Sistema de Recarga de VEs Baseado em Blockchain

Kauan Caio de A. Farias[1], **Nathielle C. Alves**[1], **Vitória T. dos Santos**[1]

[1]UEFS – Universidade Estadual de Feira de Santana
Av. Transnordestina, s/n, Novo Horizonte
Feira de Santana – BA, Brasil – 44036-900

{fariak, nalves, vtsantos}@ecomp.uefs.br

Resumo. *Com o aumento do uso de veículos elétricos (VEs) no Brasil, surge a necessidade de garantir transparência e segurança nas transações de recarga. Usuários enfrentam incertezas sobre o registro de suas reservas, enquanto empresas lidam com fraudes e disputas, afetando a confiabilidade dos dados. Este trabalho propõe um sistema de recarga de VEs baseado em blockchain, que registra de forma segura todas as transações de reserva, recarga e pagamento.*

1. Introdução

Nos últimos anos, o uso de veículos elétricos (VEs) tem se destacado de forma significativa no Brasil, impulsionado por incentivos à mobilidade sustentável e pela crescente busca por soluções energéticas mais limpas e eficientes. A Associação Brasileira do Veículo Elétrico (ABVE) reportou que, apenas no primeiro semestre de 2024, foram vendidos impressionantes 79.304 veículos leves eletrificados. Esse número representa um aumento expressivo de 146% em comparação ao mesmo período de 2023 [ABVE 2024].

Apesar do avanço na infraestrutura, os usuários de VEs ainda enfrentam desafios relacionados à transparência e segurança nas transações de recarga. A falta de um sistema confiável pode levar a fraudes, disputas por pontos de carregamento e incertezas sobre o registro de reservas e pagamentos. Esses problemas não apenas afetam a confiança dos usuários, mas também impactam negativamente a experiência do cliente e a reputação das empresas de recarga.

Diante desse cenário, este trabalho propõe o desenvolvimento de um sistema de recarga de VEs baseado em tecnologia blockchain. A implementação de um livro razão distribuído permitirá registrar de forma segura e transparente todas as transações de reserva, recarga e pagamento, promovendo a confiança entre usuários e empresas, além de garantir a auditabilidade das operações.

A solução foi implementada em Golang, com a utilização de Docker para orquestração dos ambientes e simulação de servidores distribuídos. Os testes realizados demonstraram a viabilidade da proposta, mostrando que é possível garantir a reserva de pontos de recarga em sequência e reduzir o risco de interrupções inesperadas durante o trajeto. O sistema se mostrou eficiente, flexível e alinhado às necessidades de um cenário cada vez mais conectado e colaborativo no setor de mobilidade elétrica.

O sistema foi desenvolvido utilizando uma rede Ethereum privada com consenso Proof of Authority (Clique) e contratos inteligentes em Solidity para garantir o registro transparente das operações. Scripts em Golang realizam a integração e automação das tarefas, enquanto o uso de Docker permite simular múltiplos participantes em um ambiente distribuído, assegurando transparência, descentralização e auditabilidade das transações.

2. Fundamentação Teórica

2.1. Blockchain

O termo Blockchain refere-se a uma estrutura de dados que é tanto descentralizada quanto distribuída, permitindo o registro seguro, transparente e imutável de transações em uma rede de computadores. Essa tecnologia foi inicialmente proposta por Satoshi Nakamoto em 2008 como a base para a moeda digital Bitcoin [NAKAMOTO 2008]. Desde então, sua aplicação se expandiu para diversas áreas além do setor financeiro, incluindo cadeias de suprimento, sistemas de saúde e, mais recentemente, infraestruturas de recarga de veículos elétricos.

Um blockchain pode ser entendido como um livro-razão digital distribuído, onde os dados são organizados em blocos que se conectam de forma sequencial e são protegidos por criptografia. Cada bloco contém um conjunto de transações, um código hash único e o hash do bloco anterior, formando uma cadeia contínua. Esse sistema assegura que os registros sejam imutáveis, pois qualquer tentativa de alteração em um bloco afetaria toda a cadeia subsequente, tornando fraudes facilmente detectáveis [Zheng et al. 2018]

Uma característica fundamental do blockchain é a utilização de mecanismos de consenso. Esses mecanismos, como Proof of Work (PoW), Proof of Stake (PoS) e Practical Byzantine Fault Tolerance (PBFT), garantem que todos os participantes da rede concordem sobre a validade das transações, mesmo sem a presença de uma autoridade central.

Os principais benefícios do blockchain incluem:

- **Descentralização:** Elimina a necessidade de intermediários e de um servidor central.
- **Transparência:** Todas as transações são acessíveis a todos os participantes da rede.
- **Segurança e integridade:** Os dados são criptografados, tornando quase impossível a alteração sem detecção.
- **Auditabilidade:** Cada transação pode ser verificada através do histórico completo armazenado na rede.

No contexto dos sistemas de recarga de veículos elétricos, a implementação do blockchain oferece uma maneira de garantir transparência, confiabilidade e auditabilidade nas transações de reserva, recarga e pagamento. Diferentes empresas de recarga podem atuar como nós dentro da rede blockchain, registrando operações de forma compartilhada, o que diminui a possibilidade de fraudes e facilita a auditoria entre as partes envolvidas. Além disso, a eliminação de um servidor centralizado aumenta a resiliência e a segurança do sistema.

Dentro desse cenário, destaca-se também a aplicação dos contratos inteligentes. Esses contratos são programas de computador que funcionam sobre a blockchain e têm a capacidade de executar automaticamente acordos predefinidos entre as partes envolvidas. Ao contrário dos contratos tradicionais, que dependem de intermediários ou da intervenção humana para serem cumpridos, os contratos inteligentes operam de forma autônoma, seguindo as regras definidas no código. Assim que as condições estipuladas são atendidas, as ações programadas são executadas de maneira automática e segura.

2.2. Modelo Cliente-Servidor

A arquitetura cliente-servidor é o modelo fundamental de comunicação em redes TCP/IP, organizando de forma eficiente e escalável a interação entre aplicações distribuídas. Nesse paradigma, o cliente solicita serviços e o servidor os processa e responde, permitindo que aplicações no lado cliente sejam leves, enquanto tarefas mais complexas ficam sob responsabilidade do servidor. Suas vantagens incluem portabilidade, modularidade e a possibilidade de expansão para arquiteturas mais complexas, como computação em nuvem e sistemas peer-to-peer, que ainda se baseiam, em seus níveis mais básicos, na interação cliente-servidor [COMER 2000].

2.3. Docker e Linguagem Golang

Docker é uma plataforma que permite empacotar, distribuir e executar aplicações em ambientes isolados chamados contêineres. Esses contêineres compartilham o kernel do sistema operacional, mas funcionam de forma independente (como ilustrado na Figura 1), garantindo portabilidade e consistência na execução, especialmente útil em sistemas distribuídos para facilitar o desenvolvimento, testes e deploy.

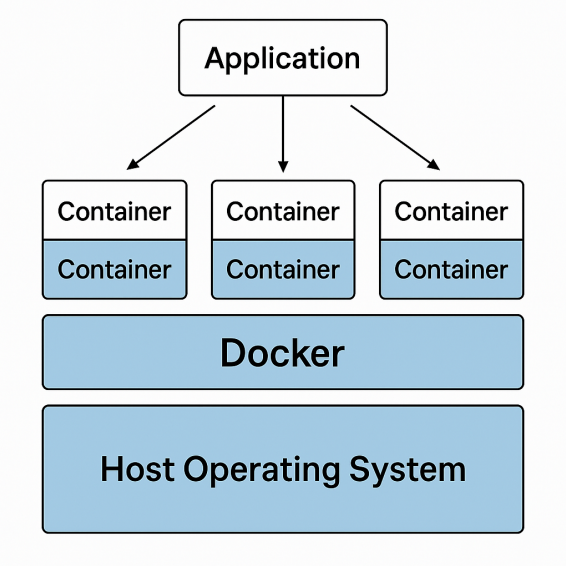

Figura 1. Diagrama de ilustração do funcionamento de containers Docker

A linguagem Go (ou Golang), desenvolvida pelo Google, é conhecida por sua simplicidade, desempenho e suporte nativo a concorrência. Uma de suas principais vantagens é o uso de goroutines, estruturas leves de execução concorrente que compartilham o mesmo espaço de memória e são gerenciadas pela própria runtime da linguagem.

Go oferece um modelo simples de concorrência com canais (chan) para comunicação segura entre goroutines, eliminando boa parte da complexidade associada a threads tradicionais. Essas características foram determinantes para a escolha da linguagem no desenvolvimento do sistema, pois possibilitaram a criação de um servidor capaz de gerenciar múltiplas conexões simultâneas de forma eficiente e com baixo custo computacional.

Embora o sistema utilize conexões TCP puras em vez de WebSockets, o conceito de comunicação em tempo real é essencial. WebSockets são uma abstração de comunicação persistente e bidirecional sobre TCP, amplamente utilizada em aplicações web modernas. Em Go, esse comportamento pode ser simulado por meio de conexões TCP tradicionais combinadas com goroutines para leitura e escrita assíncronas.

3. Metodologia

A metodologia adotada envolveu a construção de um ambiente blockchain privado, simulando múltiplos participantes do sistema, como estações de recarga e clientes (veículos elétricos), todos executados em contêineres Docker. O ambiente foi projetado para garantir a auditabilidade e a transparência das transações de reserva, recarga e pagamento das sessões de recarga de veículos elétricos. A rede Ethereum privada foi configurada com o mecanismo de consenso *Proof of Authority* (Clique), e todas as operações do sistema são gerenciadas por contratos inteligentes desenvolvidos em Solidity, assegurando o registro transparente e imutável das informações.

3.1. Ferramentas e Tecnologias

- **Linguagem Golang:** Linguagem de programação criada pelo Google, focada em desempenho, concorrência e simplicidade, utilizada para o desenvolvimento de aplicações distribuídas e de rede.
- **Solidity:** Linguagem de programação de alto nível utilizada para criar contratos inteligentes executados na blockchain Ethereum, permitindo a definição de regras automatizadas para transações digitais.
- **Geth (Go Ethereum):** Cliente oficial da rede Ethereum desenvolvido em Go, utilizado para executar um nó da blockchain, interagir com contratos inteligentes e participar da rede Ethereum.
- **Docker-Compose:** Ferramenta que facilita a configuração e execução de ambientes com múltiplos containers Docker, permitindo a orquestração de serviços de forma simples e automatizada.
- **go-ethereum:** Biblioteca oficial em Go para integração e comunicação de aplicações com a blockchain Ethereum, possibilitando a criação, leitura e interação com contratos inteligentes e transações na rede.

3.2. Arquitetura do Sistema

A arquitetura do sistema é composta por três principais camadas:

1. **Camada Blockchain:**
    - **Nó Ethereum (Geth):** Um nó privado da blockachain Ethereum é executado em um container Docker, configurado com o consenso Clique (*Proof of Authority*).
    - **Contrato Inteligente (Solidity):** O contrato gerencia o registro de postos, agendamento e controle das sessões de recarga de veículos elétricos.
2. **Camada de Backend/Integração:**
    - **Scripts em Go:** Scripts escritos em Go são responsáveis por: gerar carteiras e endereços, implantar (*deploy*) o contrato inteligente na blockchain e interagir com o contrato (consultas, transações, reservas, etc).

- **Bibliotecas:** Uso das bibliotecas *go-ethereum* e *hdwallet* para comunicação com a blockchain e manipulação de carteiras.

3. **Camada de Orquestração e Infraestrutura:**
   - **Docker & Docker Compose:** Utilizados para facilitar o setup, isolamento e execução do nó Ethereum e seus arquivos de configuração.
   - **Shell Scripts:** Automatizam a inicialização do nó, importação de contas e configuração da rede.

Usuários interagem com o sistema via scripts Go, que se comunicam com o nó Ethereum (Geth) para registrar postos, agendar sessões e registrar recargas, tudo registrado de forma transparente e imutável na blockchain.

3.3. Implementação

3.3.1. Módulo Carro

O carro elétrico, no sistema proposto, desempenha o papel de cliente da rede distribuída, sendo responsável por realizar reservas de estações, solicitar, efetuar pagamentos e consultar seu histórico de operações. Cada carro simulado é representado por um script Go, executado em um container Docker, que interage diretamente com os contratos inteligentes implantados na blockchain Ethereum privada. Essas interações garantem a transparência, segurança e auditabilidade de todas as operações realizadas.

Principais funcionalidades:

1. **Solicitar Reserva de Estação:**
   - O carro envia uma requisição (via script Go) para reservar uma estação de recarga disponível.
   - Essa solicitação gera uma transação na blockchain, registrada via contrato inteligente.
2. **Solicitar Início da Recarga:**
   - Ao chegar na estação, o carro envia um comando para iniciar a recarga.
   - A estação verifica, via blockchain, se o carro tem uma reserva válida e autorização. Se aprovado, a estação inicia a recarga.
3. **Realizar ou Confirmar Pagamento:**
   - Ao finalizar a recarga, o carro pode iniciar a transação de pagamento via contrato inteligente, debitando o valor corresponde. Essa transação fica registrada de forma transparente na blockchain.
4. **Consultar Histórico de Recargas:**
   - O carro pode consultar seu histórico de reservas, recargas e pagamentos no *ledger* distribuído, garantindo auditabilidade.

3.3.2. Estação de Recarga

A estação de recarga atua como ponto de atendimento no sistema, responsável por validar reservas na blockchain, autorizar o início das sessões de recarga, executar o carregamento do veículo e registrar na blockchain o início e término das recargas. Todas as interações são registradas de forma imutável no *ledger* distribuído, assegurando rastreabilidade e segurança das operações.

Principais funcionalidades:

1. **Validar Reserva do Carro:**
    - Ao receber uma solicitação de recarga, a estação consulta o contrato inteligente na blockchain para verificar se o carro possui uma reserva válida para aquele horário e posto.
2. **Autorizar Início da Recarga:**
    - Se a reservar for válida, a estação autoriza o início da recarga.
3. **Executar a Recarga:**
    - Realiza o carregamento do veículo.
4. **Registrar Término da Recarga:**
    - Ao finalizar a recarga, a estação envia uma transação ao contrato inteligente, informando o término da sessão.

3.3.3. Módulo Blockchain

O módulo blockchain é responsável por armazenar, validar e propagar todas as transações do sistema de forma descentralizada e imutável. Cada empresa participante opera seu próprio nó na rede Ethereum privada, configurada com o mecanismo de consenso Proof of Authority (Clique). Esse módulo mantém o ledger sincronizado entre os participantes e executa os contratos inteligentes, garantindo que todas as operações de reserva, recarga e pagamento sejam registradas com segurança, transparência e auditabilidade.

Principais funcionalidades:

1. **Registro de Reservas:**
    - Toda vez que um carro solicita uma reserva de estação, essa solicitação é registrada em um contrato inteligente.
2. **Verificação de Autorização de Recarga:**
    - Antes de iniciar a recarga, a estação consulta a blockchain para verificar se há uma reserva ativa e válida para aquele veículo.
3. **Pagamento com Registro Transparente:**
    - Ao final da recarga, o pagamento é realizado por meio de uma transação blockchain.
4. **Registrar Término da Recarga:**
    - Ao finalizar a recarga, a estação envia uma transação ao contrato inteligente, informando o término da sessão.

3.3.4. Módulo de Contrato Inteligente

O módulo blockchain é responsável por armazenar, validar e propagar todas as transações do sistema de forma descentralizada e imutável. Cada empresa participante opera seu próprio nó na rede Ethereum privada, configurada com o mecanismo de consenso Proof of Authority (Clique). Esse módulo mantém o ledger sincronizado entre os participantes e executa os contratos inteligentes, garantindo que todas as operações de reserva e recarga sejam registradas com segurança, transparência e auditabilidade.

Principais funcionalidades:

1. **Gerenciamento de Reservas:**

- Ao receber uma solicitação de reserva, o contrato verifica se a estação está disponível naquele momento e registra a operação.

2. **Autorização para Início de Recarga:**
   - Quando um veículo chega à estação e solicita o início da recarga, o contrato inteligente valida se a reserva é válida. Em caso positivo, a estação é liberada para começar o carregamento.
3. **Registro Imutável das Transações:**
   - Cada ação intermediada pelo contrato — como reservas confirmadas e recargas realizadas — é registrada na blockchain.

3.3.5. Módulo de Integração

O módulo de integração é responsável por conectar os componentes do sistema (veículos, estações e blockchain), automatizando tarefas essenciais para o funcionamento e interação com os contratos inteligentes.

Principais funcionalidades:

1. **Geração de Carteiras:**
   - Cada entidade participante (carro ou uma estação) precisa de uma carteira digital única para interagir com a blockchain.
2. **Implantação de Contratos Inteligente:**
   - Realiza o deploy do contrato inteligente na blockchain privada.
   - Usa a biblioteca go-ethereum para assinar e enviar a transação de implantação.
3. **Interação com a Blockchain:**
   - Scripts para consultar dados, registrar agendamentos, iniciar/finalizar sessões e ler o estado do contrato inteligente.

4. Resultados

Os resultados do projeto demonstraram que o ambiente do sistema foi configurado e executado com sucesso utilizando Docker e Docker Compose, permitindo a inicialização e o isolamento dos componentes de forma eficiente. O contrato inteligente desenvolvido em Solidity foi implantado corretamente na blockchain privada por meio de scripts Go, possibilitando a integração entre o backend e a rede blockchain. Todas as operações de agendamento, início e término das sessões de recarga foram registradas de maneira transparente e imutável na blockchain, garantindo a auditabilidade e a rastreabilidade das ações realizadas.

O módulo de geração de carteiras permitiu criar múltiplos endereços para diferentes participantes, simulando estações de recarga e veículos distintos na rede, o que facilita a expansão do sistema para cenários mais complexos e multiusuário. O uso do consenso Proof of Authority (Clique) proporcionou validação rápida das transações e blocos, tornando o sistema adequado para ambientes permissionados e controlados, como redes privadas de empresas ou consórcios.

De modo geral, o sistema demonstrou ser viável para garantir transparência, segurança e facilidade de auditoria nas operações de recarga de veículos elétricos, além de apresentar flexibilidade para futuras expansões e integrações com outros serviços relacionados à mobilidade elétrica.

5. Conclusão

A implementação deste projeto evidenciou o potencial do uso de blockchain privada para o gerenciamento transparente e auditável de sessões de recarga de veículos elétricos. A utilização de contratos inteligentes permitiu automatizar e registrar de forma imutável todas as operações essenciais do sistema, enquanto o ambiente orquestrado com Docker garantiu praticidade e reprodutibilidade na implantação dos componentes. O consenso Proof of Authority (Clique) mostrou-se adequado para cenários permissionados, proporcionando desempenho e controle sobre a validação das transações.

O sistema demonstrou ser capaz de registrar de maneira confiável e transparente todas as etapas do processo de recarga, desde o agendamento até a finalização das sessões, promovendo maior confiança entre os participantes. Dessa forma, o projeto atingiu seus objetivos de demonstrar a viabilidade técnica e os benefícios de transparência, segurança e auditabilidade proporcionados pela tecnologia blockchain no contexto da recarga de veículos elétricos.

Referências

ABVE (2024). 80 mil veículos eletrificados somente no primeiro semestre. Acesso em 04 jun. 2025.

COMER, D. E. (2000). *Internetworking with TCP/IP: Client-Server Programming and Applications*. Prentice Hall, Upper Saddle River, NJ.

NAKAMOTO, S. (2008). Bitcoin: A peer-to-peer electronic cash system. Acesso em 04 jun. 2025.

Zheng, Z., Xie, S., Dai, H., Chen, X., and Wang, H. (2018). Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14(4):352–375.