

Identidad digital en Hyperledger Fabric



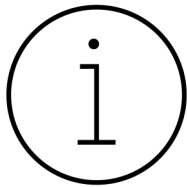
Contenido

- Introducción
- Conceptos básicos de criptografía
- Criptografía para identidad en Fabric
- Membership Service Provider (MSP)
- Control de accesos y permisos
- TLS
- Próximas sesiones



Introducción

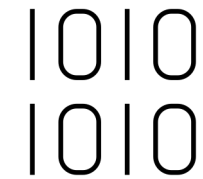
Información que representa un actor físico o virtual en un sistema informático



Nombres de usuario e información relacionada
Certificados digitales
Llaves publicas
...



Personas
Organizaciones
Aplicaciones
Dispositivos
...

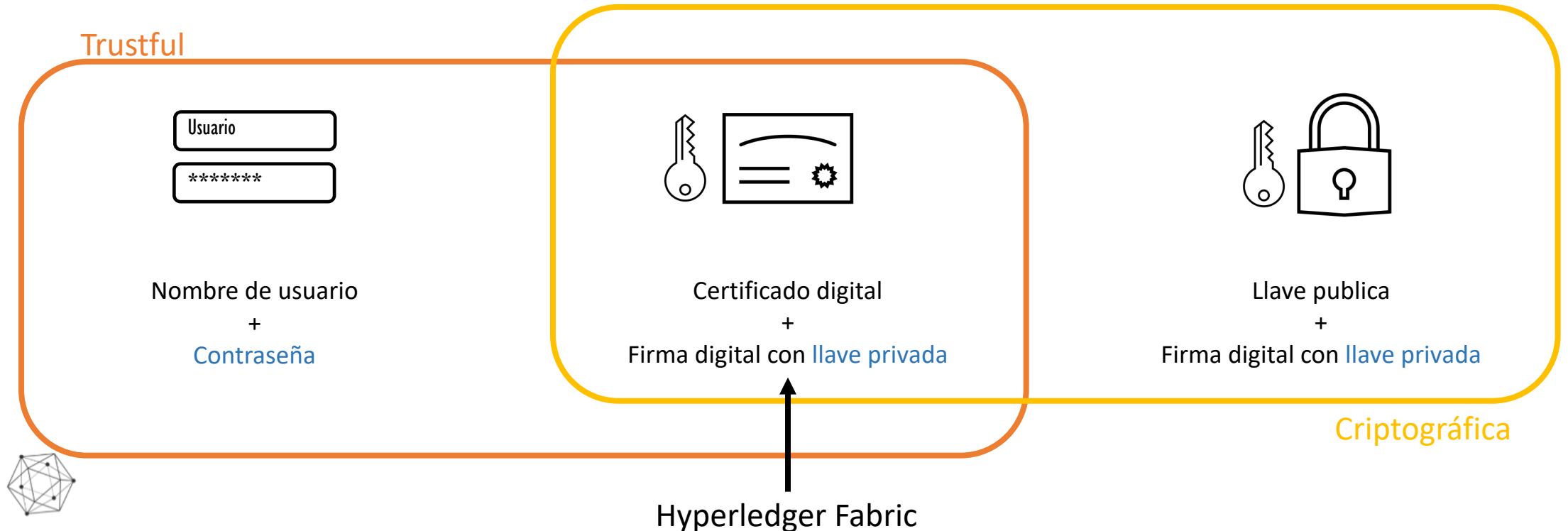


Internet o redes
Aplicaciones
Blockchains
...



Introducción

Identidad digital en combinación con **secretos** permite autenticación y autorización automática



Conceptos básicos de criptografía

Tres tipos principales de algoritmos criptográficos con diferentes usos

Criptografía simétrica
o criptografía de llave secreta



(De)cifrar

Criptografía asimétrica
o criptografía de llave pública

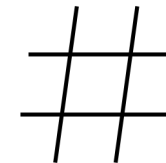


(De)cifrar sin compartir llaves privadas

Firmas digitales

Identidad Digital

Funciones hash




Integridad de información

Datos privados



Conceptos básicos de criptografía

Criptografía de llave secreta (simétrica)

Generar llave 

Cifrar

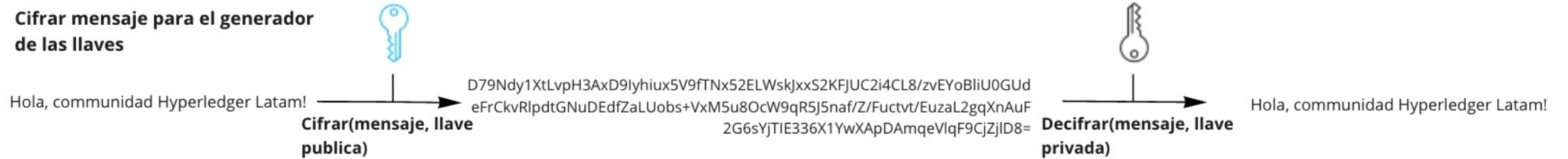


Conceptos básicos de criptografía

Criptografía de llave pública (asimétrica)

Generar par de llaves 

Cifrar mensaje para el generador de las llaves



Generador de par de llaves puede firmar mensaje digitalmente



Conceptos básicos de criptografía

Hash

Hola, comunidad Hyperledger Latam! \longrightarrow 3a7af2f7d2a0a31ac06a446b24233ef8f9591b332d172ea7b146b39217804aeb
Hash(mensaje) miro

- Unidireccional
- Por la misma entrada siempre se genera el mismo hash
- No es factible encontrar dos entradas diferentes con el mismo hash, cada entrada diferente tiene otra salida y un pequeño cambio en una entrada genera un hash completamente diferente sin correlación con el hash de la entrada original
- Entrada de longitud cualquiera, salida de longitud fija (longitud depende del algoritmo de hash usado)
- Función eficientemente computable



Conceptos básicos de criptografía

PKI (Public Key Infrastructure)

Una PKI es un arreglo que vincula llaves públicas con las respectivas identidades de entidades (como personas y organizaciones). La vinculación se establece mediante un proceso de registro y luego una emisión de certificados en y por una autoridad de certificación (CA), el componente principal de una PKI.

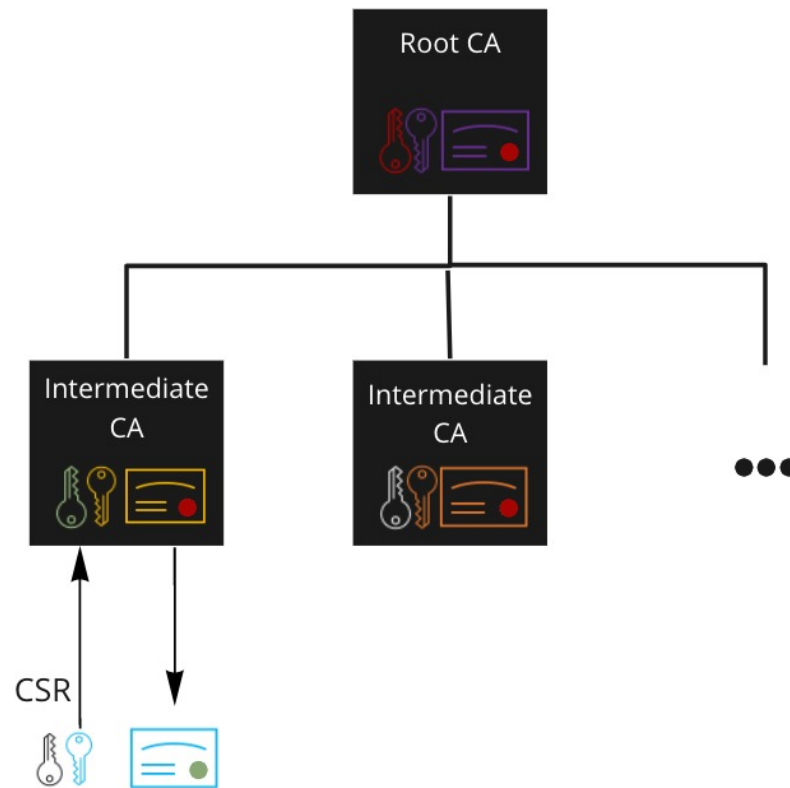
El certificado que contiene la llave pública y información de la identidad, está firmado por la propia llave privada de la CA. Entonces la confianza en la identidad se basa en la confianza en la PKI (o CA)



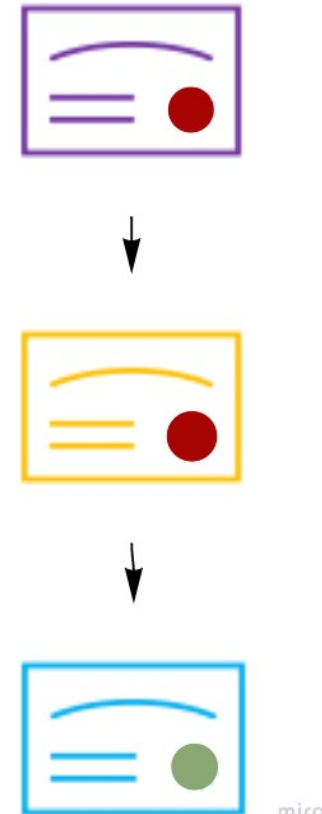
Conceptos básicos de criptografía

PKI (Public Key Infrastructure)

Jerarquía de CA



Cadena de certificados/
Cadena de confianza



Criptografía para identidad en Fabric

Firmas digitales

Cientes
(usuarios o aplicaciones)



Firman transacciones
para autenticarlas

Peers



Firman transacciones
para aprobarlas

Orderers

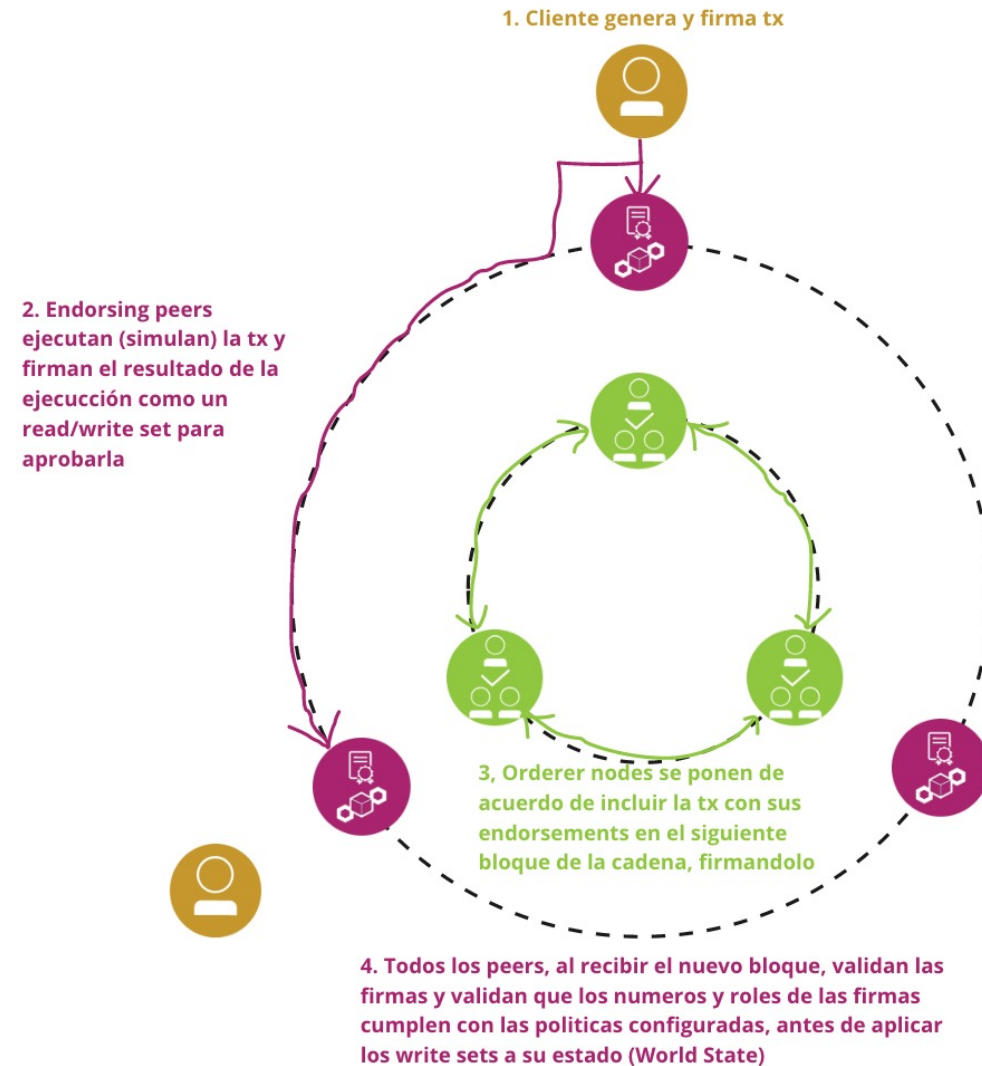


Firman bloques
bajo consentimiento



Criptografía para identidad en Fabric

Firmas digitales en flujo execute-order-commit



Criptografía para identidad en Fabric

Certificados digitales

Hyperledger Fabric es para la creación de permissioned blockchains:

Cada llave publica tiene que ser certificada por una organización participante de la red para que las firmas y con ella las transacciones serán aceptadas



Criptografía para identidad en Fabric

Servicios de PKI (o CA) para la emisión de los certificados digitales

Cryptogen

Herramienta para generar material criptográfica para Hyperledger Fabric con fines de desarrollo y pruebas. La generación es estática, cada ejecución sobrescribe todos los certificados anteriores. **No se debería usar en producción!**

Fabric CA

Fabric CA es una implementación de una Autoridad Certificadora (CA) para Hyperledger Fabric. Tiene la funcionalidad para registrar usuarios o conectar con LDAP. Los usuarios registrados pueden solicitar la emisión de su certificado, lo que directamente devuelve el certificado en la estructura de MSP. Además del servicio de CA mismo, Hyperledger Fabric ofrece también una herramienta de cliente para interactuar con las CA.

Otros servicios de PKI

Servicios administrados de PKI, p.ej. en AWS, CloudHSM para la generación de llaves y ACM-PCA (Amazon Certificate Manager - Private CA) y usar aws cli para automatizar la solicitud y emisión de certificados.



Criptografía para identidad en Fabric

Requisitos mínimos al momento de elegir un servicio de PKI

X.509

- Estándar abierto que define el formato y contenido de certificados de llaves públicas
- Contiene una llave pública y una identidad (una organización, una persona, un nombre de dominio, etc.)
- Firmado por una CA o auto-firmado
- Uso más conocido: establecer comunicación segura (cifrada) con TLS
- También usado para confiar en firmas digitales hechas con la llave privada relacionada a la llave del certificado



Criptografía para identidad en Fabric

Requisitos mínimos al momento de elegir un servicio de PKI

ECDSA (Eliptic Curve Digital Signature Algorithm)

- Dominio dentro de la criptografía asimétrica
- Algoritmo para firmas digitales
- Generación de los pares de llaves basada en las matemáticas de curvas elípticas
- Mas eficiente comparado con algoritmos mas antiguos como RSA:
 - misma nivel de seguridad por llaves mas cortas → reduce el tamaño de las transacciones
 - generación de llaves y de firmas digitales mas rápidas → transacciones mas rápidas
 - verificación de firmas digitales mas rápidas → transacciones mas rápidas



Criptografía para identidad en Fabric



Create CA

Step 1: Select CA type

Step 2: Configure CA subject name

Step 3: Configure CA key algorithm

Step 4: Configure revocation

Step 5: Add tags

Step 6: Configure CA permissions

Step 7: Review

Configure the certificate authority (CA) key algorithm

Choose the key algorithm for your CA. You can change the default selection in the Advanced section.

Advanced

- RSA 2048** The 2048-bit RSA key algorithm is widely supported by browsers and other clients. The 2048-bit size provides a good balance between security and efficiency.
- RSA 4096** RSA 4096 is less efficient than RSA 2048 and typically used only when required for specific applications. For example, some root CAs use RSA 4096.
- ECDSA P256** The ECDSA P256 algorithm is an elliptic curve cryptography (ECC) algorithm. ECC is more efficient than RSA, but not all applications support ECC. ECDSA 256 bit keys are equivalent in cryptographic strength to RSA 3072 bit keys.
- ECDSA P384** The ECDSA P384 algorithm is an elliptic curve cryptography (ECC) algorithm. ECC is more efficient than RSA, but not all applications support ECC. ECDSA 384 bit keys are equivalent in cryptographic strength to RSA 7680 bit keys.

```
aws acm-pca issue-certificate --certificate-authority-arn arn:aws:acm-pca:us-west-2:123456789012:certificate-authority/12345678-1234-1234-1234-123456789012 --csr file://C:\cert_1.csr --signing-algorithm "SHA256WITHECDSA" --validity Value=365,Type="DAYS" --idempotency-token 1234
```



Membership Service Provider (MSP)

Participación (membresía) en Hyperledger Fabric

¿Como podemos certificar llaves (off-chain)?

Public Key Infrastructure (Autoridades certificadoras)

¿Cómo podemos descentralizar ese control y dar la autonomía a cada entidad para certificar sus propios usuarios?

Dar la posibilidad que cada organización puede tener su propia PKI

¿Cómo podemos saber si un certificado pertenece a un miembro de la red y a que organización en especifica?

Compartiendo los certificados de las CA de cada miembro en la blockchain



Membership Service Provider (MSP)

Participación (membresía) en Hyperledger Fabric

Los MSP en los componentes de Fabric usan las cadenas de certificados de las CA para validar operaciones de identidades

Se configura a través de una estructura de carpetas que definen cuales CA son autorizados para emitir identidades validas para sus miembros



Membership Service Provider (MSP)

Se configuran en dos niveles

Channel MSP

- MSP de una organización para ser miembro y tener permisos administrativos en un canal
- Solamente contiene certificados de las CA de la PKI, config.yaml y opcionalmente una lista de revocación
- Guardado en la blockchain del canal y así compartido con todos los otros miembros para validar identidades
- Configurado y usado en configtx.yaml

Local MSP

- MSP de una identidad de cliente o nodo
- Contiene los certificados de las CA de la PKI de la organización, config.yaml, la llave privada y el certificado de la identidad y opcionalmente una lista de revocación
- Define permisos administrativos a nivel del nodo
- Guardado localmente en el nodo o en un HSM (pkcs#11) y solamente accesible por la identidad misma



Membership Service Provider (MSP)

Estructura de carpetas y contenido de msp

Channel MSP

- /cacerts
- /intermediatecerts
- /tlscacerts (opcional, pero recomendado)
- /tlsintermediatecerts (opcional)
- config.yaml (opcional, pero recomendado para especificar roles)
- ~~/admincerts~~ (deprecated, es mas flexible usar config.yaml para definir los certificados de admins)

Local MSP

- /cacerts
- /intermediatecerts
- /tlscacerts (opcional, pero recomendado)
- /tlsintermediatecerts (opcional)
- config.yaml (opcional, pero recomendado para especificar roles)
- ~~/admincerts~~ (deprecated, es mas flexible usar config.yaml para definir los certificados de admins)
- /keystore (private key, nunca compartir!)
- /signcerts



Membership Service Provider (MSP)



Control de accesos y permisos

Roles

- **MSP**
 - Atributos o OU en los **certificados digitales** (signcerts)
 - En **config.yaml** se pueden relacionar atributos de los certificados con roles, p.ej. client, peer, admin o orderer
- En **configtx.yaml** se especifican el MSP y los roles de cada organización y se definen políticas usando esos roles
- **Chaincode**
 - Usar los MSP ID y roles para definir políticas de endorsement
 - Librería **cid**: <https://github.com/hyperledger/fabric-chaincode-go/tree/master/pkg/cid>
 - Se puede obtener la identidad que generó, firmó y envió la transacción para validarla y controlar accesos y permisos en el chaincode



TLS (Transport Layer Security)

Comunicación segura entre y con nodos

- Certificados de TLS permiten validar el servidor y/o cliente y comunicación cifrado entre ambos
- También X.509, no obligatoriamente ECDSA, RSA también funciona para TLS en Fabric
- Recomendado tener CA diferentes a las CA de las identidades para los certificados de TLS
- mTLS necesario entre los orderers con raft
- Recomendado usar mTLS para los otros nodos también, sobretodo si los nodos están expuestos por internet



Próximas sesiones

Agenda

- Usar PKI con jerarquía de CA para cada organización en vez de cryptogen
- Descentralizar el orderer service con raft
- Chaincode
 - Identidad
 - Private Data
 - Eventos, tx hooks, (state-based endorsements...)
- Benchmarking Hyperledger Fabric con Hyperledger Caliper

