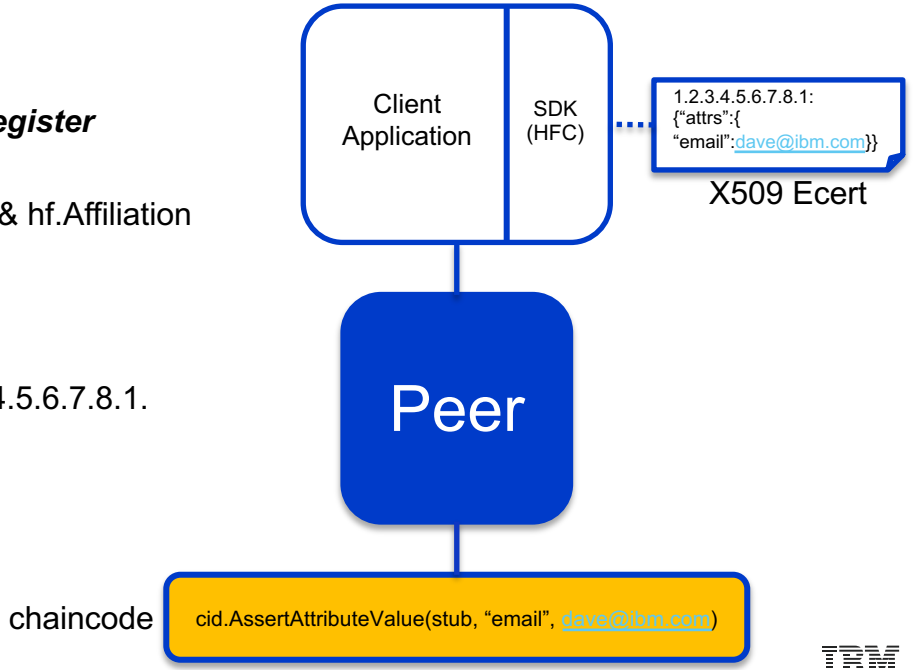
- Admin registers new user with Enroll ID
- User enrolls and receives credentials
- Additional offline registration and enrollment options available

Attribute Based Access Control

Include identity attributes in enrollment certificates for chaincode

- Include attributes in **X509 enrollment certificates** (Ecerts)
- Defined as name/value pairs: email=dave@ibm.com
- Define mandatory and optional attributes with ***fabric-ca-client register***
- Specify attribute values with ***fabric-ca-client enroll***
- Ecerts automatically include attributes: hf.EnrollmentID, hf.Type & hf.Affiliation
- API provided by Client Identity chaincode Library:
  - cid.GetAttributeValue(stub, "attr1")
  - cid.AssertAttributeValue(stub, "myapp.admin", "true")
- Stored as an extension in the Ecert with an ASN.1 OID of 1.2.3.4.5.6.7.8.1.

```
1.2.3.4.5.6.7.8.1:
    {"attrs":{"attr1":"val1"}}
```



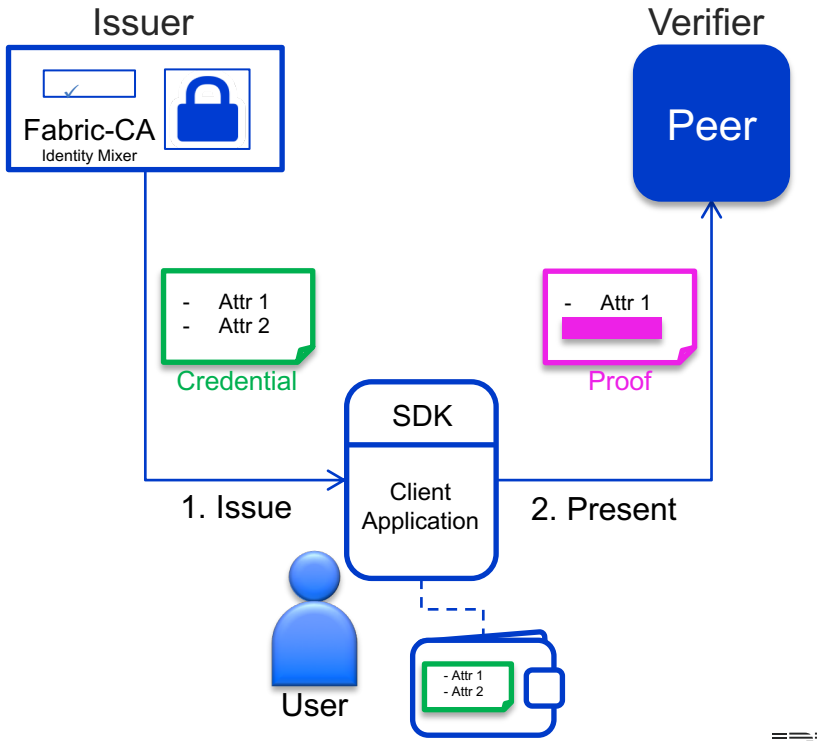
https://jira.hyperledger.org/browse/FAB-5346 + https://jira.hyperledger.org/browse/FAB-6089

Identity Mixer

Fabric includes Identity Mixer for anonymity/unlinkability of a user's identity

Support for a user's identity to be hidden but verifiable, and selective disclosure of user attributes through zero-knowledge-proofs

- **Issuer:** Issues user's identity in the form of an identity mixer *credential*. Includes all attributes associated with the user.
- **User:** Generates *proofs* from their *credential* with selectively disclosed attributes using zero-knowledge-proof. These *proofs* do not disclose the user's true identity and are unlinkable as each proof is different.
- **Verifier:** Verifies the *proof* based on the public certificate of the issuer.

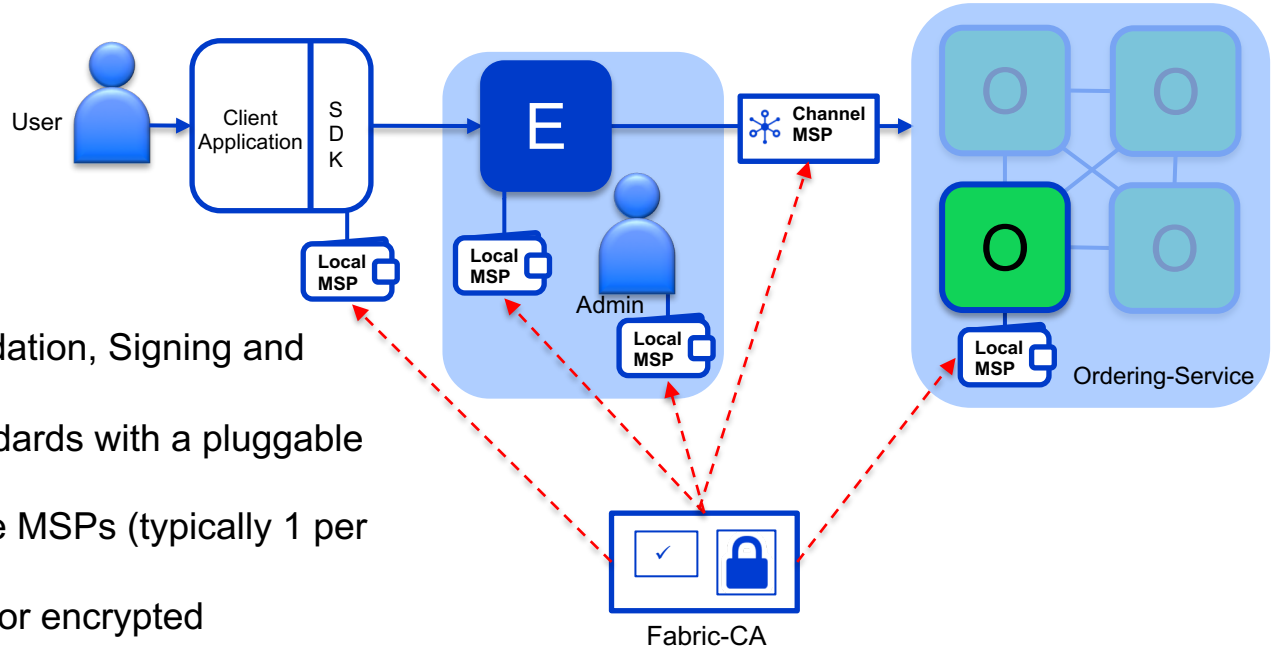


https://jira.hyperledger.org/browse/FAB-10120

Membership Services Provider - Overview

A MSP manages a set of identities within a distributed Fabric network

- Provides identity for:
  - Peers and Orderers
  - Client Applications
  - Administrators
- Identities can be issued by:
  - Fabric CA
  - An external CA
- Provides: Authentication, Validation, Signing and Issuance
- Supports different crypto standards with a pluggable interface
- A network can include multiple MSPs (typically 1 per org)
- Includes TLS crypto material for encrypted communications

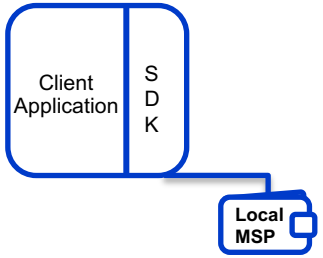


User Identities

Each client application has a local MSP to store user identities

- Each local MSP includes:
  - **Keystore**
    - **Private key** for signing transactions
  - **Signcert**
    - **Public x.509 certificate**

- May also include TLS credentials
- Can be backed by a Hardware Security Module (HSM)



user@org1.example.com	
keystore	<private key>
signcert	user@org1.example.com-cert.pem

Admin Identities

Each Administrator has a local MSP to store their identity

- Each local MSP includes:
  - **Keystore**
    - **Private key** for signing transactions
  - **Signcert**
    - **Public x.509 certificate**
- May also include TLS credentials
- Can be backed by a Hardware Security Module (HSM)

admin@org1.example.com	
keystore	<private key>
signcert	admin@org1.example.com-cert.pem

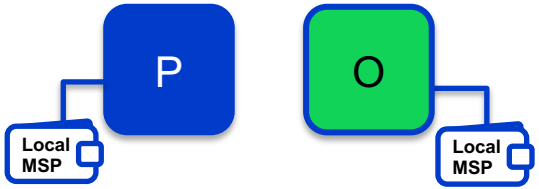
Peer and Orderer Identities

Each peer and orderer has a local MSP

- Each local MSP includes:
  - **keystore**
    - **Private key** for signing transactions
  - **signcert**
    - **Public x.509 certificate**

- In addition Peer/Orderer MSPs identify authorized administrators:
  - **admincerts**
    - List of **administrator certificates**
  - **cacerts**
    - The **CA public cert** for verification
  - **crls**
    - List of **revoked certificates**

- Peers and Orderers also receive channel MSP info
- Can be backed by a Hardware Security Module (HSM)

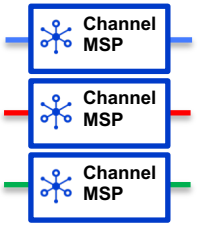


peer@org1.example.com	
admincerts	admin@org1.example.com-cert.pem
cacerts	ca.org1.example.com-cert.pem
keystore	<private key>
signcert	peer@org1.example.com-cert.pem
crls	<list of revoked admin certificates>

Channel MSP information

Channels include additional organisational MSP information

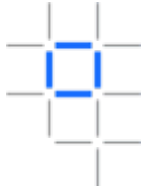
- Determines which orderers or peers can join the channel
- Determines client applications read or write access to the channel
- Stored in configuration blocks in the ledger
- Each channel MSP includes:
  - **admincerts**
    - Any public certificates for administrators
  - **cacerts**
    - The CA public certificate for this MSP
  - **crls**
    - List of revoked certificates
- Does not include any private keys for identity



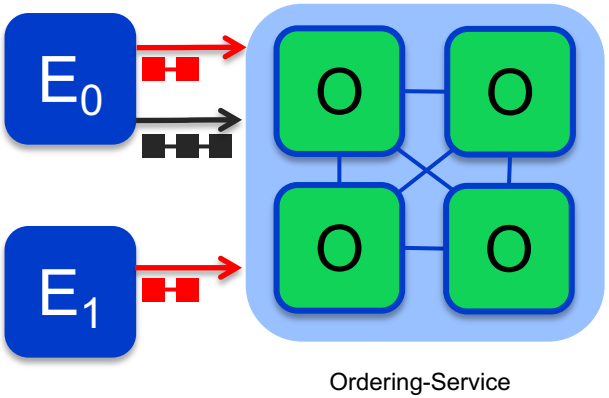
Channels

ID = MSP1	
admincerts	admin.org1.example.com-cert.pem
cacerts	ca.org1.example.com-cert.pem
crls	<list of revoked admin certificates>

Channels

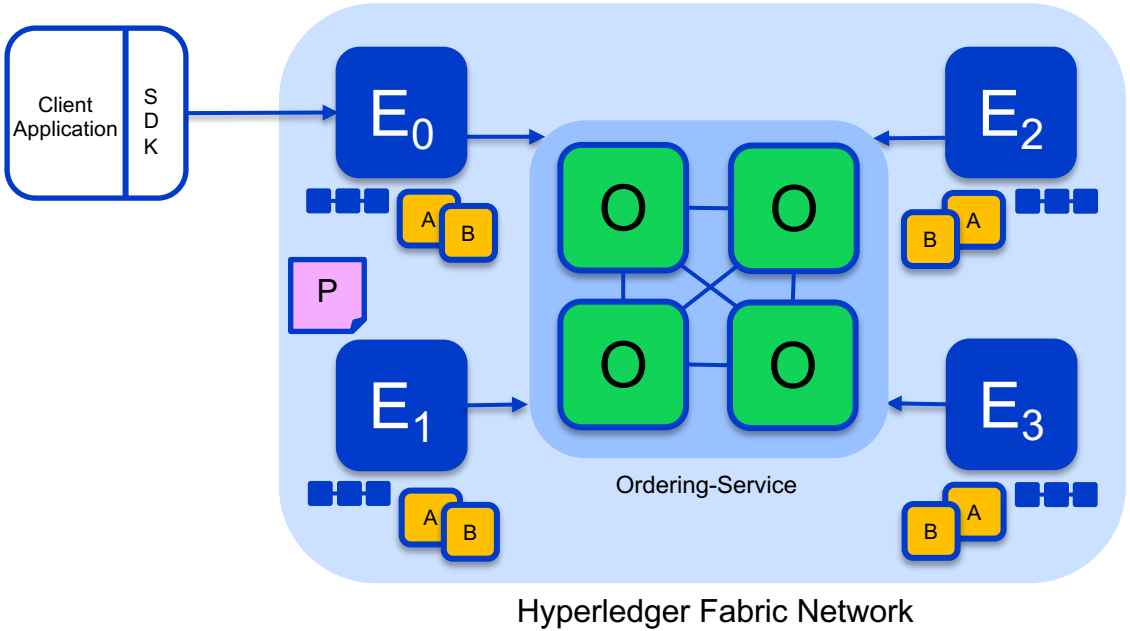


Channels provide privacy between different ledgers

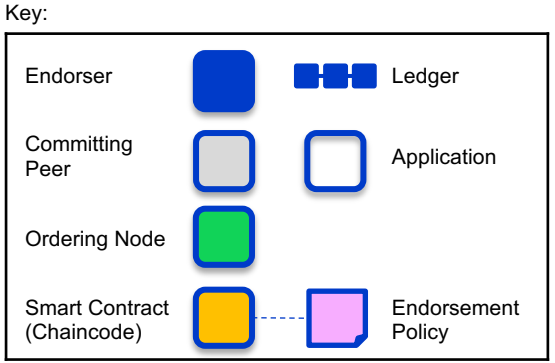


- Ledgers exist in the scope of a channel
  - Channels can be shared across an entire network of peers
  - Channels can be permissioned for a specific set of participants
- Chaincode is installed on peers to access the worldstate
- Chaincode is instantiated on specific channels
- Peers can participate in multiple channels
- Concurrent execution for performance and scalability

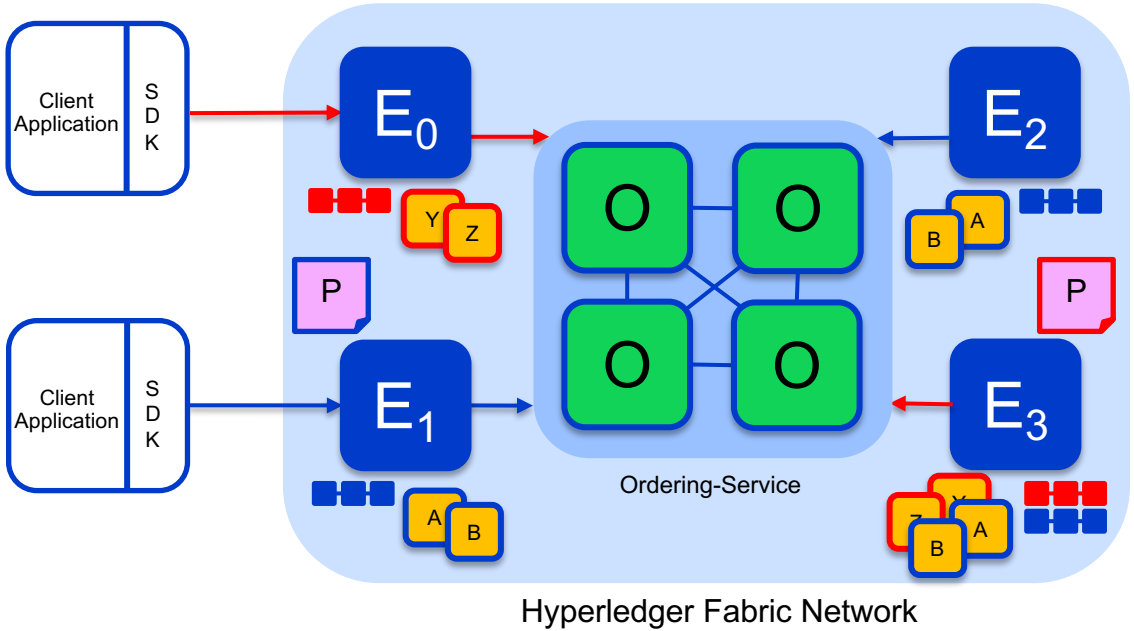
Single Channel Network



- Similar to v0.6 PBFT model
- All peers connect to the same system channel (blue).
- All peers have the same chaincode and maintain the same ledger
- Endorsement by peers E_0, E_1, E_2 and E_3

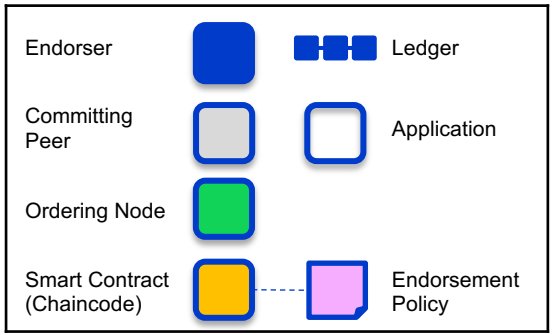


Multi Channel Network



- Peers E_0 and E_3 connect to the red channel for chaincodes Y and Z
- E_1, E_2 and E_3 connect to the blue channel for chaincodes A and B

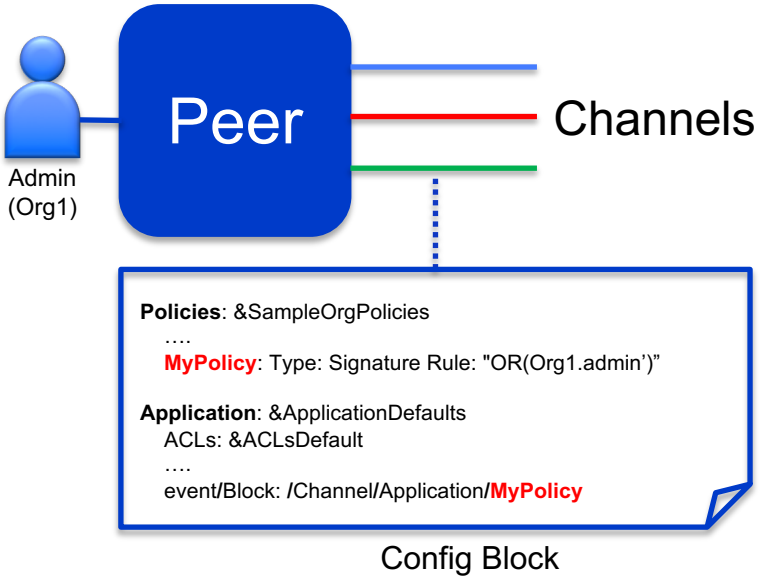
Key:



ACL mechanism per channel

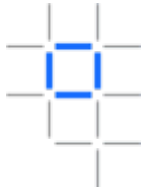
Support policy based access control for peer functions per channel

- Access control defined for channel and peer resources:
  - User / System chaincode
  - Events stream
- Policies specify identities and include defaults for:
  - Readers
  - Writers
  - Admins
- Policies can be either:
  - Signature : Specific user type in org
  - ImplicitMeta : “All/Any/Majority” signature types
- Custom policies can be configured for ACLs



https://jira.hyperledger.org/browse/FAB-8727

Private Data Collections

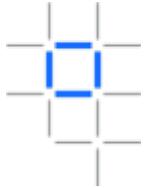


Allows data to be private to only a set of authorized peers

Fabric 1.0 & 1.1	Fabric 1.2
• Data privacy across channels only	• Data privacy within a channel
• Transaction proposal and worldstate read/write sets visible to all peers connected to a channel	• Transaction proposal and worldstate read/write sets available to only permissioned peers
• Ordering service has access to transactions including the read/write sets	• Ordering service has only evidence of transactions (hashes)
	• Complements existing Fabric channel architecture
	• Policy defines which peers have private data

https://jira.hyperledger.org/browse/FAB-8718

Private Data Collections - Explained



1. **Private data:**
    1. Excluded from transactions by being sent as ‘transient data’ to endorsing peers.
    2. Shared peer-to-peer with only peers defined in the collection policy.

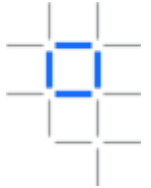
2. **Hashes of private data** included in transaction proposal for evidence and validation.
    1. Peers not in the collection policy and the Orderer only have hashes.

3. **Peers maintain** both a public worldstate and a private worldstate.

4. **Private data** held in a transient store between endorsement and validation.

https://jira.hyperledger.org/browse/FAB-1151

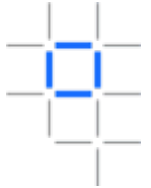
Channels vs Private Data



- Organizations that do not participate in a channel have no knowledge of the transactions in that channel
- Organizations in a channel that are not members of a private data collection know that these “insiders” are transacting, they just do not know the data involved
- Your business use case requirements may favor one approach over the other
- Ordering service receives the transactions in a channel
- Ordering service does not receive private data
  - It receives only the hashes of the private data
  - Private data itself is shared among authorized peers for peer-to-peer gossip

https://jira.hyperledger.org/browse/FAB-1151

Further data privacy enhancements



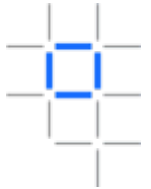
- **State based endorsement – Privacy preserving (FAB-8820)**
  - Planned Fabric v2.0(*)
  - Hides state based endorsement policy

- **Private data – Local collections (FAB-7593)**
  - Planned Fabric v2.0(*)
  - Client decides dissemination policy, not statically defined on chaincode

- **Private data – Org specific collections (FAB-10889)**
  - Planned Fabric v2.0(*)
  - Useful for storing state specific to a single org

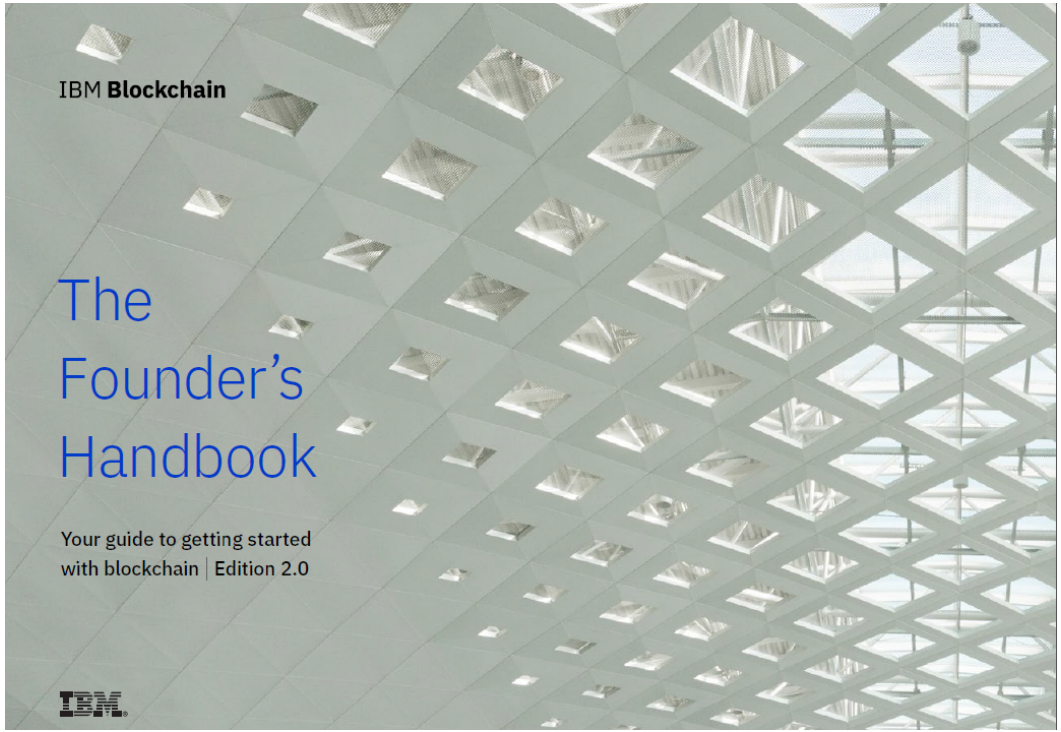
(*) Planned releases are determined by the Hyperledger community and are subject to change

Further Hyperledger Fabric Information



- Project Home: https://www.hyperledger.org/projects/fabric
- GitHub Repo: https://github.com/hyperledger/fabric
- Latest Docs: https://hyperledger-fabric.readthedocs.io/en/latest/
- Community Chat: https://chat.hyperledger.org/channel/fabric
- Project Wiki: https://wiki.hyperledger.org/projects/fabric
- Design Docs: https://wiki.hyperledger.org/community/fabric-design-docs

IBM Blockchain handbook



https://public.dhe.ibm.com/common/ssi/ecm/28/en/28014128usen/the-founders-handbook-edition-2_28014128USEN.pdf

Thank you

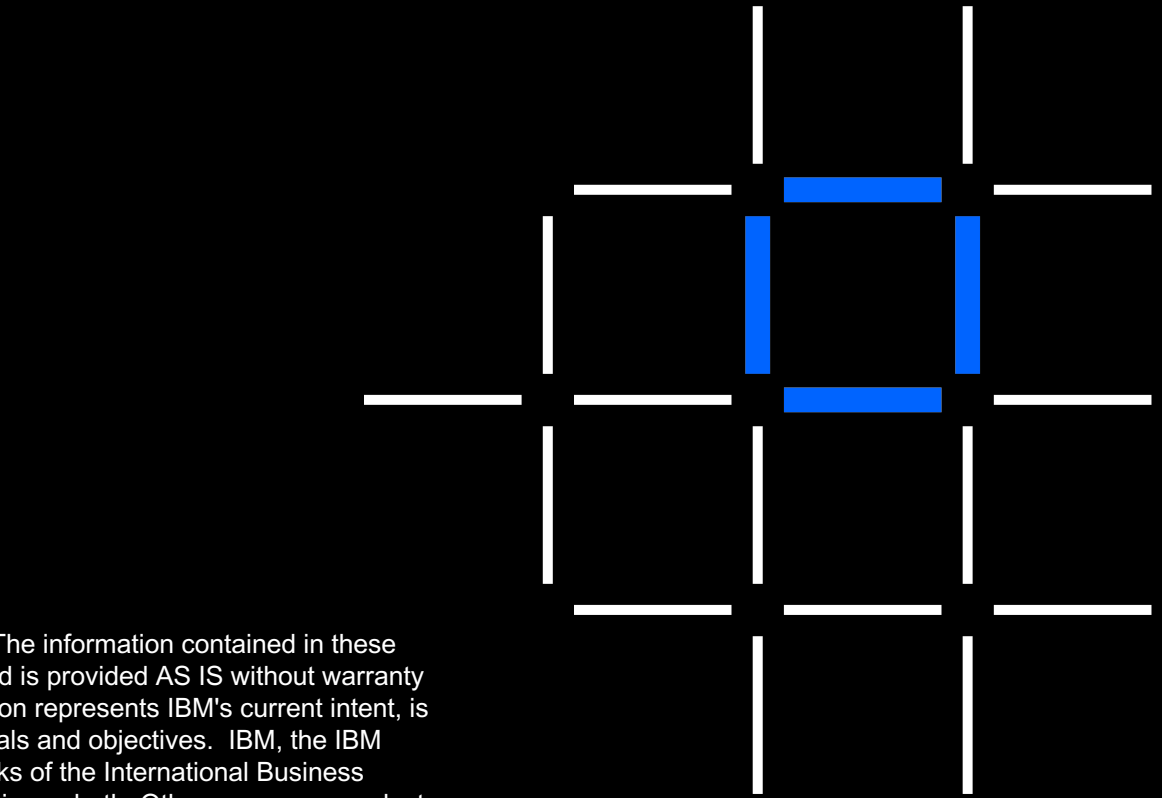
Barry Silliman
silliman@us.ibm.com

IBM **Blockchain**

www.ibm.com/blockchain

developer.ibm.com/blockchain

www.hyperledger.org



IBM


IBM