Evolución hacia BaaS (Blockchain-as-a-Service) multi-tenant con Aislamiento de Canales y Gestión Soberana

1. Rediseño de la Capa de Identidad y Red

El sistema evoluciona para ser un orquestador dinámico de infraestructura blockchain, priorizando los pilares de la seguridad de la información.

Gestión de Identidades y Acceso (IAM)

- **Usuarios (Humanos):** Se autentican mediante login/password en el dashboard de **React.js** (JWT). Actúan como administradores de sus recursos, pudiendo solicitar sistemas y delegar permisos de lectura.
- **Sistemas (Emisores de Datos):** Cualquier entidad de software (scripts, servidores, apps) con capacidad de realizar peticiones HTTPS y manejar un JSON.
- **Aislamiento por Canal (Confidencialidad):** Al aprobarse un sistema, se genera un **nuevo canal de Fabric** exclusivo. Esto garantiza que los datos de un sistema son invisibles e inaccesibles para cualquier otro actor de la red blockchain, logrando una segmentación total a nivel de ledger.
- **Certificados X.509 (Autenticidad y No Repudio):** Cada sistema posee un certificado único generado por la Fabric CA. Todas las transacciones son firmadas digitalmente por el emisor, impidiendo que este niegue la autoría del dato (No Repudio).

2. Nueva Arquitectura de Datos (LightModel de Alta Integridad)

Se utiliza el **LightModel** para optimizar el rendimiento sin comprometer la seguridad.

Wallet en Base de Datos (Criptografía Segura Local)

Para eliminar la vulnerabilidad de almacenar llaves en el sistema de archivos, implementamos una Wallet Cifrada:

- **AES-256-GCM:** Las llaves privadas se almacenan en MySQL cifradas con este algoritmo, utilizando una `MASTER_KEY` volátil (solo en memoria del servidor).
- **Vector de Inicialización (IV):** Cada llave se cifra con un IV único para evitar ataques de análisis de frecuencia.

Estructura de Base de Datos Evolucionada

Tabla	Propósito de Ciberseguridad
Usuarios	Gestión de credenciales y roles (RBAC).
Sistemas	Identificación del emisor y vinculación a su canal privado de Fabric.
Wallet_Identities	Almacén de secretos criptográficos cifrados (X.509 + Private Keys).
User_Systems_Access	Control de acceso granular: el propietario delega lectura a terceros.
Data (Light)	Almacenamiento del JSON completo y metadatos de auditoría (`tx_id`, `hash`).

3. Flujos de Operación de Seguridad

Provisión de Sistema y Canal

1. **Solicitud:** El usuario solicita la creación de un sistema emisor.
2. **Aprobación:** Un Administrador de Sistemas aprueba la solicitud, disparando un script de automatización que crea el canal en Fabric y emite el certificado X.509.
3. **Propiedad:** El usuario solicitante queda registrado como `Owner`, con control total para revocar accesos o delegar lecturas.

Flujo de Envío y Verificación (Integridad)

1. **Emisión:** El sistema origen envía el JSON y su API Key.
2. **Blindaje:** La API calcula el hash SHA-256, recupera la identidad del sistema de la DB cifrada y firma la transacción en el canal privado.
3. **Auditoría:** En cualquier momento, el propietario o delegados pueden realizar una **verificación cruzada**: el sistema recalcula el hash del JSON en MySQL y lo compara contra el hash inmutable en la blockchain.

4. Nueva Capa de Presentación (React.js)

El frontend deja de ser una visualización estática para convertirse en una **Consola de Gestión de Ciberseguridad**:

- **Gestor de Confianza:** Interfaz para invitar a otros usuarios a ver sistemas específicos (delegación de permisos).
- **Monitor de Integridad:** Indicadores visuales que validan que el JSON local coincide con el registro en el ledger.
- **Gestión de Identidades:** Generación y revocación de API Keys y visualización (pero no extracción) de certificados de seguridad.