UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO

Isadora Delatore Francisco

Blockchain em Sistemas de Saúde: Um estudo de arquiteturas

São Carlos
2025

Isadora Delatore Francisco

BLOCKCHAIN EM SISTEMAS DE SAÚDE: UM ESTUDO DE ARQUITETURAS

Trabalho de Conclusão de Curso submetido à Universidade Federal de São Carlos, como requisito para obtenção do grau de Engenharia de Computação

São Carlos, Julho de 2025

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Isadora Delatore Francisco

Esta Monografia foi julgada adequada para a obtenção do título de Engenharia de Computação, sendo aprovada em sua forma final pela banca examinadora:

Orientador: Prof. Dr. Hélio Crestana Guardia
Universidade Federal de São Carlos - UFSCar

Prof. Dr. Paulo Matias
Universidade Federal de São Carlos - UFSCar

Prof. Dr. Fredy João Valente
Universidade Federal de São Carlos - UFSCar

São Carlos, Julho de 2025

AGRADECIMENTOS

Agradeço meu orientador, Prof. Dr. Hélio Crestana Guardia, pelo apoio em minha jornada acadêmica e pelo exemplo em excelência. Agradeço também a minha família, principalmente meu pai e minha avó, pelo apoio incondicional de minhas escolhas em todos esses anos de graduação e agradeço aos meus amigos da graduação em Engenharia Química que nunca deixaram de me auxiliar em quaisquer dificuldades e que foram especialmente atenciosos no processo de escrita deste trabalho.

Resumo

O aumento da digitalização de dados no setor da saúde, impulsionado por iniciativas de modernização hospitalar, sistemas de Registros Eletrônicos de Saúde (EHR) e dispositivos médicos conectados (IoMT), tem gerado um volume crescente de informações clínicas sensíveis circulando entre diferentes instituições. Esse cenário traz à tona desafios importantes relacionados à segurança da informação, à privacidade dos pacientes e à interoperabilidade entre sistemas heterogêneos. Garantir que esses dados sejam acessados apenas por usuários autorizados, mantidos íntegros durante todo o seu ciclo de vida e compartilhados de forma auditável entre organizações é uma demanda cada vez mais crítica e é nesse contexto que sistemas *blockchain* podem ser explorados como solução. A tecnologia *blockchain* oferece uma estrutura descentralizada e imutável que permite registrar transações de forma segura, transparente e auditável, mesmo em ambientes fragmentados onde as informações tem diferentes origens cada qual com sua base e modelo de dados. Sua aplicação na área da saúde pode viabilizar o compartilhamento controlado de dados entre instituições, garantir a integridade dos registros e fornecer trilhas de auditoria confiáveis, sem depender de uma autoridade central. A pesquisa foi estruturada em duas etapas principais: uma revisão bibliográfica que identificou padrões e tendências em soluções *blockchain* aplicadas à saúde, e a proposta e implementação de arquiteturas baseadas nesses padrões utilizando a ferramenta Hyperledger Fabric. Como contribuição, este trabalho apresenta duas arquiteturas que buscam integrar tecnologias relevantes da literatura: Sistema de Serviço Notário para compartilhamento rastreável de dados entre organizações e Armazenamento de Dados de Dispositivos Médicos Eletrônicos direto em blockchain. A partir dessas arquiteturas, foi esquematizada a configuração de uma rede permissionada com múltiplas organizações, criação de canais, definição de perfis de acesso e início do desenvolvimento da chaincode.

Palavras-Chave: *blockchain*, Sistemas de Saúde, Smart Contracts, *blockchain* SUS

Abstract

The increasing digitalization of data in the healthcare sector, driven by hospital modernization initiatives, Electronic Health Record (EHR) systems, and connected medical devices (IoMT), has led to a growing volume of sensitive clinical information circulating among different institutions. This scenario brings to light significant challenges related to information security, patient privacy, and interoperability between heterogeneous systems. Ensuring that such data is accessed only by authorized users, maintained with integrity throughout its lifecycle, and shared in an auditable manner across organizations is an increasingly critical demand. In this context, blockchain systems can be explored as a potential solution.Blockchain technology offers a decentralized and immutable structure that enables secure, transparent, and auditable transaction recording, even in fragmented environments where data originates from diverse sources, each with its own database and data model. Its application in healthcare can enable controlled data sharing between institutions, ensure the integrity of records, and provide reliable audit trails without relying on a central authority.This research was structured in two main stages: a literature review that identified patterns and trends in blockchain-based solutions for healthcare, and the proposal and implementation of architectures based on those patterns using the Hyperledger Fabric platform. As a contribution, this work presents two architectures that aim to integrate relevant technologies found in the literature: a Notary Service System for traceable data sharing between organizations, and a Direct Storage System for Electronic Medical Device data on the blockchain. Based on these architectures, a permissioned network was planned with multiple organizations, channels were created, access control profiles were defined, and the development of the chaincode was initiated.

Lista de abreviaturas e siglas

BC - *blockchain*

IoT - Internet das Coisas, do inglês, *Internet of Things*

IoMT - Internet das Coisas Médicas, do inglês, *Internet of Medical Things*

IPFS - Sistema de Arquivos Interplanetário, do inglês, *InterPlanetary File System*

MSP - Provedor de Serviços de Autorização, do inglês, *Membership Service Provider*

CA - Autoridade Certificadora, do inglês *Certificate Authority*

SUS - Sistema Único de Saúde

EHR - Registros Eletrônicos de Saúde, do inglês *Eletronical Health Records*

Lista de ilustrações

Figura 1 – Arquitetura de Serviço Notário 28
Figura 2 – Arquitetura de Armazenamento Direto 28
Figura 3 – Exemplo de uma transação 30
Figura 4 – Rede *blockchain* para as Arquiteturas Implementadas 33
Figura 5 – Execução do comando cryptogen 35
Figura 6 – Certificados Gerados para o Orderer 35
Figura 7 – Verificação de Certificados para o Orderer 36
Figura 8 – Criação do Bloco Gênesis 36
Figura 9 – Criação do nó âncora para organização HealthDeviceOrg 36
Figura 10 – contêineres ativos da rede no Docker Desktop 37
Figura 11 – Criação do Canal através do Ordenador 37
Figura 12 – Junção do par ao canal Existente 37

Lista de tabelas

Tabela 1 – Artigos selecionados para implementação da arquitetura generalizada . 23

Sumário

1 INTRODUÇÃO 19

2 REVISÃO BIBLIOGRÁFICA 21
2.1 BC em Sistemas de Saúde 21
2.2 Mapeamento sistemático 22
2.2.1 Método de Busca e Seleção de Arquivos 22
2.2.2 Padrões e Tendências Identificados 24

3 MATERIAIS E MÉTODOS 25
3.1 Cenários e Arquiteturas Propostas 25
3.2 Ferramentas Utilizadas 27
3.2.1 Hyperledger Fabric 29
3.2.2 Docker e Docker Compose 30
3.3 Implementação 30

4 RESULTADOS 35

5 CONCLUSÃO 39

6 APÊNDICE 41
6.1 Arquivos de Configuração da Governança Blockchain 41
6.1.1 crypto-config.yaml 41
6.1.2 configtx.yaml 41
6.2 Docker-compose 45
6.2.1 docker-compose-base.yaml 45

REFERÊNCIAS 49

1 Introdução

Blockchain é uma estrutura de dados distribuída que permite transações seguras, transparentes e à prova de violação, sem a necessidade de uma autoridade central [Zheng et al. 2017]. A arquitetura da *blockchain* consiste em uma rede distribuída ponto a ponto, onde não existe um único ponto de controle ou falha. Os dados são organizados em blocos que armazenam grupos de transações, os quais estão interligados por meio de funções hash criptográficas, assegurando que qualquer alteração em um bloco comprometa a integridade de toda a cadeia [Dinh et al. 2018]. Essa estrutura descentralizada utiliza algoritmos de consenso para validar as transações entre os participantes, eliminando a necessidade de intermediários e promovendo transparência, autenticidade e resistência a fraudes. Medidas de segurança embutidas no sistema garantem não repúdio, rastreabilidade e autenticidade, ao mesmo tempo que isolam os efeitos de comportamentos imprudentes aos seus responsáveis, preservando a resiliência da rede mesmo diante de ataques ou tentativas de interrupção [Tapscott e Tapscott 2016].

No setor da saúde, a tecnologia *blockchain* vem sendo explorada como uma solução para aprimorar a segurança, a interoperabilidade e o controle sobre os dados sensíveis dos pacientes. Entre aplicações relevantes da tecnologia estão a integração de Registros Eletrônicos de Saúde (EHR) entre diferentes instituições, o armazenamento seguro de dados provenientes de dispositivos da Internet das Coisas Médicas (IoMT), e o suporte a auditorias e rastreabilidade em cadeias de suprimentos [Ismail, Materwala e Hennebelle 2021].

No contexto brasileiro, esses benefícios são especialmente relevantes diante dos desafios históricos de fragmentação dos sistemas de informação em saúde pública e privada. A adoção da *blockchain* também se alinha às diretrizes da Lei Geral de Proteção de Dados (LGPD), que estabelece princípios como a minimização de dados, o controle do titular sobre suas informações e a responsabilidade dos agentes de tratamento [Santos, Oliveira e Costa 2024].

Objetivos do Trabalho

O objetivo central desta pesquisa é revisar criticamente as aplicações de *blockchain* no setor de saúde, conforme descritas na literatura científica revisada por pares, e a partir disso, propor arquiteturas simplificadas e funcionais com base em *blockchain* permissionada considerando as principais tendências identificadas na revisão bibliográfica. As arquiteturas desenvolvidas têm como foco dois cenários principais: (i) o compartilhamento seguro e rastreável de registros médicos entre diferentes organizações de saúde, respeitando os

princípios de privacidade e auditoria, e (ii) a persistência distribuída de dados provenientes de dispositivos médicos conectados à Internet das Coisas Médicas (IoMT), garantindo a integridade e disponibilidade desses dados para profissionais de saúde autorizados. Também foi iniciada a implementação destas arquiteturas com Hyperledger Fabric para verificar a viabilidade da arquitetura e refinar aspectos como a identificação dos usuários, governança da rede e possível implementação do chaincode. Com isso, a pesquisa busca não apenas avaliar o estado da arte, mas também fornecer subsídios técnicos e práticos para futuras aplicações reais de *blockchain* em contextos clínicos e institucionais, reforçando os pilares de interoperabilidade, segurança e descentralização.

2 Revisão Bibliográfica

2.1 BC em Sistemas de Saúde

Com o avanço da transformação digital no setor da saúde, busca-se substituir progressivamente os sistemas tradicionais baseados em papel por soluções digitais mais eficientes, seguras e interoperáveis. A digitalização visa superar limitações históricas como a lentidão no acesso a registros, a dificuldade de compartilhamento de informações entre instituições e a alta propensão a erros manuais. Nesse cenário, prontuários eletrônicos de saúde (EHRs), exames laboratoriais, imagens médicas e registros administrativos passaram a ser registrados, processados e compartilhados em ambientes digitais, promovendo maior agilidade no atendimento, melhor continuidade do cuidado e mais precisão nas decisões clínicas [Wisner Audrey Lyndon 2019].

A transformação digital também impulsionou a introdução de dispositivos da Internet das Coisas Médicas (IoMT), como sensores de monitoramento remoto, dispositivos vestíveis e equipamentos hospitalares inteligentes. Esses dispositivos ampliam significativamente a quantidade e a granularidade dos dados gerados sobre os pacientes, permitindo um acompanhamento mais próximo, em tempo real, e com maior suporte à personalização dos tratamentos [Pradyumna et al. 2024].

Porém, conforme dito por [Shahnaz, Qamar e Khalid 2019], a digitalização traz novos desafios, entre eles o risco de acesso não autorizado a informações sensíveis dos pacientes, a possibilidade de adulteração ou perda de integridade dos registros médicos, e a dificuldade de integrar dados que estão dispersos em sistemas heterogêneos e isolados, utilizados por diferentes hospitais, clínicas e operadoras de saúde.

Nesse contexto, soluções baseadas em *blockchain* têm sido cada vez mais exploradas como uma forma de aprimorar a segurança, a interoperabilidade e o controle sobre os dados em saúde. A tecnologia permite o armazenamento descentralizado das informações, possibilitando que diferentes instituições compartilhem dados de forma auditável, íntegra e transparente [Ismail, Materwala e Hennebelle 2021]. Além disso, diferentes mecanismos de controle de acesso embutidos nos sistemas *blockchain* buscam assegurar que apenas partes autorizadas possam visualizar ou registrar informações, fortalecendo a governança e a confiabilidade do fluxo de dados entre os envolvidos [Vaiyapuri Adel Binbusayyis 2021].

2.2 Mapeamento sistemático

Para condução do trabalho a revisão bibliográfica foi realizada com o objetivo de identificar os principais usos de BC para manutenção persistente e segura de dados pessoais de saúde.

2.2.1 Método de Busca e Seleção de Arquivos

Os artigos selecionados para o mapeamento sistemático foram retirados das bases de artigos IEEE Xplore, Google Scholar e o portal da CAPES a partir das palavras chaves: *blockchain*, Decentralized, Architecture, Healthcare. Inicialmente foram definidos os critérios de busca, inclusão e exclusão dos artigos. Foram considerados artigos de pesquisa e revisão já publicados sobre o assunto que abordam o uso de *blockchain* para quaisquer aplicações relacionadas a manutenção segura e persistente de registros médicos. A triagem inicial se deu com base no título e resumo excluindo quaisquer artigos cuja proposta não era relacionada diretamente a *blockchain* e que trouxessem propostas voltadas para arquiteturas. Ao todo foram levantados 45 artigos relevantes e, destes, 14 artigos foram selecionados para leitura completa. Após leitura dos artigos, os dados extraídos foram classificados visando identificar semelhanças entre as implementações. Os artigos selecionados foram:

Nro	**Título**	**Ano**	**Contribuição**
1	[Kleinaki et al. 2018]	2018	Sistema de consulta de dados em banco de dados convencional que registra a hash da consulta e dos resultados em BC
2	[Khatoon 2020]	2020	Diferentes aplicações em saúde utilizando BC utilizando sistema de acesso e banco de dados descentralizado, armazenando hash dos dados no bloco
3	[Pham, Tran e Nakashima 2018]	2018	Sistema de registros de informações de dispositivos *Internet of Things* (IoT) em *blockchain* com fila de prioridade para casos urgentes e perfis de acesso de usuário
4	[Rossetto, Sega e Leithardt 2022]	2022	*blockchain* para armazenamento da encriptação dos dados consultados em um banco de dados descentralizado
5	[Ansar e Ganesh 2022]	2022	*blockchain* para armazenamento da hash de EMR armazenados em cloud e armazenamento de informações de dispositivos IoT

6	[Stephane e Ma 2021]	2021	Arquitetura visando a descentralização do SUS utilizando IPFS e *blockchain* para armazenamento dos dados
7	[Hossein et al. 2021]	2021	Sistema para armazenamento de informações de dispositivos IoT em *blockchain* com controle de acesso e edge computing
8	[Sharma et al. 2020]	2020	Arquitetura de armazenamento de dados de dispositivos IoT com smart contracts de controle de acesso e criptografia
9	[Zhang et al. 2018]	2018	Serviço de gerenciamento de tokens de acesso para diferentes bancos de dados centralizados em *blockchain*
10	[Zhuang et al. 2020]	2020	Arquitetura de múltiplas camadas de *blockchain* para controle de acesso de dados médicos, com uma camada para obtenção de dados, uma camada para smart contracts de validação de usuário e uma camada de interface web para desenvolvimento
11	[Griggs et al. 2018]	2018	Aplicação *blockchain* para dados IoT com smart contracts específicos para cada tipo de informação, voltado para monitoração do paciente
12	[Haritha e Anitha 2023]	2023	Arquitetura *blockchain* de múltiplas camadas para diferentes níveis de segurança de dados em que diferentes usuários possuem acesso
13	[Roehrs, Costa e Righi 2017]	2017	Aplicação que utiliza *blockchain* para armazenamento de registros médicos diversos distribuídos entre diferentes pares com camada de roteamento visando escalabilidade
14	[Chandini e Basarkod 2023]	2023	Sistema para armazenamento de hashs de registros médicos oriundos de bancos de dados descentralizados

Tabela 1 – Artigos selecionados para implementação da arquitetura generalizada

Dentre esses:

- 5 artigos, [3], [5], [7], [8], [11], são voltados para informações de dispositivos IoT,
- Os artigos [1], [5], [9], [10], [12] utilizam armazenamento centralizado, cloud ou não

especificam o tipo de armazenamento

- 4 desdes, [2], [4], [6], [14] utilizam *InterPlanetary File System* (IPFS) ou outro armazenamento descentralizado
- Os artigos [3], [5], [7], [8], [11], [13] não possuem armazenamento externo à *blockchain*
- 10 usam smart contracts nas soluções propostas
- Artigos [10] e [12] possuem múltiplas camadas de *blockchain*
- O artigo [13] faz uso de outras tecnologias descentralizadas, como P2P

2.2.2 Padrões e Tendências Identificados

O armazenamento descentralizado tem se mostrado uma abordagem relevante, com quatro artigos utilizando tecnologias como *InterPlanetary FileSystem* (IPFS) para descentralização dos dados. Esse modelo permite garantir a integridade e disponibilidade das informações sem depender de um provedor centralizado e também permitindo maior escalabilidade [Bigini, Freschi e Lattanzi 2021].

Outro ponto importante é a quantidade de propostas que não utilizam armazenamento externo à *blockchain*. Seis artigos apresentam modelos em que todos os dados são mantidos diretamente nas informações distribuídas. Majoritariamente as soluções envolvendo dados de dispositivos Internet das Coisas Médicas (IoMT) fazem uso da *blockchain* para armazenamento direto e monitoração [Griggs et al. 2018]. Tal arquitetura pode apresentar desafios relacionados ao custo e ao desempenho da rede [Sharma et al. 2020].

O controle de acesso aos dados pode ser realizado baseado em papéis (como o atributo do usuário de saúde, "Hospital", "Médico", "Paciente"), em atributos do usuário, em histórico de acessos anteriores, em controle de uso ou outras formas a depender dos requisitos da rede [Hossein et al. 2021]. Tal controle pode ser realizado via *smart contracts* para evitar processamento externo, o que diminui vulnerabilidades por minimizar a quantidade de sistemas externos interagindo com a rede [Maesa e Mori 2020].

Diante deste contexto, as arquiteturas propostas neste trabalho foram concebidas com bases nos padrões mais recorrentes e viáveis na literatura levantada.

3 Materiais e Métodos

Esta seção descreve os métodos adotados para o desenvolvimento deste trabalho, desde a formulação dos cenários até a descrição da governança das arquiteturas propostas com Hyperledger Fabric. Primeiramente, são apresentados os cenários de uso, seguidos pela modelagem conceitual das arquiteturas e pelos processos de esquematização utilizando a plataforma Hyperledger Fabric.

3.1 Cenários e Arquiteturas Propostas

Para a definição das arquiteturas foram considerados os seguintes cenários:

- Cenário A: O médico da Organização de Saúde A deseja acessar os últimos exames realizados por um paciente na Organização de Saúde B. Ambas as organizações fazem parte da mesma rede *blockchain*. Ele acessa a aplicação com suas credenciais e seleciona quais informações de exames deseja consultar. Isso gera uma transação na blockchain de solicitação de consulta que vai ser liberada ou não a depender do perfil do usuário (Médico). A aplicação recupera os dados diretamente do banco de dados da Organização B mediante status de permissão na primeira transação (bloco) de solicitação, criptografa e armazena também o registro da consulta executada pela Organização B. Como o usuário possui permissão adequada, as informações são exibidas para ele. Neste cenário diferentes interessados com suas bases de dados próprias e pacientes buscam compartilhar informações médicas com rastreabilidade e não-repúdio.

- Cenário B: Médica da Organização de Saúde B possui um paciente sob monitoração. O paciente possui um oxímetro de pulso, que mede a saturação do oxigênio no sangue e a frequência cardíaca a cada intervalo de tempo e envia para a *blockchain* como uma transação. Ela gostaria de saber se houve queda de saturação nas últimas três horas. Para isso, ela acessa o sistema de monitoração da organização com suas credenciais e acessa o monitor do dispositivo correto no paciente desejado. O sistema recupera as informações dos blocos no intervalo de tempo desejado e as exibe em tela, assim como um registro das últimas consultas realizadas para aquele paciente e dispositivo. A transação de consulta é registrada em *blockchain*. É necessário considerar que os dispositivos IoMT consigam processar e enviar dados em forma de transações para a BC.

Os cenários implicam a existência de uma interface gráfica simplificada para uso de

diferentes usuários com diferentes níveis de conhecimento. Essa interface pode ser unificada entre os interessados ou pode ser um conjunto de APIs para transações com a *blockchain*. Em ambos os cenários, supõe-se um sistema para a manutenção destas interfaces e para a ordenação dos blocos na rede, conforme será descrito melhor nas arquiteturas. Para atender aos cenários, as seguintes arquiteturas foram propostas:

- **Arquitetura de Serviço Notário (Figura 1):** Os dados permanecem armazenados em uma base externa (centralizada ou descentralizada), enquanto a *blockchain* registra apenas as solicitações e respostas de consultas. Esse modelo é útil para aplicações que exigem auditabilidade, interoperabilidade e confidencialidade, sem comprometer a privacidade de dados sensíveis. Neste cenário, o objetivo é o compartilhamento de informações de forma rastreável entre diferentes organizações da área da saúde. O sistema solicitante (cliente) inicia uma requisição para acessar dados mantidos por outro sistema de saúde, fornecendo suas credenciais e o tipo de informação desejada. Um *smart contract* recebe a solicitação e verifica se o usuário possui um perfil de acesso autorizado, com base em seu papel (*role*) e identidade digital validada via MSP. Caso a verificação falhe, o acesso é negado, e o evento é registrado em *blockchain*, garantindo transparência e não repúdio. Se autorizado, o *smart contract* emite um evento de leitura que é escutado por um componente *backend* externo à blockchain (*listener*), responsável por realizar a integração com o banco de dados da organização detentora da informação. Esse componente *backend* executa a consulta, encripta os dados com a chave pública do solicitante, e grava uma nova transação na *blockchain* contendo os metadados da consulta (usuário, *timestamp*, tipo de dado requisitado), o status da solicitação e os dados encriptados. Por fim, o cliente recebe os dados, realiza a desencriptação local com sua chave privada e os exibe ao usuário autorizado. A *blockchain*, nesse modelo, atua como um mecanismo confiável de autenticação, autorização, auditoria e rastreabilidade, permitindo o compartilhamento seletivo de dados entre organizações, sem a necessidade de replicação total da base de dados em todos os nós.

- Arquitetura de Armazenamento Direto (Figura 2): Os dados são armazenados diretamente na *blockchain*, garantindo integridade e rastreabilidade. Essa abordagem é adequada para o armazenamento seguro de dados majoritariamente utilizados para monitoração e que não precisam ser armazenados em um banco de dados, mas sobre os quais é possível realizar análises, alertas e monitorações. Assim, para essa proposta deve-se considerar que tal dispositivo é capaz de coletar e enviar os dados encriptados para a *blockchain*. O sistema de saúde interessado deve solicitar acesso ao *smart contracts* que faz a verificação de suas credenciais e garante que se trata de um usuário autorizado ao acesso ao dado. Toda tentativa de consulta aos

dados, autorizada ou não, também é inserida em um bloco da rede para garantir rastreabilidade.

Em ambos os casos os dados são encriptados para garantir a segurança da informação e foram definidos perfis de acesso validados por *smart contracts*, de forma que apenas usuários com permissão podem ver a informação.

A encriptação dos dados na própria blockchain pode representar uma falha crítica de segurança e também vai contra a característica determinística do *chaincode*. As chaves não podem ser expostas para o ambiente da blockchain em que diversas organizações tem acesso e determinados tipos de criptografia (com chaves assimétricas ou simétricas com chaves aleatórias) podem gerar resultados diferentes para cada execução. Portanto, a encriptação dos dados é realizada por meio de criptografia simétrica externamente à blockchain, pelas organizações. Quando uma organização de saúde recebe uma solicitação de dados válida, os dados consultados em seu banco são encriptados utilizando um algoritmo simétrico, com uma chave previamente compartilhada com a organização solicitante. Após a encriptação, os dados são registrados em uma nova transação na *blockchain*, permitindo que apenas a parte que detém a mesma chave consiga desencriptá-los corretamente. A segurança deste modelo depende da confidencialidade da chave, que deve ser distribuída de forma segura entre as partes envolvidas antes do compartilhamento de dados. Essa chave pode ser armazenada localmente em um ambiente seguro ou gerenciada por sistemas especializados como cofres de segredo (*secret vaults*).

O requisito de determinismo do Fabric também impede que o *chaincode* acesse dados fora da BC. Por isso no modelo de Sistema Notário é necessário que haja dois blocos para uma consulta: um de solicitação e um com o retorno criptografado da solicitação.

Após a generalização dos padrões arquiteturais identificados na literatura, foi realizada a implementação utilizando Hyperledger Fabric. O objetivo dessa etapa foi validar a viabilidade prática das soluções propostas em um ambiente controlado.

3.2 Ferramentas Utilizadas

As seguintes tecnologias foram utilizadas para a implementação das soluções propostas:

- Hyperledger Fabric 2: Hyperledger Fabric é uma *blockchain* permissionada, desenvolvida sob a iniciativa Hyperledger da Linux Foundation.

- Docker e Docker-Compose: Docker é uma plataforma que permite criar, testar e implantar aplicações em ambientes isolados chamados *contêineres*, garantindo que o código funcione de forma consistente em diferentes máquinas.

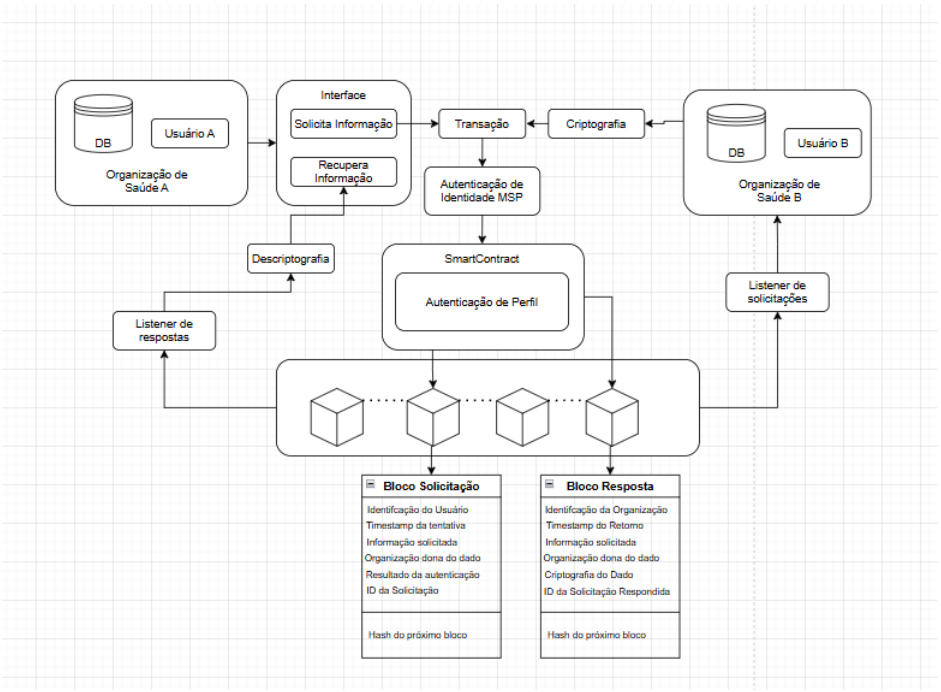

Figura 1 – Arquitetura de Serviço Notário

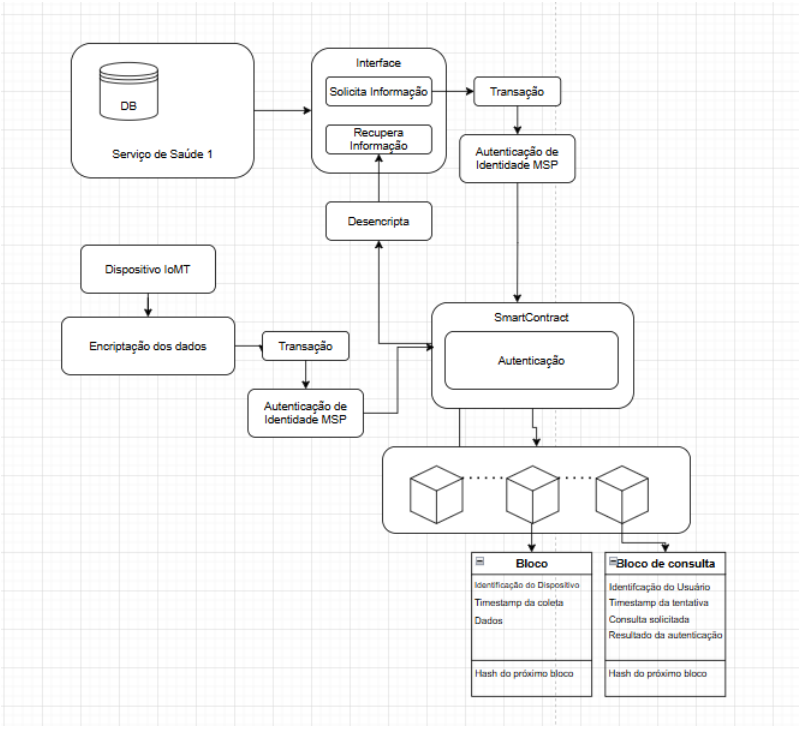

Figura 2 – Arquitetura de Armazenamento Direto

3.2.1 Hyperledger Fabric

Conforme descrito na documentação oficial [Hyperledger Fabric Contributors 2023], Hyperledger Fabric é uma plataforma de *blockchain* comissionada projetada para uso corporativo, em que apenas usuários com permissão podem visualizar e interagir com a rede.

A rede é composta por *peers*, que são os nós responsáveis pelo processamento e armazenamento das transações. Para que a comunicação entre os *peers* de diferentes organizações aconteça, cada organização pode definir um ou mais pares âncora (*anchor peers*). Os pares âncora atuam como pontos de contato entre organizações distintas, permitindo a troca de informações e a propagação de transações na rede.

Além dos *peers*, a rede do Hyperledger Fabric conta com os *orderers*, que são responsáveis pela ordenação e distribuição dos blocos de transações para os nós. Do ponto de vista das organizações participantes, o *ordering service* é essencial para garantir a consistência da *blockchain*, pois define a sequência em que as transações serão validadas e aplicadas. Cada organização depende do *ordering service* para receber blocos confirmados, garantindo que todas as partes mantenham um estado consistente e confiável da rede.

Diferentes organizações podem participar de diferentes *blockchain*s a partir dos canais. Um canal é um ambiente privado dentro da rede *blockchain* que permite que um grupo específico de organizações compartilhe transações e dados sem que outras participantes tenham acesso. Cada canal tem seu próprio conjunto de nós, nós âncora e um serviço de ordenação, funcionando como uma *blockchain* isolada dentro da infraestrutura geral.

As identidades dos participantes são autorizadas a partir do *Membership Service Provider* (MSP), que fornece certificados digitais para autenticar usuários e organizações. Esses certificados são emitidos por uma Autoridade Certificadora (CA) e são usados para assinar transações. Cada *peer* de cada organização, assim como os *orderers*, possui seu próprio conjunto de certificados.

As transações no Hyperledger Fabric são divididas em três etapas: proposta, ordenação e validação. Inicialmente um cliente autenticado envia uma proposta de transação para um conjunto de pares que entram na política de endosso do *smart contract*. Tais pares simulam a execução do *chaincode* sem alterar a *blockchain* e retornam ao cliente os dados lidos, os dados escritos e suas assinaturas digitais. O cliente deve coletar assinaturas o suficiente de acordo com as políticas definidas para a rede antes de agrupá-las e enviar para o par ordenador. É responsabilidade do ordenador receber todas as transações dos clientes, ordená-las e agrupá-las em um bloco. O *orderer* não verifica a se as transações inseridas são válidas, essa validação ocorre somente após o *broadcast* da *blockchain* para os pares e é realizada por estes. Os pares somente efetivam as transações dos registros distribuídos após validar os endossos e conflitos de estados. Todo processo é sustentado

por um sistema de identidades robusto baseado no MSP e assim, apenas participantes com credenciais válidas podem interagir com a rede. A Figura 3 apresenta esse processo.

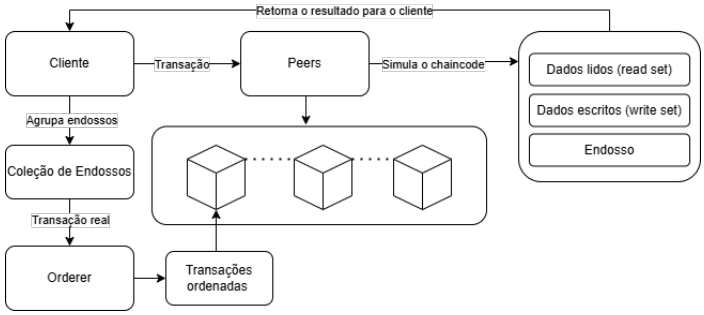

Figura 3 – Exemplo de uma transação

3.2.2 Docker e Docker Compose

Docker e Docker-Compose foram utilizados para configurar e rodar múltiplos nós da rede *blockchain* (*peers* e *orderer*) sem a necessidade de configurar cada máquina manualmente.

3.3 Implementação

Para criar uma rede no Hyperledger Fabric foi necessário, primeiro, definir a governança da rede a ser criada. Nesta etapa, define-se através de arquivos de configuração YAML quantas organizações participam, seus *peers*, *anchor peers*, *orderers* e outras informações relacionadas. Também são definidas uma série de políticas de acesso, consenso e especificações dos canais. O cenário foi modelado com três organizações distintas:

- **Entidade Governamental**: responsável pelo par de ordenação (*orderer*) e pela transmissão dos blocos da *blockchain*. Também é uma entidade capaz de inserir e consultar registros na rede.
- **Serviço de Saúde A**: organização representa um prestador não-governamental de serviços de saúde que consulta e insere registros médicos na rede.
- **Dispositivos de Saúde A**: dispositivos IoT simulados, responsáveis apenas pela inserção de dados médicos, sem realizar consultas.

Esse modelo foi inspirado no contexto brasileiro, onde o Sistema Único de Saúde (SUS) atua como uma entidade responsável pela integração entre dados de instituições públicas e privadas. Assim, a rede simula uma infraestrutura de *blockchain* mantida pelo governo para o compartilhamento auditável de informações médicas entre organizações do setor.

A geração dos certificados de autenticação da rede foi realizada por meio de um arquivo de definição da rede (`crypto-config.yaml`), no qual foram especificados o nome

e domínio de cada organização, a quantidade de *peers* por entidade e os parâmetros do ordendor, atribuído à Entidade Governamental. Os certificados digitais e identidades MSPs das organizações foram gerados automaticamente a partir dessas configurações, com o `cryptogen`, ferramenta nativa do *Fabric*. É necessário destacar que em ambientes produtivos os certificados não são gerados utilizando `cryptogen` pois são definidos externamente pelas CAs das organizações participantes.

```
cryptogen generate --config=./crypto-config.yaml
```

Na etapa seguinte, foi criado o arquivo de configuração da rede (`configtx.yaml`), que define políticas para a rede, seus pares, ordenador e canal. Ele é dividido em diferentes seções

- *Organizations*: Esta seção lista os membros participantes da rede e define suas políticas de leitura, escrita, aprovação e administração.
- *Orderer*: Define configurações para o par ordenador como leitura, escrita, aprovação, administração, tamanho de bloco em bytes e número de mensagens por bloco
- *Aplication*: Define parâmetros para a aplicação
- *Channel*: Decreve as políticas relacionadas ao canal
- *Capabilities*: Seção utilizada para gestão de versões de arquivos binários
- *Profiles*: Permite o setup de diferentes configurações em apenas um único arquivo

Por se tratar de um ambiente não produtivo as políticas de leitura e escrita foram definidas para qualquer membro participante das organizações enquanto a aprovação e administração podem ser realizadas por qualquer par administrador de qualquer organização.

A ferramenta `configtxgen` utiliza este arquivo de configuração para a geração dos blocos gênesis, artefatos de canal e para criar os pares âncora de cada organização participante do canal. O bloco gênesis é o primeiro bloco de um canal, essencial para inicializar o serviço de ordenação (*orderer*) e estabelecer os parâmetros da rede. `GovOrdererGenesis` é o nome do perfil de configuração definido, `healthchannel` é o identificador lógico do canal de ordenação, e `healthblock.pb` é o artefato resultante, o primeiro bloco necessário para criação dos canais. Uma vez que apenas os pares âncora de cada organização podem interagir uns com os outros, `configtxgen` foi utilizado para a geração dos *anchor peers* dentro do canal.

Os comandos utilizados foram:

```
configtxgen -profile GovOrdererGenesis -outputBlock ./config/healthblock.pb \
-channelID healthchannel
```

```
configtxgen -profile HSChannel \
  -outputAnchorPeersUpdate ./config/HealthServiceOrgMSPanchors.tx \
  -channelID healthchannel -asOrg HealthServiceOrg
configtxgen -profile HSChannel \
  -outputAnchorPeersUpdate ./config/HealthDispositiveOrgMSPanchors.tx \
  -channelID healthchannel -asOrg HealthDeviceOrg
```

As configurações para o *orderer* são realizadas em um único arquivo de configuração chamado `orderer.yaml` dividido nas seções:

- *General*: propriedades gerais do ordenador
- *FileLedger*: diretório onde o ordenador manterá o registro distribuído (*ledger*)
- *Consensus*: define o tipo de consenso entre kafta, etcdraft e solo
- *Kafka*: define o setup do ambiente caso o ordenador seja definido como kafka
- *Debug*: controla informações relacionadas ao debug, como os diretórios para registros de transações broadcast ou deliver.
- *Operations*: endpoint para monitoração e alertas da rede
- *Metrics*: configurações relacionadas às métricas emitidas pelo ordenador

É este arquivo que o ordenador utiliza em sua inicialização. Os pares têm um arquivo semelhante chamado `core.yaml` executado em sua inicialização que não será descrito neste trabalho pois foram utilizadas as configurações padrões obtidas da imagem docker do HyperLedger Fabric.

A próxima etapa na implementação foi a virtualização do processo. Para simular ambientes distintos interagindo com a *blockchain* tais pares das organizações foram simulados em docker *contêineres*. Para cada par integrante da rede foi gerado um contêiner diferente. Foi utilizado um arquivo `docker-compose-base.yaml` com imagens fornecidas pelo hyperledgerFabric contendo os certificados TLS e MSP gerados nas etapas anteriores, os artefatos do canal e para o par ordenador também foi incluído o bloco gênesis. Para ambas as arquiteturas a rede resultante está representada na Figura 4.

O arquivo `docker-compose-base.yaml` foi criado para definir os contêineres dos principais componentes da rede *blockchain*, com ênfase nos pares, ordenadores e seus respectivos volumes, variáveis de ambiente, portas e mapeamentos de diretórios locais. Esse arquivo atua como uma especificação da infraestrutura dos nós da rede, permitindo que os serviços sejam inicializados com as configurações necessárias para a operação da *blockchain*. Os certificados de autenticação e segurança para cada membro foram replicados para seus respectivos contêineres assim como os blocos gerados nos passos anteriores.

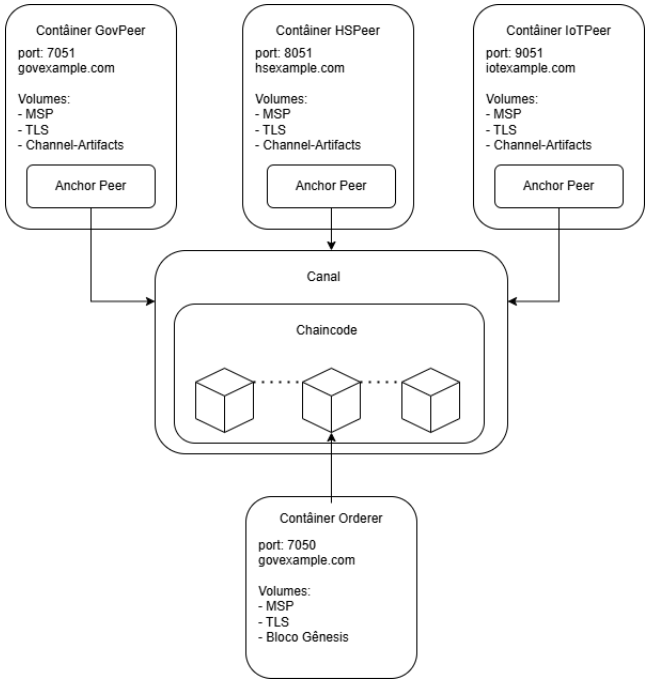

Figura 4 – Rede *blockchain* para as Arquiteturas Implementadas

Após a rede estar configurada foi necessário fazer a criação do canal e juntar os pares ao canal. O comando `osnadmin` permite a criação do canal utilizando o bloco gênesis definido anteriormente. Ele se comunica com o contêiner do *orderer*, que faz as validações de autenticidade dos certificados e cria a estrutura de diretórios localmente para reconhecer e aceitar blocos desse canal.

```
osnadmin channel join  -o orderer.govexample.com:7053   --ca-file $GOV_TLS_CA \
--client-cert $GOV_CLIENT_CERT   --client-key  $GOV_CLIENT_KEY  \
--channelID healthchannel --config-block ./config/healthblock.pb
```

Após a criação do canal, foi executado o comando `peer channel join` em cada contêiner de par das organizações definidas para juntá-los ao canal criado.

A *chaincode* no hyperledger é a ferramenta de implementação de um ou mais *smart contracts* no canal. O processo para que uma *chaincode* seja executado na rede BC é o desenvolvimento, empacotamento, instalação (em que todos os pares devem instalar a *chaincode*) e aprovação (pelo menos um dos administradores das organizações participantes deve aprovar a *chaincode*). Para as arquiteturas propostas a *chaincode* foi utilizada para validação do perfil, encriptação dos dados e registro da transação, conforme o pseudocódigo apresentado a seguir:

```
função SolicitarConsulta(solicitanteID, tipoDeDadoDesejado):

    // 1. Obter identidade e perfil do solicitante
    perfilSolicitante = verificarPerfil(solicitanteID)

    // 2. Verificar se o perfil tem permissão para esse tipo de dado
    se NOT autorizado(perfilSolicitante, tipoDeDadoDesejado):
        registrarNoLedger({
            solicitante: solicitanteID,
```

```
            tipoDeDado: tipoDeDadoDesejado,
            status: "negado",
            motivo: "perfil sem permissão",
            timestamp: agora()
        })
        retornar "Acesso negado"

    // 3. Registrar no ledger a solicitação autorizada (sem query nem dados)
    registrarNoLedger({
        solicitante: solicitanteID,
        tipoDeDado: tipoDeDadoDesejado,
        status: "permitido",
        timestamp: agora()
    })

    // 4. Emitir evento para componente externo que irá executar a consulta e processar dados

    retornar "Solicitação autorizada, aguardando processamento externo"
```

Para o desenvolvimento da código é sugerida a linguagem de programação Go (Golang), mais indicada para esse tipo de aplicação no Hyperledger pois o próprio Hyperledger Fabric é desenvolvido em Go, de forma que a *chaincode* roda como processo nativo e a integração com o resto do sistema é mais eficiente. O desenvolvimento e a aplicação do *chaincode* não foram realizados nesse trabalho.

No Hyperledger Fabric, a gestão de identidades e perfis de acesso é realizada por meio do *Membership Service Provider* (MSP), que define e valida as credenciais dos participantes da rede. A autenticação baseia-se em certificados digitais X.509, emitidos por uma autoridade certificadora confiável. Ao utilizar a ferramenta *cryptogen* para geração dos certificados, as informações de perfil de acesso podem ser inseridas, por exemplo, no campo *Common Name* (CN) do certificado. Cada certificado conterá o CN que representa a identidade e o papel (role) do usuário ou peer dentro da organização. Dessa forma, o *chaincode* pode realizar verificações de autorização baseadas no CN extraído da identidade do solicitante. Contudo, essa abordagem é limitada, pois a personalização dos perfis depende do conteúdo estático do certificado, dificultando alterações dinâmicas sem regeneração dos certificados.

A próxima etapa consiste na validação dessas arquiteturas por meio da execução parcial de seus componentes, cujos resultados e limitações são discutidos na seção 4.

4 Resultados

Nesta seção, são apresentados os resultados obtidos com a implementação das arquiteturas propostas utilizando as ferramentas apresentadas. O objetivo foi verificar a viabilidade técnica das arquiteturas sugeridas na revisão bibliográfica por meio da criação de ambientes funcionais de *blockchain* permissionada voltados ao contexto da saúde. A primeira etapa consistiu na geração dos certificados de autenticação, nos artefatos de canal e nos pares âncora.

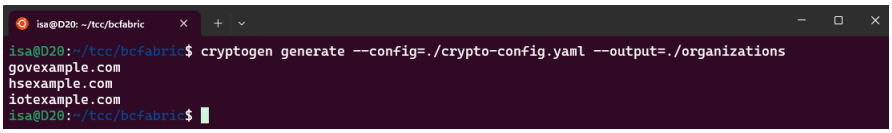

Figura 5 – Execução do comando cryptogen

Conforme é possível observar na figura 6, o comando utilizado gerou os diretórios necessários e seus certificados para todos os pares da rede, incluindo usuários Admin para cada organização.

Figura 6 – Certificados Gerados para o Orderer

O *script* `verifica-certificados.sh` foi desenvolvido para validar os certificados utilizados na rede. Ele verifica o período de validade dos certificados, se estes foram assinados por uma CA confiável e se há correspondência entre as chaves privadas e seus

respectivos certificados públicos. Conforme figura 7 é possível ver que os certificados foram gerados corretamente.

```
isa@D20:~/tcc/bcfabric$ ./verifica-certificados.sh ./organizations/ordererOrganizations/govexample.com
Verificando certificados da organização: ./organizations/ordererOrganizations/govexample.com

Verificando MSP principal:
Certificado: ./organizations/ordererOrganizations/govexample.com/msp/cacerts/ca.govexample.com-cert.pem
notBefore=Jun 23 16:14:00 2025 GMT
notAfter=Jun 21 16:14:00 2035 GMT

Certificado: ./organizations/ordererOrganizations/govexample.com/msp/tlscacerts/tlsca.govexample.com-cert.pem
notBefore=Jun 23 16:14:00 2025 GMT
notAfter=Jun 21 16:14:00 2035 GMT

Verificando Admin:
Certificado: ./organizations/ordererOrganizations/govexample.com/users/Admin@govexample.com/msp/signcerts/Admin@govexamp
le.com-cert.pem
notBefore=Jun 23 16:14:00 2025 GMT
notAfter=Jun 21 16:14:00 2035 GMT

Comparando chave e certificado:
  Hash da chave: (stdin)= 156c892756338f7b6a308288ca7af7ac7bc26dd57f0a43d0c69044a0ea264da8
  Hash do cert.: (stdin)= 156c892756338f7b6a308288ca7af7ac7bc26dd57f0a43d0c69044a0ea264da8
Chave e certificado combinam

Verificando se ./organizations/ordererOrganizations/govexample.com/users/Admin@govexample.com/msp/signcerts/Admin@govexa
mple.com-cert.pem é assinado por CA:
./organizations/ordererOrganizations/govexample.com/users/Admin@govexample.com/msp/signcerts/Admin@govexample.com-cert.p
em: OK
```

Figura 7 – Verificação de Certificados para o Orderer

As figuras 8 e 9 mostram a criação com sucesso dos artefatos de canais, bloco gênesis e *anchor peers* para todos os pares.

```
isa@D20:~/tcc/bcfabric$ configtxgen -outputBlock ./config/gov-genesis.block -profile GovOrdererGenesis -channelID ordere
rchannel
2025-06-23 13:58:46.339 -03 0001 INFO [common.tools.configtxgen] main -> Loading configuration
2025-06-23 13:58:46.344 -03 0002 INFO [common.tools.configtxgen.localconfig] completeInitialization -> orderer type: sol
o
2025-06-23 13:58:46.344 -03 0003 INFO [common.tools.configtxgen.localconfig] Load -> Loaded configuration: /home/isa/tcc
/bcfabric/configtx.yaml
2025-06-23 13:58:46.345 -03 0004 INFO [common.tools.configtxgen] doOutputBlock -> Generating genesis block
2025-06-23 13:58:46.346 -03 0005 INFO [common.tools.configtxgen] doOutputBlock -> Creating system channel genesis block
2025-06-23 13:58:46.346 -03 0006 INFO [common.tools.configtxgen] doOutputBlock -> Writing genesis block
```

Figura 8 – Criação do Bloco Gênesis

```
isa@D20:~/tcc/bcfabric$ configtxgen -profile HSChannel -outputAnchorPeersUpdate ./channel-artifacts/HealthDispositiveOrg
anchors.tx -channelID healthchannel -asOrg HealthDispositiveOrg
2025-06-23 14:11:37.552 -03 0001 INFO [common.tools.configtxgen] main -> Loading configuration
2025-06-23 14:11:37.556 -03 0002 INFO [common.tools.configtxgen.localconfig] Load -> Loaded configuration: /home/isa/tcc
/bcfabric/configtx.yaml
2025-06-23 14:11:37.556 -03 0003 INFO [common.tools.configtxgen] doOutputAnchorPeersUpdate -> Generating anchor peer upd
ate
2025-06-23 14:11:37.558 -03 0004 INFO [common.tools.configtxgen] doOutputAnchorPeersUpdate -> Writing anchor peer update
isa@D20:~/tcc/bcfabric$
```

Figura 9 – Criação do nó âncora para organização HealthDeviceOrg

Para a inicialização dos pares e do nó ordenador da rede, foi utilizado o Docker Compose com o arquivo `docker-compose-base.yaml`. Além dessa definição, cada serviço depende de um arquivo de configuração específico para controlar seu comportamento interno no momento de inicialização. O arquivo `orderer.yaml` foi utilizado pelo serviço do ordenamento, especificando parâmetros como o endereço de escuta, diretórios de armazenamento, método de bootstrap, certificados TLS e a ativação da API de administração. Já o arquivo `core.yaml` foi usado pelos pares e define configurações essenciais como o nome do *peer*, nível de log, caminhos para o MSP e para os certificados TLS, bem como opções relacionadas à descoberta, endosso e acesso à cadeia de blocos. Os arquivos podem ser importados

pela prórpia imagem fornecida pelo Hyperledger Fabric quanto modificados pelo usuário e replicados para o volume. Também é possível modificar algumas configurações dos pares e do ordenador através de variáveis de ambiente. Por não haver necessidade de modificações tais arquivos não foram criados externamente, deixando a inicialização por conta das configurações padrões da última versão de imagem do hyperledger.

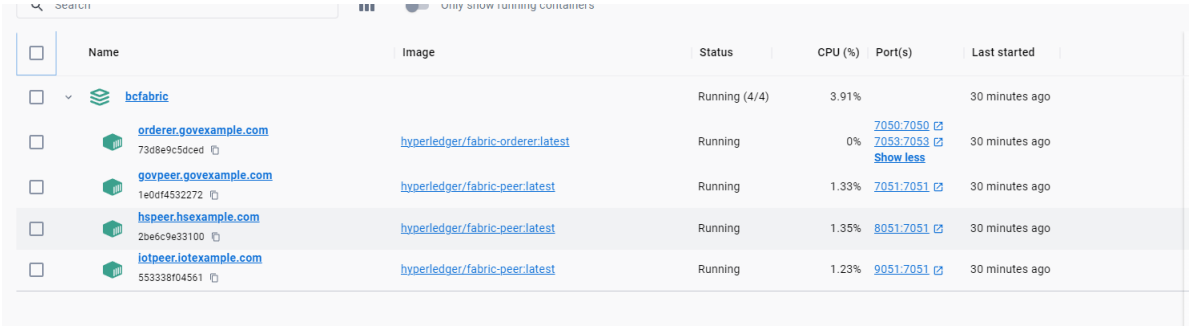

Figura 10 – contêineres ativos da rede no Docker Desktop

Nas figuras 11 e 12 é possível observar a criação bem sucedida do canal assim como o sucesso em juntar os pares ao canal utilizando os binários do HyperLedger.

```
isa@D20:~/tcc/bcfabric$ osnadmin channel join  -o orderer.govexample.com:7053   --ca-file $GOV_TLS_CA   --cl
ient-cert $GOV_CLIENT_CERT   --client-key  $GOV_CLIENT_KEY  --channelID healthchannel --config-block ./confi
g/healthblock.pb
Status: 201
{
        "name": "healthchannel",
        "url": "/participation/v1/channels/healthchannel",
        "consensusRelation": "consenter",
        "status": "active",
        "height": 1
}
```

Figura 11 – Criação do Canal através do Ordenador

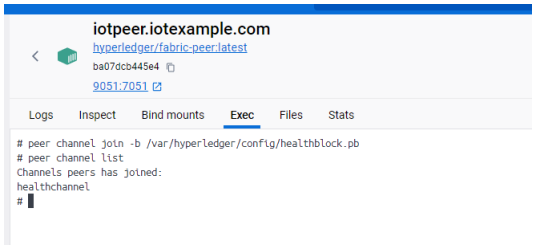

Figura 12 – Junção do par ao canal Existente

A partir das arquiteturas propostas, foi possível configurar e inicializar a rede utilizando o Hyperledger Fabric. Foram gerados os artefatos criptográficos, definidos os perfis de acesso das organizações e configurados os peers e orderers com sucesso. O canal foi criado, os peers foram conectados, e as entidades da rede puderam ser autenticadas via MSP. Embora a lógica do chaincode tenha sido projetada com base nos requisitos de acesso e privacidade das arquiteturas (como controle por organização e criptografia dos dados), a etapa de implementação e validação funcional do smart contract não foi finalizada. Dessa forma, os testes completos de leitura, gravação e verificação de permissões entre

as organizações não foram realizados. Mesmo com limitações, os resultados obtidos até o momento indicam a viabilidade da estruturação de uma rede permissionada baseada em *blockchain* para aplicações em saúde, e apontam caminhos concretos para finalização e testes futuros do smart contract.

5 Conclusão

Conclui-se, a partir da revisão bibliográfica, que a tecnologia *blockchain* apresenta-se como uma solução promissora para diferentes aplicações em saúde e sobre diferentes tipos de dados médicos e está em constante desenvolvimento de soluções voltadas à melhoria da eficiência, segurança e redução do consumo energético. Além disso, muitos artigos propõem a utilização de *blockchain*s combinadas com diferentes tecnologias emergentes visando diminuir possíveis vulnerabilidades, como o vazamento e roubo de dados.

As duas arquiteturas apresentadas sugeridas apresentam baixa complexidade de implementação enquanto ainda possuem viabilidade de aplicações reais. A primeira, baseada em um serviço notário, destaca-se pela simplicidade e alta maleabilidade para modificações de acordo com os interesses das organizações envolvidas. A segunda arquitetura foi sugerida para dispositivos IoMT (por representarem uma parcela considerável dos dados em saúde gerados atualmente) abre possibilidades para diferentes frameworks de monitoração e alertas em tempo real enquanto mantém a segurança dos dados.

Estratégias mais robustas para criptografia e segurança para impedir a cópia indevida de dados são recomendadas pois a encriptação dos dados sugerida segue um modelo básico que exibe o compartilhamento seguro de chaves anterior ao processo de compartilhamento. Múltiplas camadas de BC podem ser exploradas para encapsular dados de registros de consultas, principalmente no que diz respeito aos dispositivos IoT.

Para ambos os casos é necessário também explorar possibilidades de autonomia e empoderamento do paciente sobre suas próprias informações que justifiquem o uso da BC, uma vez que as arquiteturas propostas ainda consideram apenas transações entre sistemas de saúde e dispositivos eletrônicos. Um modelo mais ético e centrado no usuário deve conter mecanismos que permitam ao paciente definir, revisar e revogar permissões de acesso conforme seus próprios critérios. Isso pode ser viabilizado, por exemplo, com o uso de atributos vinculados à identidade do paciente, contratos inteligentes que implementem consentimento explícito e granular, ou mesmo interfaces que possibilitem a gestão dessas permissões de maneira acessível. Essa abordagem fortalece a governança dos dados em saúde, promovendo não apenas a interoperabilidade segura entre organizações, mas também o respeito à autodeterminação dos indivíduos em relação ao uso e compartilhamento de suas informações sensíveis.

A implementação das arquiteturas propostas não foi concluída por dificuldades encontradas durante o processo. A curva de aprendizado para a plataforma HyperLedger Fabric é elevada, especialmente para casos em que se deseja fugir dos exemplos genéricos e construir uma governança inteiramente customizada. Há escassez de conteúdo inicial

e intermediário acessível com foco em soluções personalizadas e múltiplas organizações. Além disso, existem mudanças consideráveis entre as versões 1.x e 2.x do Hyperledger afetando diretamente a criação de canais, o uso do `osnadmin`, a instalação de chaincode e a existência ou não do *system channel*, o que torna a compatibilidade entre materiais e tutoriais seguidos bem reduzida.

Ainda assim, as contribuições em termos de modelagem arquitetural e de governança apresentadas por este estudo permanecem válidas e úteis para orientar novas iniciativas no campo da saúde digital.

Além dos tópicos citados, como trabalhos futuros recomenda-se a exploração de frameworks complementares para o desenvolvimento que permitam maior granularidade em relação aos usuários e a escrita e instalação de *smart contracts*. Por fim, destaca-se a necessidade de investigações futuras que aprofundem a análise de aspectos como custo, segurança, desempenho, escalabilidade e governança das soluções propostas, a fim de validar sua aplicabilidade em contextos de larga escala e sensíveis como os sistemas de saúde.

6 Apêndice

6.1 Arquivos de Configuração da Governança Blockchain

6.1.1 crypto-config.yaml

```
#Ordenador
OrdererOrgs:
  - Name: Orderer
    Domain: govexample.com
    EnableNodeOUs: true
    Specs:
      - Hostname: orderer

#### Peers das organizacoes
PeerOrgs:
  - Name: GovOrg
    Domain: govexample.com
    EnableNodeOUs: true
    Specs:
      - Hostname: govpeer.govexample.com
        CommonName: govpeer.govexample.com
    Users:
      Count: 1

  - Name: HealthServiceOrg
    Domain: hsexample.com
    EnableNodeOUs: true
    Specs:
      - Hostname: hspeer.hsexample.com
        CommonName: hspeer.hsexample.com
    Users:
      Count: 1

  - Name: HealthDeviceOrg
    Domain: iotexample.com
    EnableNodeOUs: true
    Specs:
      - Hostname: iotpeer.iotexample.com
        CommonName: iotpeer.iotexample.com
    Users:
      Count: 1
```

6.1.2 configtx.yaml

```
Capabilities:
  Application: &ApplicationCapabilities
    V2_0: true
  Orderer: &OrdererCapabilities
    V2_0: true
```

```
  Channel: &ChannelCapabilities
    V2_0: true

Organizations:
  - &Orderer
    Name: Orderer
    ID: OrdererMSP
    MSPDir: ./crypto-config/ordererOrganizations/govexample.com/msp
    Policies: &OrdererPolicies
      Readers:
          Type: Signature
          Rule: "OR('OrdererMSP.member')"
      Writers:
          Type: Signature
          Rule: "OR('OrdererMSP.member')"
      Admins:
          Type: Signature
          Rule: "OR('OrdererMSP.admin')"
      Endorsement:
          Type: Signature
          Rule: "OR('OrdererMSP.member')"

  - &GovOrg
    Name: GovOrg
    ID: GovOrgMSP
    MSPDir: ./crypto-config/peerOrganizations/govexample.com/msp
    Policies: &GovOrgPolicies
      Readers:
          Type: Signature
          Rule: "OR('GovOrgMSP.member')"
      Writers:
          Type: Signature
          Rule: "OR('GovOrgMSP.member')"
      Admins:
          Type: Signature
          Rule: "OR('GovOrgMSP.admin')"
      Endorsement:
          Type: Signature
          Rule: "OR('GovOrgMSP.member')"
    AnchorPeers:
      - Host: govpeer.govexample.com
        Port: 7051

  - &HealthServiceOrg
    Name: HealthServiceOrg
    ID: HealthServiceOrgMSP
    MSPDir: ./crypto-config/peerOrganizations/hsexample.com/msp
    Policies: &HealthServiceOrgPolicies
      Readers:
          Type: Signature
          Rule: "OR('HealthServiceOrgMSP.member')"
      Writers:
          Type: Signature
          Rule: "OR('HealthServiceOrgMSP.member')"
      Admins:
          Type: Signature
          Rule: "OR('HealthServiceOrgMSP.member')"
```

```
      Endorsement:
          Type: Signature
          Rule: "OR('HealthServiceOrgMSP.member')"
    AnchorPeers:
      - Host: hspeer.hsexample.com
        Port: 8051

  - &HealthDeviceOrg
    Name: HealthDeviceOrg
    ID: HealthDeviceOrgMSP
    MSPDir: ./crypto-config/peerOrganizations/iotexample.com/msp
    Policies: &HealthDeviceOrgPolicies
      Readers:
          Type: Signature
          Rule: "OR('HealthDeviceOrgMSP.member')"
      Writers:
          Type: Signature
          Rule: "OR('HealthDeviceOrgMSP.member')"
      Admins:
          Type: Signature
          Rule: "OR('HealthDeviceOrgMSP.member')"
      Endorsement:
          Type: Signature
          Rule: "OR('HealthDeviceOrgMSP.member')"
    AnchorPeers:
      - Host: iotpeer.iotexample.com
        Port: 9051

Orderer: &OrdererDefaults

  OrdererType: etcdraft

  Addresses:
    - orderer.govexample.com:7050
  Policies:
    Readers:
        Type: ImplicitMeta
        Rule: "ANY Readers"
    Writers:
        Type: ImplicitMeta
        Rule: "ANY Writers"
    Admins:
        Type: ImplicitMeta
        Rule: "ANY Admins"

    BlockValidation:
        Type: ImplicitMeta
        Rule: "ANY Writers"

  EtcdRaft:
    Consenters:
      - Host: orderer.govexample.com
        Port: 7053
        ClientTLSCert: ./crypto-config/ordererOrganizations/govexample.com/orderers/orderer.govexample.com/tl
        ServerTLSCert: ./crypto-config/ordererOrganizations/govexample.com/orderers/orderer.govexample.com/tl

  BatchTimeout: 2s
```

```
  BatchSize:
    MaxMessageCount: 10
    AbsoluteMaxBytes: 98 MB
    PreferredMaxBytes: 512 KB

  Capabilities:
    <<: *OrdererCapabilities

Application: &ApplicationDefaults

  ACLs: &ACLsDefault

    lscc/ChaincodeExists: /Channel/Application/Readers
    lscc/GetDeploymentSpec: /Channel/Application/Readers
    lscc/GetChaincodeData: /Channel/Application/Readers
    lscc/GetInstantiatedChaincodes: /Channel/Application/Readers
    qscc/GetChainInfo: /Channel/Application/Readers
    qscc/GetBlockByNumber: /Channel/Application/Readers
    qscc/GetBlockByHash: /Channel/Application/Readers
    qscc/GetTransactionByID: /Channel/Application/Readers
    qscc/GetBlockByTxID: /Channel/Application/Readers
    cscc/GetConfigBlock: /Channel/Application/Readers
    cscc/GetConfigTree: /Channel/Application/Readers
    cscc/SimulateConfigTreeUpdate: /Channel/Application/Readers
    peer/Propose: /Channel/Application/Writers
    peer/ChaincodeToChaincode: /Channel/Application/Readers
    event/Block: /Channel/Application/Readers
    event/FilteredBlock: /Channel/Application/Readers
    _lifecycle/CheckCommitReadiness: /Channel/Application/Writers
    _lifecycle/CommitChaincodeDefinition: /Channel/Application/Writers
    _lifecycle/QueryChaincodeDefinition: /Channel/Application/Readers


  Policies: &ApplicationDefaultPolicies

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

  Organizations:

  Capabilities:
    <<: *ApplicationCapabilities
```

```
Channel: &ChannelDefaults
  Policies:
    Readers:
        Type: ImplicitMeta
        Rule: "ANY Readers"
    Writers:
        Type: ImplicitMeta
        Rule: "ANY Writers"
    Admins:
        Type: ImplicitMeta
        Rule: "ANY Admins"

  Capabilities:
    <<: *ChannelCapabilities

Profiles:

  GovOrdererGenesis:
      <<: *ChannelDefaults
      Orderer:
          <<: *OrdererDefaults
          Organizations:
              - <<: *Orderer

      Application:
        <<: *ApplicationDefaults
        Organizations:
            - <<: *GovOrg
            - <<: *HealthServiceOrg
            - <<: *HealthDeviceOrg
```

6.2 Docker-compose

6.2.1 docker-compose-base.yaml

```
version: '2'

networks:
  gov:

volumes:
  data-orderer.govexample.com:
  data-govpeer.govexample.com:
  data-hspeer.hsexample.com:
  data-iotpeer.iotexample.com:

services:

  orderer.govexample.com:
    container_name: orderer.govexample.com
    image: hyperledger/fabric-orderer:latest
    command: orderer
    environment:
      - FABRIC_CFG_PATH=/var/hyperledger/config
```

```
    - FABRIC_LOGGING_SPEC=ERROR
  volumes:
  - ${PWD}/config:/var/hyperledger/config
  - ${PWD}/crypto-config/ordererOrganizations/govexample.com/orderers/orderer.govexample.com/msp:/var/hyperledger
  - ${PWD}/crypto-config/ordererOrganizations/govexample.com/orderers/orderer.govexample.com/tls:/var/hyperledger
  - data-orderer.govexample.com:/var/ledger
  - ${PWD}/orderer.yaml:/var/hyperledger/config/orderer.yaml
  - ${PWD}/crypto-config/ordererOrganizations/govexample.com/tlsca:/var/hyperledger/tlsca
  ports:
    - 7050:7050
    - 7053:7053
  networks:
    - gov

govpeer.govexample.com:
  container_name: govpeer.govexample.com
  image: hyperledger/fabric-peer:latest
  working_dir: $HOME
  command: peer node start
  environment:
    - FABRIC_LOGGING_SPEC=WARNING
    - GOPATH=/opt/gopath
    - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=${COMPOSE_PROJECT_NAME}bcfabric_gov
    - CORE_PEER_ID=govpeer.govexample.com
    - CORE_PEER_LOCALMSPID=GovOrgMSP
    - CORE_PEER_ADDRESS=govpeer.govexample.com:7051
    - CORE_PEER_LISTENADDRESS=0.0.0.0:7051
    - CORE_PEER_CHAINCODEADDRESS=govpeer.govexample.com:7052
    - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:7052
    - CORE_PEER_GOSSIP_BOOTSTRAP=govpeer.govexample.com:7051
    - CORE_PEER_GOSSIP_EXTERNALENDPOINT=govpeer.govexample.com:7051
    - CORE_PEER_MSPCONFIGPATH=/var/hyperledger/msp
    - CORE_PEER_TLS_ENABLED=true
    - CORE_PEER_TLS_CERT_FILE=/var/hyperledger/tls/server.crt
    - CORE_PEER_TLS_KEY_FILE=/var/hyperledger/tls/server.key
    - CORE_PEER_TLS_ROOTCERT_FILE=/var/hyperledger/tls/ca.crt
  volumes:
    - ${PWD}/config:/var/hyperledger/config
    - ${PWD}/crypto-config/peerOrganizations/govexample.com/peers/govpeer.govexample.com/msp:/var/hyperledger/msp
    - ${PWD}/crypto-config/peerOrganizations/govexample.com/peers/govpeer.govexample.com/tls:/var/hyperledger/tls
    - /var/run/:/var/run/
    - data-govpeer.govexample.com:/var/hyperledger/production
  depends_on:
    - orderer.govexample.com
  ports:
    - 7051:7051
  networks:
    - gov

hspeer.hsexample.com:
  container_name: hspeer.hsexample.com
  image: hyperledger/fabric-peer:latest
  working_dir: $HOME
  command: peer node start
  environment:
    - FABRIC_LOGGING_SPEC=WARNING
    - GOPATH=/opt/gopath
```

```
    - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=${COMPOSE_PROJECT_NAME}bcfabric_gov
    - CORE_PEER_ID=hspeer.hsexample.com
    - CORE_PEER_LOCALMSPID=HealthServiceOrgMSP
    - CORE_PEER_ADDRESS=hspeer.hsexample.com:7051
    - CORE_PEER_LISTENADDRESS=0.0.0.0:7051
    - CORE_PEER_CHAINCODEADDRESS=hspeer.hsexample.com:7052
    - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:7052
    - CORE_PEER_GOSSIP_BOOTSTRAP=hspeer.hsexample.com:7051
    - CORE_PEER_GOSSIP_EXTERNALENDPOINT=hspeer.hsexample.com:7051
    - CORE_PEER_MSPCONFIGPATH=/var/hyperledger/msp
    - CORE_PEER_TLS_ENABLED=true
    - CORE_PEER_TLS_CERT_FILE=/var/hyperledger/tls/server.crt
    - CORE_PEER_TLS_KEY_FILE=/var/hyperledger/tls/server.key
    - CORE_PEER_TLS_ROOTCERT_FILE=/var/hyperledger/tls/ca.crt
  volumes:
    - ${PWD}/config:/var/hyperledger/config
    - ${PWD}/crypto-config/peerOrganizations/hsexample.com/peers/hspeer.hsexample.com/msp:/var/hyperledger/
    - ${PWD}/crypto-config/peerOrganizations/hsexample.com/peers/hspeer.hsexample.com/tls:/var/hyperledger/
    - /var/run/:/var/run/
    - data-hspeer.hsexample.com:/var/hyperledger/production
  depends_on:
    - orderer.govexample.com
  ports:
    - 8051:7051
  networks:
    - gov

iotpeer.iotexample.com:
  container_name: iotpeer.iotexample.com
  image: hyperledger/fabric-peer:latest
  working_dir: $HOME
  command: peer node start
  environment:
    - FABRIC_LOGGING_SPEC=WARNING
    - GOPATH=/opt/gopath
    - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=${COMPOSE_PROJECT_NAME}bcfabric_gov
    - CORE_PEER_ID=iotpeer.iotexample.com
    - CORE_PEER_LOCALMSPID=HealthDeviceOrgMSP
    - CORE_PEER_ADDRESS=iotpeer.iotexample.com:7051
    - CORE_PEER_LISTENADDRESS=0.0.0.0:7051
    - CORE_PEER_CHAINCODEADDRESS=iotpeer.iotexample.com:7052
    - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:7052
    - CORE_PEER_GOSSIP_BOOTSTRAP=iotpeer.iotexample.com:7051
    - CORE_PEER_GOSSIP_EXTERNALENDPOINT=iotpeer.iotexample.com:7051
    - CORE_PEER_MSPCONFIGPATH=/var/hyperledger/msp
    - CORE_PEER_TLS_ENABLED=true
    - CORE_PEER_TLS_CERT_FILE=/var/hyperledger/tls/server.crt
    - CORE_PEER_TLS_KEY_FILE=/var/hyperledger/tls/server.key
    - CORE_PEER_TLS_ROOTCERT_FILE=/var/hyperledger/tls/ca.crt
  volumes:
    - ${PWD}/config:/var/hyperledger/config
    - ${PWD}/crypto-config/peerOrganizations/iotexample.com/users/Admin@iotexample.com/msp:/var/hyperledger
    - ${PWD}/crypto-config/peerOrganizations/iotexample.com/peers/iotpeer.iotexample.com/tls:/var/hyperledg
    - /var/run/:/var/run/
    - data-iotpeer.iotexample.com:/var/hyperledger/production
  depends_on:
    - orderer.govexample.com
```

```
ports:
  - 9051:7051
networks:
  - gov
```

Referências

ANSAR, S.; GANESH, K. An effective blockchain-based smart contract system for securing electronic medical data in smart healthcare application. *Concurrency and Computation: Practice and Experience*, 2022. Received: 7 February 2022; Revised: 13 July 2022; Accepted: 7 August 2022. Citado na página 22.

BIGINI, G.; FRESCHI, V.; LATTANZI, E. A review on blockchain for the internet of medical things: Definitions, challenges, applications, and vision. *Electronics*, MDPI, v. 10, n. 24, p. 3033, 2021. Disponível em: <https://www.mdpi.com/2079-9292/10/24/3033>. Citado na página 24.

CHANDINI, A. G.; BASARKOD, P. I. Patient centric pre-transaction signature verification assisted smart contract based blockchain for electronic healthcare records. *Journal of Ambient Intelligence and Humanized Computing*, v. 14, p. 4221–4235, 2023. Original Research. Citado na página 23.

DINH, T. T. A. et al. Untangling blockchain: A data processing view of blockchain systems. *IEEE Transactions on Knowledge and Data Engineering*, v. 30, n. 7, p. 1366–1385, jul. 2018. Citado na página 19.

GRIGGS, K. N. et al. Healthcare blockchain system using smart contracts for secure automated remote patient monitoring. *Journal of Medical Systems*, v. 42, n. 7, p. 130, 2018. Citado 2 vezes nas páginas 23 e 24.

HARITHA, T.; ANITHA, A. Multi-level security in healthcare by integrating lattice-based access control and blockchain-based smart contracts system. *IEEE Access*, v. 11, p. 1–15, 2023. Citado na página 23.

HOSSEIN, K. M. et al. Bchealth: A novel blockchain-based privacy-preserving architecture for iot healthcare applications. *Computer Communications*, v. 180, p. 31–47, 2021. Citado 2 vezes nas páginas 23 e 24.

Hyperledger Fabric Contributors. *Hyperledger Fabric*. 2023. Acesso em: 22 jun. 2025. Disponível em: <https://hyperledger-fabric.readthedocs.io/en/release-2.5/whatis.html>. Citado na página 29.

ISMAIL, L.; MATERWALA, H.; HENNEBELLE, A. A scoping review of integrated blockchain-cloud (bcc) architecture for healthcare: Applications, challenges and solutions. *Sensors*, MDPI, v. 21, n. 11, p. 3753, 2021. Disponível em: <https://www.mdpi.com/1424-8220/21/11/3753>. Citado 2 vezes nas páginas 19 e 21.

KHATOON, A. A blockchain-based smart contract system for healthcare management. *Electronics*, v. 9, n. 1, p. 94, 2020. Citado na página 22.

KLEINAKI, A.-S. et al. A blockchain-based notarization service for biomedical knowledge retrieval. *Computational and Structural Biotechnology Journal*, v. 16, p. 288–297, 2018. Citado na página 22.

MAESA, D. D. F.; MORI, P. Blockchain 3.0 applications survey. *Journal of Parallel and Distributed Computing*, Elsevier, v. 138, p. 99–114, 2020. Citado na página 24.

PHAM, H. L.; TRAN, T. H.; NAKASHIMA, Y. A secure remote healthcare system for hospital using blockchain smart contract. In: IEEE. *2018 IEEE Globecom Workshops (GC Wkshps)*. [S.l.], 2018. p. 1–6. Citado na página 22.

PRADYUMNA, G. R. et al. Empowering healthcare with iomt: Evolution, machine learning integration, security, and interoperability challenges. *IEEE Access*, v. 12, 2024. Citado na página 21.

ROEHRS, A.; COSTA, C. A. da; RIGHI, R. M. Omniphr: A distributed architecture model to integrate personal health records. *Journal of Biomedical Informatics*, v. 71, p. 70–81, 2017. Citado na página 23.

ROSSETTO, A. G. de M.; SEGA, C.; LEITHARDT, V. R. An architecture for managing data privacy in healthcare with blockchain. *Sensors*, v. 22, n. 21, p. 8292, 2022. Citado na página 22.

SANTOS, J.; OLIVEIRA, M.; COSTA, A. Blockchain aplicado à saúde: Uma revisão sistemática da literatura. *Revista Brasileira de Informática em Saúde*, v. 20, n. 1, p. 10–25, 2024. Citado na página 19.

SHAHNAZ, A.; QAMAR, U.; KHALID, A. Blockchain technology in healthcare: A systematic review. *Healthcare Informatics Research*, Korean Society of Medical Informatics, v. 25, n. 2, p. 87–96, 2019. Citado na página 21.

SHARMA, A. et al. Blockchain based smart contracts for internet of medical things in e-healthcare. *Electronics*, v. 9, n. 10, p. 1609, 2020. Citado 2 vezes nas páginas 23 e 24.

STEPHANE, C. N.; MA, M. S. T. F. Armazenamento descentralizado no sistema Único de saúde brasileiro (sus) usando interplanetary file system (ipfs) e blockchain. *Revista de Direito*, Viçosa, v. 13, n. 1, 2021. Citado na página 23.

TAPSCOTT, D.; TAPSCOTT, A. *Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world*. [S.l.]: Penguin, 2016. Citado na página 19.

VAIYAPURI ADEL BINBUSAYYIS, V. V. T. Security, privacy and trust in iomt enabled smart healthcare system: A systematic review of current and future trends. *Miscellaneous*, v. 12, n. 2, 2021. Citado na página 21.

WISNER AUDREY LYNDON, C. A. C. K. The electronic health record's impact on nurses' cognitive work: An integrative review. *Elsevier ScienceDirect Journals*, v. 94, 2019. Citado na página 21.

ZHANG, P. et al. Fhirchain: Applying blockchain to securely and scalably share clinical data. *Computational and Structural Biotechnology Journal*, v. 16, p. 267–278, 2018. Citado na página 23.

ZHENG, Z. et al. An overview of blockchain technology: Architecture, consensus, and future trends. *Proceedings of the IEEE*, IEEE, v. 105, n. 9, p. 1527–1538, 2017. Citado na página 19.

ZHUANG, Y. et al. Generalizable layered blockchain architecture for health care applications: Development, case studies, and evaluation. *Journal of Medical Internet Research*, v. 22, n. 7, p. e19029, 2020. Citado na página 23.