MARINHA DO BRASIL
DIRETORIA DE ENSINO DA MARINHA
CENTRO DE INSTRUÇÃO ALMIRANTE ALEXANDRINO

CURSO DE APERFEIÇOAMENTO AVANÇADO EM
SEGURANÇA DA INFORMAÇÃO E COMUNICACÕES

TRABALHO DE CONCLUSÃO DE CURSO

APLICABILIDADE DA TECNOLOGIA DO BLOCKCHAIN NA MARINHA DO BRASIL

PRIMEIRO-TENENTE RODRIGO ANTONIO RIBEIRO DE ALMEIDA

Rio de Janeiro
2023

PRIMEIRO-TENENTE RODRIGO ANTONIO RIBEIRO DE ALMEIDA

APLICABILIDADE DA TECNOLOGIA DO BLOCKCHAIN NA MARINHA DO BRASIL

Monografia apresentada ao Centro de Instrução Almirante Alexandrino como requisito parcial à conclusão do Curso de Aperfeiçoamento Avançado em Segurança Da Informação e Comunicações.

Orientadores:
Prof. Dr. Carlos Vinicio Rodriguez
Capitão-Tenente Warlley Paulo Freire

CIAA
Rio de Janeiro
2023

PRIMEIRO-TENENTE RODRIGO ANTONIO RIBEIRO DE ALMEIDA

APLICABILIDADE DA TECNOLOGIA DO BLOCKCHAIN NA MARINHA DO BRASIL

Monografia apresentada ao Centro de Instrução Almirante Alexandrino como requisito parcial à conclusão do Curso de Aperfeiçoamento Avançado em Segurança Da Informação e Comunicações.

Aprovada em _____

Banca Examinadora:

_____

Capitão de Mar e Guerra (RM1-EN) Gian Karlo Huback Macedo de Almeida – CIAA

_____

Carlos Vinicio Rodriguez, D. Sc – PUC Rio

_____

Capitão-Tenente Warlley Paulo Freire – CoNavOpEsp

CIAA
Rio de Janeiro
2023

Dedico esse trabalho a minha família, amigos e instrutores, que me ajudaram durante minha jornada em busca do conhecimento necessário para desenvolver a presente monografia.

AGRADECIMENTOS

À medida que concluo esta etapa significativa de minha carreira naval, sinto-me profundamente agradecido por todas as pessoas que estiveram ao meu lado durante minha singradura, apoiando-me e fazendo deste trabalho de conclusão de curso uma realidade. Gostaria de expressa minha sincera gratidão a todos que de alguma forma me auxiliaram a chegar até aqui.

Primeiramente, desejo expressar minha gratidão a minha família que sempre acreditou em mim e me apoiou incondicionalmente. Agradeço a minha mãe por seu amor, encorajamento e apoio emocional ao longo de todos esses anos. Cada conquista minha é um reflexo direto da confiança que depositou em mim.

Agradeço também a meus orientadores, Prof. Dr. Carlos Vinicio Rodriguez e Capitão-Tenente Warlley Paulo Freire, por seu apoio inestimável e orientação durante todo o processo. Suas sugestões, opiniões e paciência foram cruciais para que o presente trabalho tosse bem-sucedido. Sem a orientação de vocês, não seria possível alcançar o nível de excelência desejado.

Agradeço aos meus amigos e colegas de turma, por compartilharem esta jornada comigo. Nossas conversas estimulantes, colaborações e amizade significaram muito para mim. O curso de aperfeiçoamento avançado não teria sido o mesmo, sem a amizade de vocês. Em especial aos amigos do curso de Segurança da Informação e Comunicações, faco agradecimento especial por todos os momentos vividos e amizade que sempre tivemos e tivemos a oportunidade de reavivar no ano de 2023.

Além disso, estendo meus agradecimentos ao Coordenador do curso de Segurança da Informação e Comunicações, CMG-RM1 Huback, cuja condução permitiu que o presente curso fosse proveitoso em termos de conhecimentos adquiridos que certamente serão de suma importância em nossas carreiras na Marinha do Brasil. Ressalto também, toda boa vontade do Comandante Huback em proporcionar uma boa divisão das diversas disciplinas ao longo do ano de 2023. Sem essa ajuda, o presente trabalho certamente não teria alcançado o nível de excelência desejado.

Estendo ainda meus agradecimentos a todos os militares, oficiais e praças, e servidores civis do Centro de Instrução Almirante Alexandrino, cujo trabalho aguerrido permitiu que o Curso de Aperfeiçoamento Avançado do ano de 2023, ocorresse de forma tranquila e sem nenhum tipo de problemas relacionados a organização. Sem o trabalho de vocês, nada disso seria possível.

Este trabalho de conclusão de curso é resultado de um esforço coletivo e de inúmeras contribuições. Ressalto meu agradecimento a todo corpo docente que conduziu o referido curso. Embora apenas os nomes citados acima sejam meus orientadores formais, toda instrução oferecida, conhecimentos passados e dúvidas tiradas pelo corpo docente, de certa forma contribuíram para a execução deste trabalho.

*“*O maior inimigo do conhecimento
não é a ignorância, é a ilusão do
conhecimento*”*
Stephen Hawking.

APLICABILIDADE DA TECNOLOGIA DO BLOCKCHAIN NA MARINHA DO BRASIL

Resumo

O ambiente digital contemporâneo enfrente um crescente desafio relacionado a segurança com as ameaças cibernéticas tornando-se cada vez mais sofisticadas e prejudiciais. Este trabalho de conclusão de curso abordo esse problema crítico, destacando a necessidade de medidas eficazes para proteger sistemas e dados sensíveis. Para ilustras a urgência desse problema, o malware Stuxnet é apresentado como exemplo emblemático de um ataque cibernético. São apresentados os fundamentos da tecnologia *blockchain*, dentre os quais ressalta-se sua estrutural, princípio de funcionamento e conceitos importantes. A *blockchain*, é uma tecnologia de registro distribuído emergiu como uma tecnologia inovadora para questões de segurança e confiabilidade em ambientes digitais. Sua natureza imutável oferece potencial para reforçar a integridade dos dados. Além disso, foram desenvolvidos dois protótipos de *blockchain*, utilizadas para leitura de sensores e para comunicação segura. Este trabalho destaca a importância da tecnologia citada como uma solução eficaz para fortalecer a segurança em ambientes virtuais, reduzindo vulnerabilidades e promovendo confiabilidade de sistemas. Com as implementações práticas, demonstra-se a validade e versatilidade dessa tecnologia na resolução de problemas de segurança cibernética.

Palavras- chave: Blockchain, malware, segurança, tecnologia.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTAS DE SIGLAS E ABREVIATURAS

C3I	*Command, control, communication and intelligence*
CLP	Controlador Lógico Programável
CPU	*Central Processing Unit*
IMO	*International Maritime Organization*
PDA	*Personal Digital Assistant*
PND	Plano Nacional de Defesa
RDS	Rádio Definido por Software
SDK	*Software Development Kit*
SSH	*Secure Shell*

SUMÁRIO

1 INTRODUÇÃO

É urgente a necessidade de aumentar o conhecimento sobre ameaças de ciberataques e suas vulnerabilidades. Além disso, a segurança dos meios marítimos depende da cibersegurança. (*International Maritime Organization*, 2017) Nesse sentido, a resolução MSC 428 da IMO, afirma que um sistema de gestão de segurança de meios marítimos deve considerar os riscos cibernéticos.

O PND (Brasil, 2020a), determina que o espaço cibernético requer especial atenção de forma a garantir o funcionamento dos sistemas de informação, de gerenciamento e de comunicações de interesse social. O referido normativo, determina, também, que na área da ciência e tecnologia, o desenvolvimento da tecnologia cibernética deve ser priorizado.

Além disso, no setor cibernético, devem ser incluídas tecnologias de comunicações entre unidades das forças armadas. Tal integração implica em aprimorar a segurança da Informação e das Comunicações e a Segurança Cibernética em todas as instâncias do Estado. Dessa forma, faz-se necessário desenvolver o preparo e emprego da estrutura do sistema de defesa cibernética buscando fomentar pesquisa, desenvolvimento e inovação com auxílio da comunidade acadêmica. (Brasil, 2020a)

No âmbito militar, quatro colisões de *destroyers*[1] americanos, em 2017, podem fornecer a percepção de possíveis consequências dos ciberataques a meios navais. Em que pese a Marinha dos Estados Unidos não confirmar que os problemas ocorridos foram fruto de ataques cibernéticos, essa tese pode ser avaliada através da correlação entre dados disponíveis em fontes abertas, obtendo-se um *insight* sobre como os meios navais podem ser afetados por uma arma cibernética sofisticada. (Freire, 2023)

1.1 Apresentação do Problema

A ampliação do uso de ambientes virtuais traz consigo novas vulnerabilidades. Os ciberataques, devido à sua capacidade de causar danos a diversas plataformas, têm se tornado uma preocupação estratégica prioritária não apenas para os Estados-Nações, mas também para a comunidade internacional. (Nunes, 2012)

Segundo Cornella et al. (2020), em 2017, 235GB de dados sigilosos pertencentes às inteligências da Coreia do Sul e dos Estados Unidos, foram roubados pela Coreia do Norte.

[1] Destroyer é uma embarcação militar rápida utilizada para uma variedade de funções. (Britannica, 2023)

No mesmo ano, a comissão europeia afirmou que mais de 4.000 ataques de ransomware[2] foram detectados por dia.

Um exemplo de malware extremamente sofisticado é o Stuxnet. Ele é uma criação complexa que possui muitos componentes e funcionalidades. Inicialmente foi desenvolvido para atacar controles industriais ou sistemas similares. Seu objetivo final é reprogramar esses controles modificando os códigos de um Controlador Lógico Programável (CLP) para fazer com que ele trabalhe de maneira que as alterações feitas pelo atacante permanecem invisíveis ao atacado. (Falliere et al., 2011)

Um cenário possível de um ataque utilizando o Stuxnet, segundo Falliere et al. (2011), pode ocorrer em sistemas de controles que utilizam vários CLPs, ainda que tais sistemas operam apenas em redes internas, estratégia de defesa conhecida como *‘‘air gap"*. Antes de iniciar o cenário de ataque proposto, os atacantes precisam conhecer os esquemas dos sistemas de controle. O próximo passo do ataque, é o desenvolvimento de um ambiente espelhado, que inclua CLPs para testar os códigos. Além disso, os certificados digitais de terceiros que entraram legitimamente nas instalações físicas podem ser utilizados para assinar os códigos maliciosos que contenham os arquivos de driver a serem inseridos na estrutura do sistema.

Contudo, para efetivamente infectar o alvo, o Stuxnet precisa ser introduzido no ambiente de alguma forma. Uma das possíveis formas de se realizar essa infecção, é infectar um terceiro voluntário ou desconhecido que possa ter acesso a instalação, ou ainda um servidor interno. Para tal, pode-se utilizar a utilização de mídias removíveis ou até mesmo infectar equipamentos de diagnostico de empresas de manutenção. (Falliere et al., 2011)

Após a infecção bem-sucedida, o Stuxnet passa a buscar dentro da infraestrutura alvo computadores usados para programar os CLPs. Uma vez que o malware consegue infectar um alvo adequado, ele executa a modificação no código programado no CLP. O Stuxnet ainda tem a capacidade de ocultar suas modificações, e assim, o usuário que tentar verificar a existência de códigos maliciosos não conseguirá perceber sua presença. (Falliere et al., 2011)

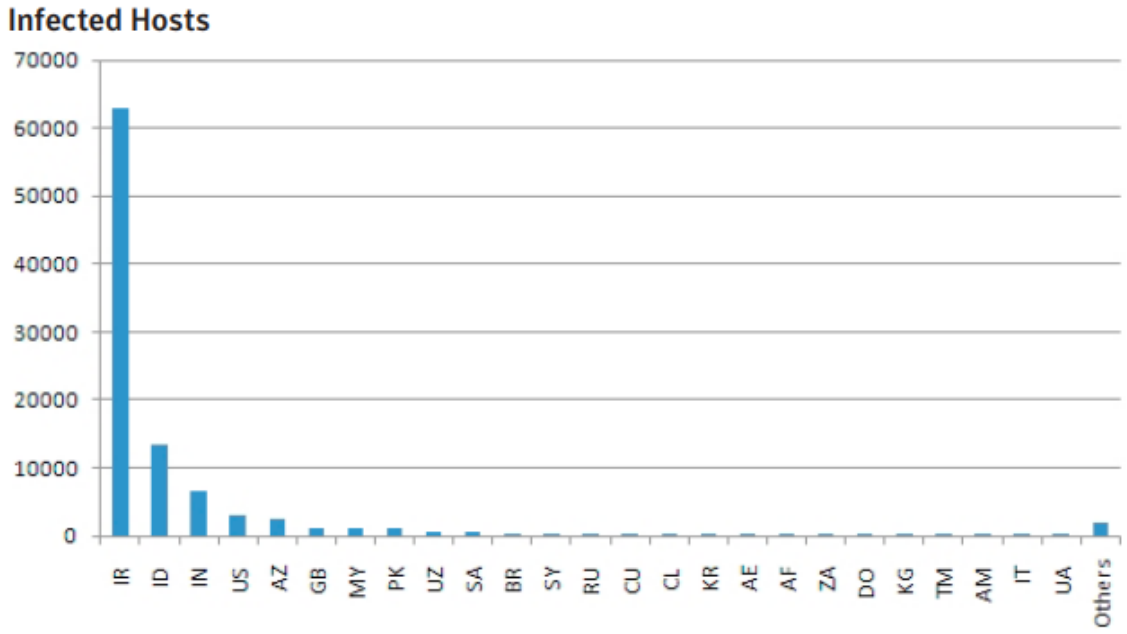
Tendo em vista o possível cenário de ataque proposto, onde para iniciar o ataque o agente malicioso precisa conhecer a infraestrutura dos sistemas de controle. (Falliere et al., 2011) é possível perceber que é necessário ter cuidado com a implementação prática dos sistemas de controle dos novos meios da Marinha do Brasil. Segundo o site Defesanet (2023) as novas fragatas classe Tamandaré terão sua tecnologia fornecida pela empresa alemã, Thyssenkrupp através de sua plataforma de construção de navios de defesa Classe MEKO, que

[2] Malware que encripta arquivos de um computador infectado. (Richardson; North, 2019)

já opera em 15 países. A empresa brasileira EMBRAER ficará encarregada pela integração entre sensores e armamentos ao sistema de gerenciamento de combate. A subsidiária da Thyssenkrupp Marine Systems, Atlas Elektronik, será uma das encarregadas do enlace de dados táticos e do sistema integrado de gerenciamento da plataforma, juntamente com a americana L3Harris.

Para corroborar com a importância do assunto, pode-se citar um sistema de monitoramento do Stuxnet, proposto pela empresa Symantec em julho de 2010. Nesse sistema, é possível observar a ampla capacidade de infecção de malware, inclusive em diversos países, conforme figura 1. Em setembro daquele ano, já haviam sido infectados mais de 100.000 hospedeiros. (Falliere et al. 2011)

Figura 1 - Hosts infectados com o Stuxnet em setembro de 2010.



Fonte: Stuxnet Dossier (Falliere et al., 2011)

1.2 Justificativa e Relevância

Atualmente, a tecnologia *blockchain* possui uma ampla variedade de implementações que são usadas para gerenciamento de dados em indústrias. O modo com que essa tecnologia consegue proteger e assegurar os dados com transparência e garantindo a sua integridade e disponibilidade faz com que seja uma escolha robusta para aplicações onde dados sensíveis estejam envolvidos. Hoje em dia a tecnologia em questão é utilizada em aplicações como por exemplo: cadeias logísticas, assistência médica, finanças e governo. (Idrees, 2021)

Segundo Zhang et al (2020), uma *blockchain* permite aliviar a centralização em serviços de nuvem e armazenamento em nuvem. Um mecanismo que combina a capacidade de uma *blockchain* pode ser utilizado para aumentar a segurança e confiança dos dados armazenados pela computação de borda. Fan et al. (2018) propôs um sistema de gerenciamento de informações baseado em *blockchain*. Ainda, de acordo com Li et al. (2018), um sistema de preservação de informações médicas baseado em *blockchain* foi desenvolvido e implementado com a finalidade de providenciar armazenamento confiável para os dados.

Ademais, Freire et al (2021) propôs um sistema de monitoramento marítimo que agrega dados coletados através de dispositivos instalados em boias e aeronaves não tripuladas utilizando tecnologia *blockchain* para garantir integridade, autenticidade e disponibilidade dos dados.

O setor de defesa tem que lidar com a manipulação de dados sensíveis. A manipulação de dados, por sua vez, passa por um complexo processo de gerenciamento. Então, pode-se dizer que a rastreabilidade e a transparência são dois problemas complexos e cruciais que garantem a robustez e confiabilidade na indústria de defesa. No setor militar, é altamente recomendado rastrear as informações de maneiras eficientes e garantir a autenticidade das fontes. (Akter, 2019)

Os sistemas militares demandam formas inovadoras de combinar unidades físicas e unidades lógicas, combinando softwares, hardware e para completar as missões determinadas. Diante desse cenário, surge a figura dos sistemas C3I[3] *(Command, control, communication and intelligence)*. Nesses sistemas, é possível agregar grande quantidade de informações do estado-maior e comandantes para implementar e monitorar questões operacionais digitalizando a estrutura do banco de dados distribuído. (Akter, 2019)

1.3 Objetivos

Este trabalho tem como objetivo geral expor características da tecnologia *Blockchain* mencionando aplicações onde tal tecnologia já é empregada e propor possíveis utilizações da blockchain no contexto da Marinha do Brasil de forma a mitigar riscos em comunicações entre meios ou entre meios e distritos, e oferecer uma camada de segurança na leitura de saídas de sensores utilizados a bordo.

[3] Sistemas que cooperam com estrutura de Comando e Controle para trocar informações especificas ou para sincronizar atividades comuns no teatro de operações. Além disso, incluem componentes de rede que proporcionam apoio a computação distribuída. (Djordjevic-Kajan et al., 1997)

Como forma de exemplificar a utilização da tecnologia proposta, o objetivo específico deste trabalho é desenvolver uma solução de software baseada nos conceitos da arquitetura de uma *blockchain*, que será apresentada, utilizando a linguagem de programação Python para criar duas *blockchains* e utilizá-las para realizar a leitura de sensores e troca de comunicações de comando e controle entre meios ou meios e distritos navais. Esse protótipo inicial busca demonstrar como a tecnologia pode ser aplicada para garantir a integridade e disponibilidade de dados críticos na Marinha do Brasil.

2 REFERENCIAL TEÓRICO

Nesta sessão, será realizada uma revisão bibliográfica para que sejam explicados os conceitos relevantes da tecnologia apresentada. Além disso, serão utilizados trabalhos de terceiros já publicados que corroboram com os objetivos da presente monografia de forma a demonstrar a viabilidade do tema.

2.1 A tecnologia *Blockchain*

A primeira implementação ampla do *blockchain* ocorreu em 2008, através da utilização da criptomoeda Bitcoin, que consistia em um sistema descentralizado de pagamentos. A tecnologia apresentava o conceito de descentralização para coletar novas transações que seriam processadas em blocos e conectaria esses blocos de modo a fazer uma cadeia, através do processo de mineração. (Platt; Mcburney, 2023)

Enquanto a tecnologia *blockchain* ganhava interesse da comunidade cientifica devido a sua capacidade de realizar transações anônimas como no caso do Bitcoin, outros pesquisadores notaram que a tecnologia poderia ser aplicada em diversas aplicações na indústria. Nesse contexto, surge a *blockchain* Ethereum, que permite aos usuários, entre outras coisas, mover fundos de acordo com regras pré-definidas e registrar débitos. Dessa forma, o Ethereum, foi definido como um livro-razão que pode ser usado para construir novos programas empregando computação distribuída. De fato, essa ferramenta evoluiu e migrou para diversos campos de atuação tais como, saúde, distribuição de ingressos e sistemas de votação. (Popovski; Soussou, 2018)

Segundo Treleaven et al (2017) uma *blockchain* consegue crescer continuamente de forma distribuída, onde cada bloco é selado criptograficamente com uma impressão digital

gerada por uma função *hash*[4]. O crescimento ocorre, pois, cada bloco é conectado ao bloco anterior referindo-se ao valor do seu *hash*. Os sistemas de *blockchain* são resilientes e conseguem operar em redes descentralizadas que não necessitam de um servidor central, outro ponto relevante é que o *blockchain* utiliza um protocolo de código aberto que permite uma boa integração sem depender de uma terceira parte para executar as transações, além disso, os sistemas de *blockchain* possuem transparência pois todas as mudanças são visíveis a todas as partes. Por isso, pode-se dizer que o *blockchain* pode ser usado para que aplicações e usuários possam operar com alto grau de confiança, uma vez que as transações, quando realizadas, são imutáveis.

2.2 Conceitos Importantes

Segundo Kakavand et al. (2017), para entender a arquitetura *blockchain*, com suas questões de performance, privacidade e segurança, deve-se entender alguns conceitos técnicos. Os principais conceitos são:

1. Nó (*node*): Uma *blockchain* é mantida pelo seu software sendo executado em um computador, chamado nó. Cada nó é conectado na rede *blockchain* e pode enviar e receber transações.
2. Rede (*network*): Organizações e pessoas podem manter sistemas de computadores chamados nós e cada um desses nós executam o software de *blockchain* para comunicar uns com os outros, através de uma rede *blockchain*.
3. Contratos Inteligentes (*Smart Contracts*): Transações ou contratos convertidos em código para serem executados em uma *blockchain*.
4. Proposta de Transação (*submit transaction*): Usuários enviam transações para uma *blockchain* através de nós em uma rede, que por sua vez disseminam a transação por todos os outros nós da rede.
5. Validação de Transação (*transaction validation*): Nós em uma rede *blockchain* processam e validam cada transação. A rede ignora transações inválidas.
6. bloco (block): Os nós coletam e agrupam transações válidas em um pacote, chamado bloco. Cada bloco deve seguir um conjunto de regras pré-determinadas para ser considerado válido.

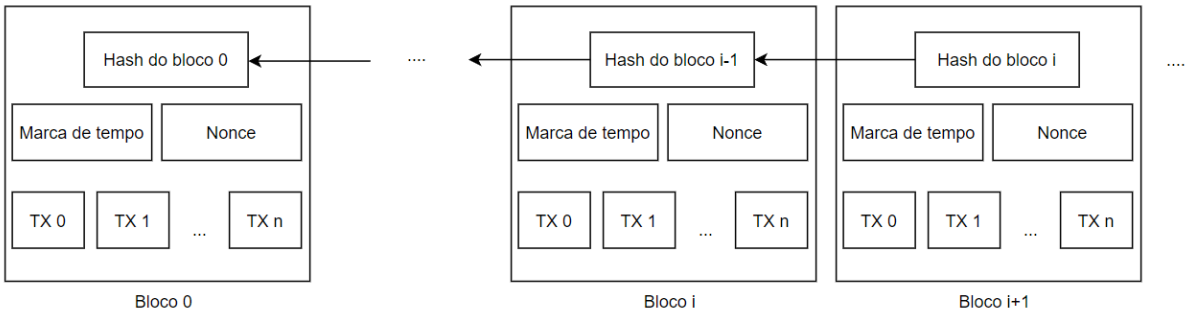
[4] Uma função *hash* converte uma entrada de tamanho variável em uma saída de tamanho fixo. Seu uso pode garantir diversos benefícios à cibersegurança tais como autenticidade e geração de números aleatórios. (Sobti; Geetha, 2012)

7. *blockchain*: Cada novo bloco válido referência o bloco válido imediatamente anterior e se conecta a ele, dessa forma é criada uma *blockchain* (“corrente de blocos”).
8. Consenso (*consensus*): É o processo de garantir que cada nó concorda com a *blockchain*.

2.3 Arquitetura

Uma *blockchain* pode ser descrita como um livro de registros onde cada transação é registrada em uma cadeia de blocos e cada bloco, após ser registrado, não pode ser apagado, por isso é chamado de *Ledger*. Cada bloco novo, aponta para o bloco imediatamente anterior, chamado bloco pai. O primeiro bloco de uma *blockchain* é chamado bloco de gênesis. (Zheng et al., 2018). A figura 2 demonstra um exemplo de *blockchain*.

Figura 2 - Exemplo de Arquitetura *Blockchain*



Fonte: Própria 2023.

Como cada bloco aponta para o bloco anterior, qualquer mudança que houvesse seria prontamente notada na *blockchain.* A marca de tempo, especifica o segundo da criação do bloco, tendo como marco inicial a data 01/01/1970 00:00 UTC e por sua vez o *Nonce* representa um campo de 4 bytes, normalmente iniciado em 0, que é incrementado para cada cálculo de *hash*. (Zheng et al., 2018)

Segundo Miers et al. (2019), todos os nós possuem cópias dos blocos que compõem a *blockchain* e cada registro contido nele é verificado individualmente conforme as regras estabelecidas. Após verificar um bloco, o nó envia uma cópia desse bloco para outros nós da rede, isso se repete até que todos tenham a nova informação. Contudo, como esse processo é feito sem uma coordenação central, cada nó pode ter uma visão diferente do estado do *blockchain* em um determinado momento, sendo assim, a *blockchain* deve utilizar algum mecanismo de consenso. Os mecanismos de consenso possibilitam a construção de um ambiente robusto a tentativas de violação, onde cada transação é verificada por uma gama de participantes não necessariamente confiáveis.

Alguns dos principais mecanismos de consenso, são Proof-of-Work (PoW), Proof of Authority (PoA) e Proof-of-Stake (PoS). (Miers et al., 2019).

1. Proof-of-Work (PoW): Todos os nós da rede calculam um *hash* específico utilizando diferentes *nonces*, ao encontrar esse *hash* o nó informa aos demais e os outros devem confirmar esse *hash* e então validar a transação. Essa é a origem do termo mineração. (Miers et al., 2019).
2. Proof of Authority (PoA): Comumente utilizado em redes *blockchain* privadas. Um grupo de autoridades é responsável por validar as transações. Para criar ou enviar um bloco válido, o responsável depende da aprovação de todas as autoridades envolvidas na rede. (Miers et al., 2019). Os *Smart Contracts* podem ser utilizados para validar os dados armazenados nos blocos, visto que permitem implementar as regras que definirão o comportamento da *blockchain*. (Freitas, 2023)
3. Proof-of-Stake (PoS): Nesse mecanismo, os mineradores que tenham mais participação na rede, possuem maior vantagem na mineração. é possível que seja utilizado formas randomização para tal. (Miers et al., 2019)

2.4 *Smart Contracts*

A integração entre a tecnologia do *blockchain* e dos *smart contracts*, permitiu uma grande flexibilidade para desenvolver e delinear problemas do mundo real gastando menos recursos e menos tempo do que utilizando outro tipo de sistema. Um *smart contract* permite executar um código sem que seja preciso a participação de terceiros. Sua estrutura contém valor, endereço, funções e estado. (Mohanta, 2018)

Um *smart contract* é um protocolo de transação computadorizada que executa os termos de um contrato (Szabo, 1997). Em *blockchain*, um *smart contract* é um fragmento de código que consegue ser executado por mineradores automaticamente. (Zheng et al., 2018)

Um *smart contract* escrito corretamente, deve descrever todos os possíveis resultados de um contrato. Ele é acionado por mensagens/transações enviadas para seu endereço. Além disso, um *smart contract* é determinístico, o mesmo input sempre irá produzir o mesmo output. Ressalta-se ainda que um *smart contract* pertence a uma *blockchain* e seu código pode ser inspecionado por todos os participantes na rede. Eles agem como agentes autônomos cujo comportamento é completamente previsível, por isso eles podem ser confiáveis para impulsionar qualquer lógica na cadeia que possa ser expressa como uma função de entrada de dados. (Christidis, 2016)

Os contratos agem como um cofre criptográfico que contém determinado ativo ou código e apenas se destrava quando condições específicas forem alcançadas. (Buterin, 2014) O conceito dos *smart contracts* pode ser aplicado a muitas aplicações e podem oferecer diversos benefícios, dentre os quais, destacam-se:

1. velocidade e atualizações em tempo real;
2. precisão;
3. baixo risco de execução;
4. custo baixo; e
5. potenciais novos modelos de negócios.

Além disso, segundo Mohanta (2018) descreve possíveis casos de uso para os *smart contracts*. Nos casos de uso citados, Mohanta destaca uso em cadeias de suprimento, direitos autorais, sistema financeiro, seguros, sistemas de saúde e internet das coisas (IoT). Neste último caso de uso, ressalta-se a capacidade de monitoramento inteligente através de *blockchains* compostas de sistemas de IoT.

2.5 *Blockchain* Permissionada

Em *blockchains* não permissionadas, isto é, públicas, qualquer um pode participar sem nenhum tipo de identificação de entrada. As *blockchains* públicas, tipicamente envolvem operações com criptomoedas e normalmente utilizam o protocolo de consenso PoW. Por outro lado, as *blockchains* permissionadas funcionam em um grupo conhecido de participantes. Nesse caso, elas conseguem prover um modo seguro de integração entre entidades que compartilham os mesmos objetivos. (Androulaki et al., 2018)

Corroborando com o conceito de *blockchain* permissionada, Kakavand et al. (2017) define que em blockchain privadas, cada participante é conhecido dos outros participantes e novos possíveis participantes devem ser convidados.

Existem alguns exemplos de plataformas de *blockchain* que podem ser utilizadas para implementar uma *blockchain* permissionada, dentre as quais, pode-se mencionar o próprio Ethereum. Essa *blockchain* permite construir e executar aplicações descentralizadas que permitem interação direta entre participantes da rede. Essa plataforma foi construída para garantir altos níveis de segurança contra os ataques de negação de serviço. Todos os nós de uma rede que executa uma *blockchain* Ethereum deve executar uma Máquina Virtual Ethereum de forma a manter o consenso na rede. (Kakavand et al. 2017)

Ressalta-se também a plataforma H*yperLedger Fabric*, desenvolvida pela fundação Linux que é uma plataforma blockchain de código aberto utilizada em diversas áreas que

introduz uma arquitetura de rede que visa a resiliência, flexibilidade, escalabilidade e confidencialidade. Além disso, possui uma arquitetura modular e configurável e suporta *smart contracts* gerados em linguagens de programação de propósito geral além de linguagens de domínio específico. (Linux Foundation, 2019)

Sendo uma *blockchain* permissionada, a H*yperLedger Fabric* permite haver confidencialidade por meio de sua arquitetura de canal e recursos de dados. Nos seus canais, participantes da rede estabelecem uma sub rede onde cada membro tem visibilidade para um conjunto particular de transações. Dessa forma, apenas os nós que participam de um canal têm acesso aos *smart contracts* e dados transacionados, preservando assim a privacidade e confidencialidade de ambos. (Linux Foundation, 2019)

A *HyperLedger Fabric* permite encriptar todas as transações e prover acesso apenas a determinadas partes interessadas. Também permite esconder a identidade, padrões de transação e termos dos contratos de terceiros não autorizados. Além disso, essa plataforma permite que dispositivos de baixo poder computacional, como dispositivos IoT (*Internet of Things*), sejam empregados em sua rede (Freire, 2022).

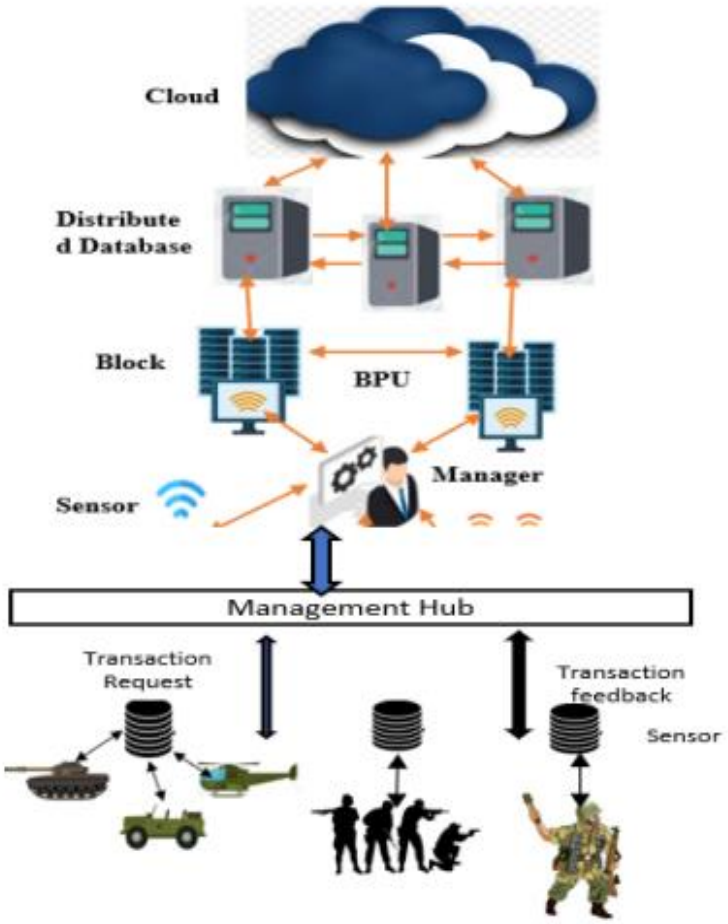
2.6 Trabalhos Relacionados

Akter et al. (2019), apresenta um canal de comunicação seguro para Comando Controle Comunicação e Inteligência (C3I) baseado em *blockchain* para sistemas militares. Em seu trabalho, os autores ressaltam que o sucesso no campo de batalha depende de uma rede de comunicações adequada e para isso, propõe um modelo de rede *blockchain* descentralizada geograficamente distribuída.

No modelo proposto, cada transação foi aprovada por uma colaboração inteligente de diversas entidades que, graças às propriedades oferecidas pela tecnologia *blockchain*, nenhuma solicitação que apresenta alguma vulnerabilidade será validada por todos os nós da rede. Por outro lado, as transações válidas são armazenadas e oferecidas ao requerente.

A arquitetura proposta no referido trabalho é composta de 5 itens: sensores, hub de gerenciamento, gerenciador, plataforma de *blockchain* e sistema de armazenamento.

Figura 3 - Ilustração de um sistema militar habilitado para *blockchain*



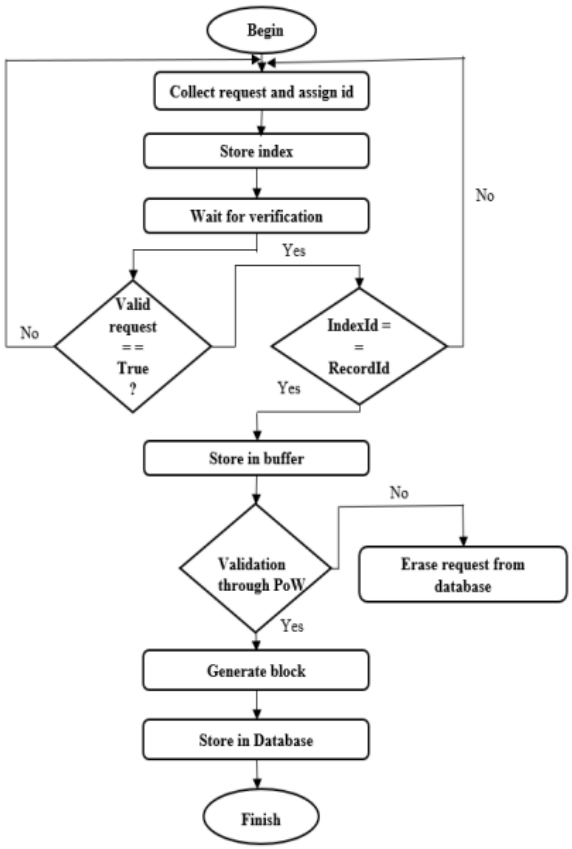
Fonte: Highly Secured C3I Communication Network Based on Blockchain Technology for Military System (Akter et al., 2019)

No artigo citado, Akter et al. (2019) menciona que manter a segurança e as transações ocorrendo em de forma rápida é crucial para um sistema de tempo crítico. Normalmente os sensores coletam solicitações de transação e transferem para o gerenciador. Inicialmente os próprios sensores fazem um pré-processamento das informações adquiridas e a aquisição das informações é feita pelo gerenciador. Toda transação é verificada pela chave privada do gerenciador, que assina digitalmente cada uma delas. Ressalta-se ainda o diagrama de fluxo para o referido sistema proposto.

Para identificar uma transação válida, um id particular e único é designado e armazenado. O mecanismo de verificação é executado, caso o resultado seja negativo, o processo é reiniciado e caso o resultado seja positivo, o id é comparado com a lista de ids armazenados. Se bem-sucedida a comparação, o respectivo id vai para o mecanismo de

consenso PoW que gerara o bloco correspondente e o armazena na base de dados. Caso o mecanismo de consenso indique falha, a solicitação é apagada.

Figura 4 - Diagrama de fluxo do sistema militar habilitado para *blockchain*



Fonte: Highly Secured C3I Communication Network Based on Blockchain Technology for Military System (Akter et al., 2019)

Como mencionado, o sucesso do sistema proposto depende do tempo de execução. a fim de garantir determinar os tempos necessários para cada transação, foram feitos experimentos baseados em simulações, onde o Akter et al. (2019), chegou aos seguintes resultados:

Figura 5 - tempos medidos para execução do sistema militar habilitado para *blockchain*

Parameters description	Values
Number of dispached transaction	0.2856 (micro sec)
Transaction arrival rate	3.55 (micro sec)
Number of transaction	2100 (1/ block)
Mining rate	0.0016 (micro sec))
Number of total transaction	306720
Simulation Time	86400 sec

Fonte: Highly Secured C3I Communication Network Based on Blockchain Technology for Military System (Akter et al., 2019)

Os tempos apresentados por Akter et Al. (2019) constatam que a tecnologia *blockchain* proporciona um baixo *overhead* e dessa forma garante que possa ser utilizada em sistemas que demandam baixa latência mantendo a QoS[5] (*Quality of Service*) adequada.

Como exemplo de outro trabalho relacionado, pode-se mencionar um Sistema de Monitoramento Marítimo proposto por Freire et al. (2021). No artigo mencionado, o autor propõe um sistema baseado em *blockchain* para realizar monitoramento marítimo, implementação que garante integridade, autenticidade e disponibilidade. Esse sistema usa dispositivos como boias e veículos aéreos não tripulados como clientes de uma rede *blockchain* possuidores de suas próprias credenciais.

Os clientes da rede proposta seriam equipados com receptores AIS de baixo custo implementados usando um microcomputador embarcado Raspberry Pi e um dongle RDS[6] (Rádio Definido por Software). Assim, os clientes poderiam transferir dados da área de monitoramento em transações de *blockchain*. A aplicação do cliente consiste em módulos que utilizam Python capazes de requerer funções de *smart contracts* usadas no Fabric Python SDK (*Software Development Kit*).

No trabalho de Freire et al. (2021), para simular uma operação em ambiente real, foram enviadas a estrutura proposta 1500 entradas AIS. Esse experimento permitiu verificar que apesar do alto consumo de CPU e memória, quando comparado com tráfego em *Secure Shell* (SSH), apresentada parâmetros aceitáveis de QoS justificados pelo aprimoramento geral da segurança cibernética do sistema.

[5] Mecanismo usado para garantir desempenho desejado de aplicativos essenciais. (Garret, 2018)
[6] Radio cuja modulação de formas de onda do canal é definida em software. (Mitola, 2000)

3 METODOLOGIA

Este trabalho de conclusão de curso, objetiva verificar a aplicabilidade de *blockchains* permissionadas nas operações navais, implementando um exemplo de *blockchain* a fim de demonstrar as funcionalidades pesquisadas. A partir das comprovações obtidas, serão apresentadas possíveis aplicações da tecnologia blockchain nas diversas áreas de atuação da Marinha do Brasil.

3.1 Classificação da Pesquisa

Uma pesquisa exploratória consegue tornar um problema mais explícito ou construir uma ou mais hipóteses. Seu maior objetivo é aprimorar ideias ou descobrir intuições, possui um planejamento flexível e possibilita considerar vários aspectos do objeto do estudo. (Gil, 2016)

Pesquisas explicativas têm preocupação em identificar os fatores que determinam ou contribuem para que determinados fenômenos ocorram. O conhecimento científico está assentado nos resultados oferecidos pelos estudos explicativos. Pesquisas também podem ser classificadas quanto às técnicas utilizadas, dentre as técnicas, destacam-se pesquisa bibliográfica que é desenvolvida com base em materiais já elaborados. (Gil, 2010)

3.2 Limitações do Método

A implementação proposta de uma *blockchain* foi desenvolvida em ambiente de desenvolvimento controlado, onde não foi intenção entrar nos menores níveis de detalhamentos da rede, mas sim, demonstrar de forma prática e abrangente os conceitos apresentados teoricamente. Para desenvolver a proposta de *blockchain*, foi utilizada a linguagem de programação Python.

A referida linguagem foi escolhida pois é dinâmica e orientada a objetos e pode ser utilizada para desenvolver qualquer tipo de aplicação. Além disso, a linguagem tem uma sintaxe clara e simples e seu uso pode ser associado a grandes ganhos de produtividade e com a produção de aplicações de alta qualidade e de fácil manutenção. ressalta-se ainda que o Python, possui algumas características dentre as quais, destacam-se: (Coelho, 2007)

1. Multiplataforma: é possível que seja instalado em qualquer tipo de computadores, desde PDAs (*personal digital assistant*) até supercomputadores.

2. Portabilidade: Aplicações feitas em Python podem ser distribuídas para plataformas distintas daquela em que foi desenvolvida.
3. Software Livre: A linguagem é gratuita.
4. Extensibilidade: O Python pode ser expandido incorporando em seu código módulos escritos em outras linguagens de programação.
5. Flexibilidade: O Python possui módulos, inclusive em sua distribuição básica, para interagir com diversos tipos de aplicações existentes.
6. Operação com Arquivos: Permite manipular arquivos, possibilitando ler e escrever arquivos de forma simples.
7. Uso interativo: Permite testar comandos antes que tais comandos sejam incluídos em aplicações mais complexas.

No desenvolvimento, uma das implementações propostas consiste em uma *blockchain* na qual cada bloco guarda leituras feitas por um sensor. No caso apresentado foi utilizado um CLP Arduino Uno juntamente com um sensor de temperatura TMP36 uma protoboard e fios para conexão.

Na outra implementação, foi codificada em Python uma *blockchain* que tem por finalidade permitir o uso de uma rede blockchain para troca de mensagens entre usuários previamente cadastrados. A proposta visa estabelecer uma rede de Comando e Controle entre navios e distritos navais para apoiar as operações militares. A limitação desta implementação é a efetiva conexão de nós a rede proposta. A *blockchain* não foi desenvolvida em plataformas especificas como a *HyperLedger Fabric,* citada no trabalho, portanto tem intuito apenas exemplificativo e de demonstrar como seria a aplicação de uma ferramenta efetivamente funcional no contexto proposto.

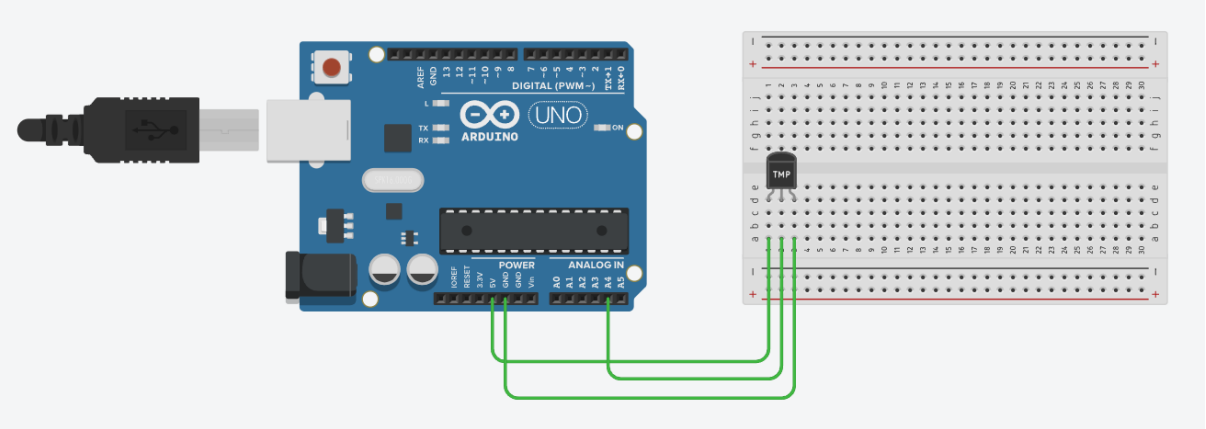
3.3 Descrição do sistema

Tendo em vista as vantagens da utilização da tecnologia *blockchain* e utilizando os conceitos explanados nas seções anteriores deste trabalho, foram desenvolvidas duas aplicações diferentes que servem respectivamente para efetuar a leitura de saídas de sensores e estabelecer uma rede de comunicação baseada em *blockchain.*

3.4 Leitura de sensores

Na *blockchain* utilizada para verificar a leitura de sensores, foi utilizada a lógica de programação semelhante, contudo, em cada bloco ao invés de ser gravada uma mensagem, é gravada uma leitura de um sensor. Para realizar o modelo de implementação, foi utilizado uma simulação de uma placa Arduino Uno, uma protoboard e um sensor de temperatura TMP36, conectados conforme figura 6 abaixo. O circuito implementado para leitura de temperatura é propositalmente simples pois tem a finalidade apenas de demonstrar a viabilidade da solução proposta. Além disso, ressalta-se que a implementação feita pode ser usada para leituras de saídas de quaisquer tipos de sensores.

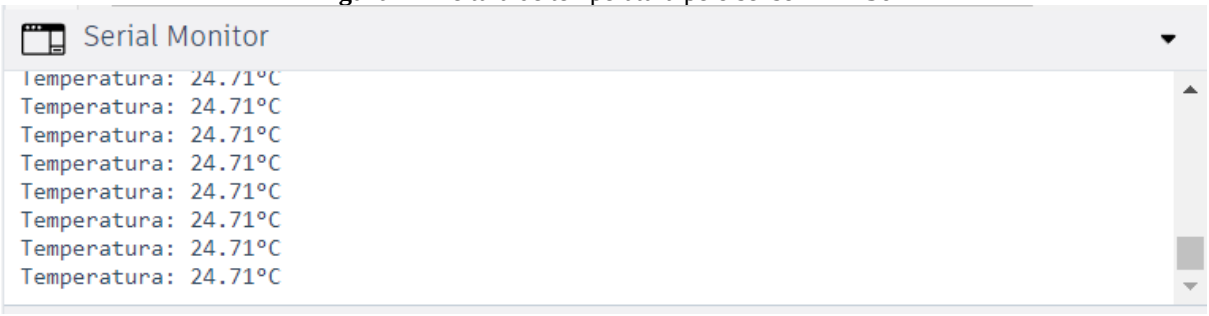
Figura 6 - Circuito para leitura de temperaturas



Fonte: Própria 2023.

O sensor utilizado possui 3 conexões que são utilizadas para: fornecer, a partir do Arduino Uno, os 5V necessários ao seu funcionamento; fornecer, a partir do sensor, para o CLP uma tensão de saída proporcional à temperatura medida e um fio terra. Em ambiente de testes, foram realizadas medições de temperatura a cada 1 segundo e o resultado foi obtido conforme figura 7 abaixo. Em virtude da periodicidade das leituras, os valores resultantes foram iguais, contudo, ressalta-se que a periodicidade de leitura pode ser alterada de acordo com a vontade do usuário.

Figura 7 - Leitura de temperatura pelo sensor TMP36



Fonte: Própria 2023

O sistema de *blockchain* utilizado para compartilhar dados de leituras de sensores proposto tem as seguintes premissas:

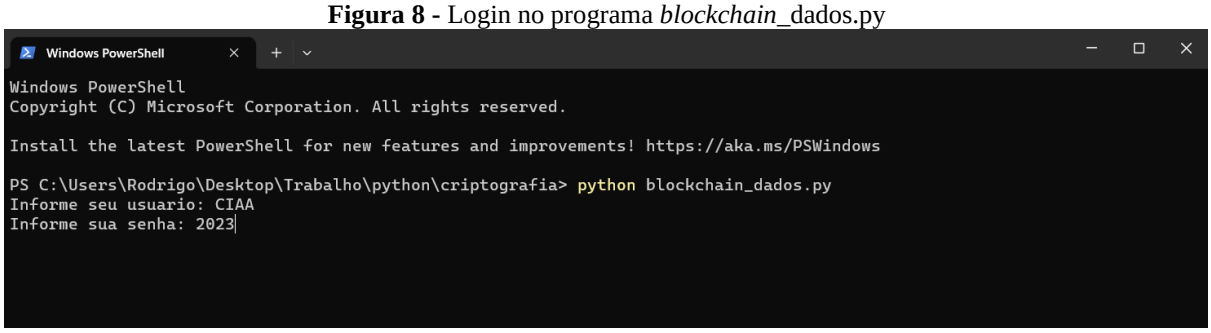
1. A periodicidade das leituras de sensores deve ser padronizada de sensores devem ser estabelecidas conforme necessidade do equipamento e do usuário.
2. As leituras feitas pelo sensor são gravadas em um arquivo chamado dados.csv. Nesse arquivo, é gravado o data-hora da leitura e seu valor. Cada linha do arquivo corresponde a uma leitura
3. Para tornar o este trabalho mais didático, foram incluídas apenas 10 leituras no arquivo dados.csv e seus valores foram alterados para que seja perceptível ao leitor verificar a funcionalidade proposta.
4. Para tornar o código mais compacto, e tendo em vista que o desenvolvimento mais complexo já foi utilizado na *blockchain* anterior, para a *blockchain* de leitura de sensores, foi desenvolvido apenas um programa. Nessa implementação, os usuários e senhas foram registrados no próprio código.
5. A aplicação da *blockchain* possui uma classe Bloco e uma classe *Blockchain*, semelhantes a aplicação anterior. Possui uma rotina para ler as linhas de dados.csv e adicionar cada linha a *blockchain* criada.

Essa *blockchain* foi implementada em um arquivo nomeado blockchain_dados.py[7]. Nesse arquivo, foram estabelecidas as classes bloco e *blockchain*, que determinam a estrutura que cada bloco e cada *blockchain* devem ter respectivamente. Além disso, cada classe possui rotinas necessárias para sua utilização. Especificamente nessa implementação, as credenciais de acesso foram salvas no próprio programa, visto que a premissa utilizada para leitura de

[7] Os códigos desenvolvidos encontram-se disponível nos Anexos deste trabalho.

sensores é que essa *blockchain* seja executada internamente em um único meio, portanto os usuários autorizados a acessar as leituras foram previamente cadastrados.

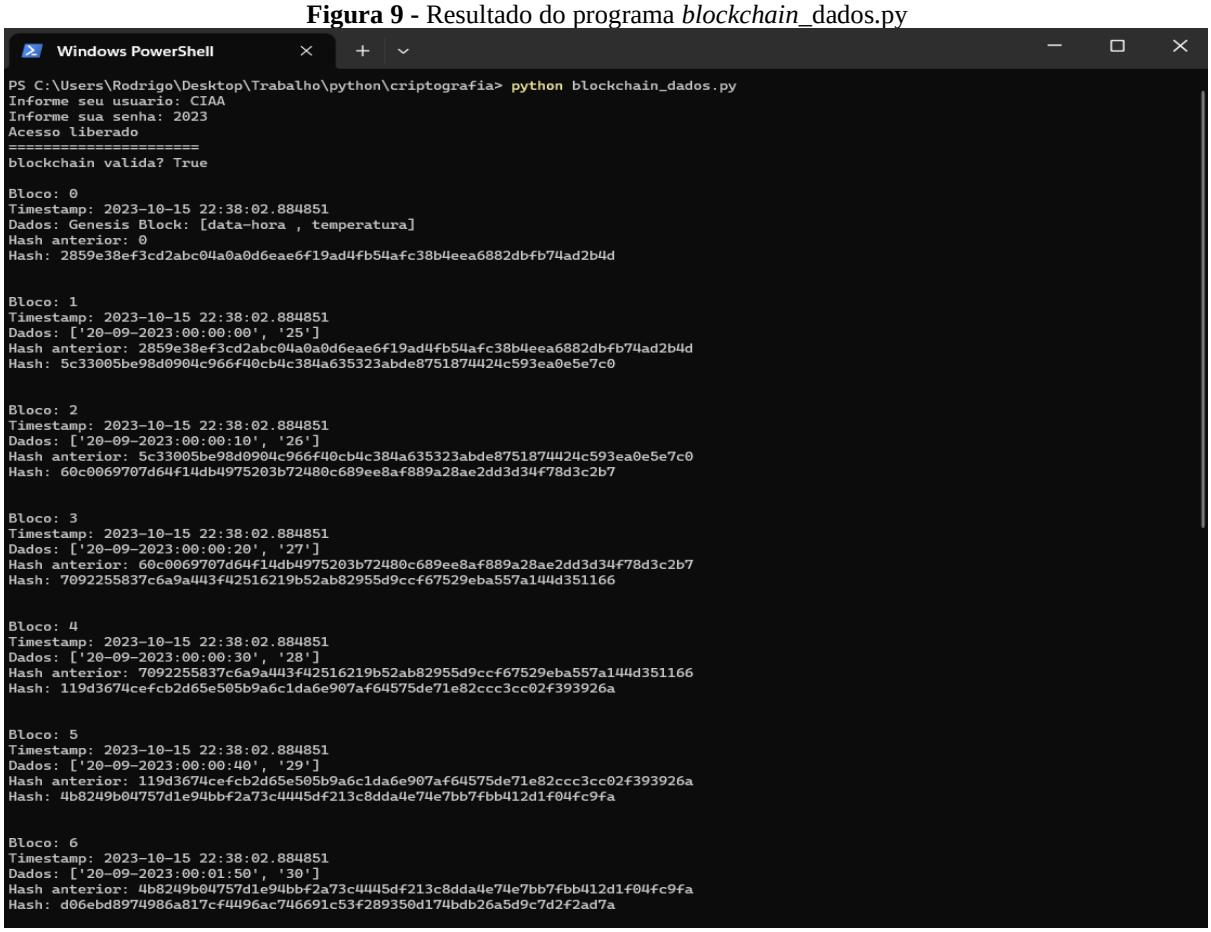
Figura 8 - Login no programa *blockchain*_dados.py



Fonte: Própria 2023.

Para possibilitar a execução do que foi proposto, foi criado um laço de repetição que conta as linhas do arquivo dados.csv e para cada linha do arquivo, adiciona a um novo bloco da *blockchain* o conteúdo da linha. Antes da exibição das leituras do sensor, a aplicação exibe para o usuário se a *blockchain* é válida ou não. Para verificar a validade da *blockchain*, é realizada uma comparação entre os *hashes* de todos os blocos i para todos os blocos i-1, semelhante ao explanado na parte teórica do presente trabalho.

Figura 9 - Resultado do programa *blockchain*_dados.py



Fonte: Própria 2023.

Figura 10 - Continuação do programa *blockchain*_dados.py

Fonte: Própria 2023.

3.5 Análise de Segurança

Devido ao fato de ser uma *blockchain* permissionada, com estrutura descentralizada, utilização de credenciais de acesso e logica de informações em cadeia, é possível fazer um comparativo da leitura de sensores utilizando a tecnologia *blockchain* com a leitura simples de sensores.

Tabela 1 - Comparação de segurança entre leitura simples e leitura utilizando *blockchain*.

Aspecto de Segurança	Leitura utilizando sensores simples	Leitura de sensores com *blockchain*
Integridade dos dados	Vulnerável a adulteração	Altamente resistente a adulteração
Prova de Autenticidade dos dados	Dificuldade de verificar a origem dos dados	Fornecimento de prova de origem dos dados
Imutabilidade dos dados	Dados podem ser modificados sem que o operador perceba	Dados são imutáveis e registrados permanentemente em uma *blockchain*
Resistencia a ataques cibernéticos	Vulnerável a ataques de invasões e falsificações	Resiliente a ataques devido a natureza descentralizada

Recuperação de dados	Dados perdidos dificilmente serão recuperados	Facilidade em recuperar dados devido ao registro permanente na *blockchain*

Fonte: Própria 2023.

3.6 Comunicação baseada em *blockchain*

Na *blockchain* utilizada para agregar uma camada de segurança ao C3I através de comunicação baseada em *blockchain*, foram desenvolvidos 5 arquivos diferentes e que trabalham de forma conjunta para realizar cada necessidade do sistema proposto. As premissas utilizadas para realizar a implementação de tal sistema foram:

1. A criação de usuário e senha, não poderá ser feita pelo usuário. Tal criação deverá ser feita por Organização Militar (OM) específica, com capacidade técnica e envolvida na segurança operacional da missão. A distribuição dessas senhas fica a cargo da OM citada.
2. Cada usuário será identificado por um trigrama para cada missão.
3. Cada mensagem enviada na rede comporta um bloco da *blockchain*.
4. Na implementação no terminal, os participantes da rede terão apenas 4 alternativas de ações, a saber: digitar sua mensagem, ler os conteúdos da conversa, verificar a validade da *blockchain* e sair do chat.
5. Os *hashes* foram omitidos do chat para melhorar a experiência do usuário, contudo, existe um comando específico para verificar a validade da *blockchain*.

Para uso no terminal gráfico, os 5 programas desenvolvidos são:

1. blockchain.py

a. Esse programa define as classes Bloco e *Blockchain*. Nesse arquivo, são definidas as características que cada bloco e cada *blockchain* deve possuir. Além disso, são definidas funções específicas para cada classe mencionada.

2. criar_usuario.py

a. Esse programa, segundo a premissa utilizada, deve ser executado por uma OM específica.

b. Permite a criação de usuários e suas senhas.

c. A execução desse programa, registra em um arquivo separado, no caso foi utilizado um arquivo usuarios.txt, o usuário e o *hash* de sua senha.

d. O armazenamento do *hash*, garante que mesmo se o arquivo usuarios.txt for comprometido, a senha não será.

3. valida_login.py

a. Esse programa realiza a verificação se usuário e senha informados pelo usuário, quando este tenta efetuar login correspondem ao usuário e senha armazenados no arquivo usuarios.txt

b. A senha é informada pelo usuário, em texto claro, dessa forma, para verificar a validade da senha, o valor fornecido pelo usuário passa por uma função *hash*. Caso o *hash* calculado seja igual ao *hash* armazenado, o sistema libera o acesso.

4. cria_blockchain.py

a. Esse programa importa o programa blockchain.py para utilizar as estruturas previamente definidas.

b. São definidas funções que definem mensagens a serem impressas na tela para o usuário e uma função que define uma *blockchain* que será usada para a comunicação proposta.

5. executavel.py

a. Esse programa importa os programas valida_login.py e cria_blockchain.py

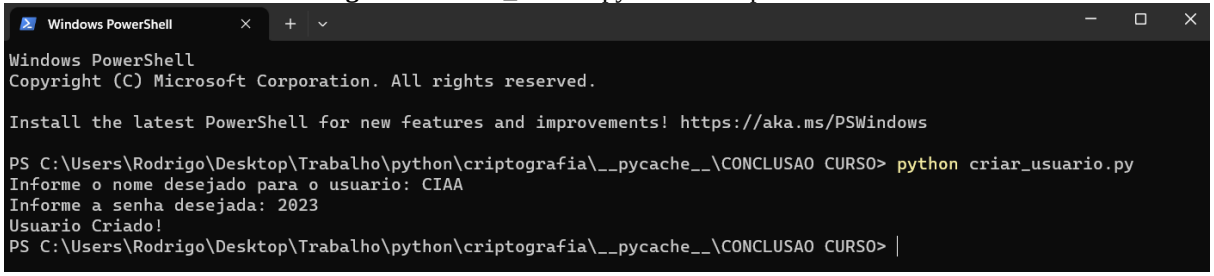
b. Esse programa é simples pois toda estrutura lógica já foi desenvolvida nos programas anteriormente citados.

c. Solicita que o usuário informe seu usuário e senha, para verificar se estão corretos, chama o programa valida_login.py.

d. Caso o login seja validado, chama o programa cria_blockchain.py e assim a *blockchain* consegue ser estabelecida.

A aplicação desenvolvida, quando executada via terminal foi implementada e pode ser utilizada conforme figura abaixo. Na figura 11, o programa criar_usuario.py é executado, então o usuário informa que deseja criar um perfil ‘CIAA’, cuja senha será 2023.

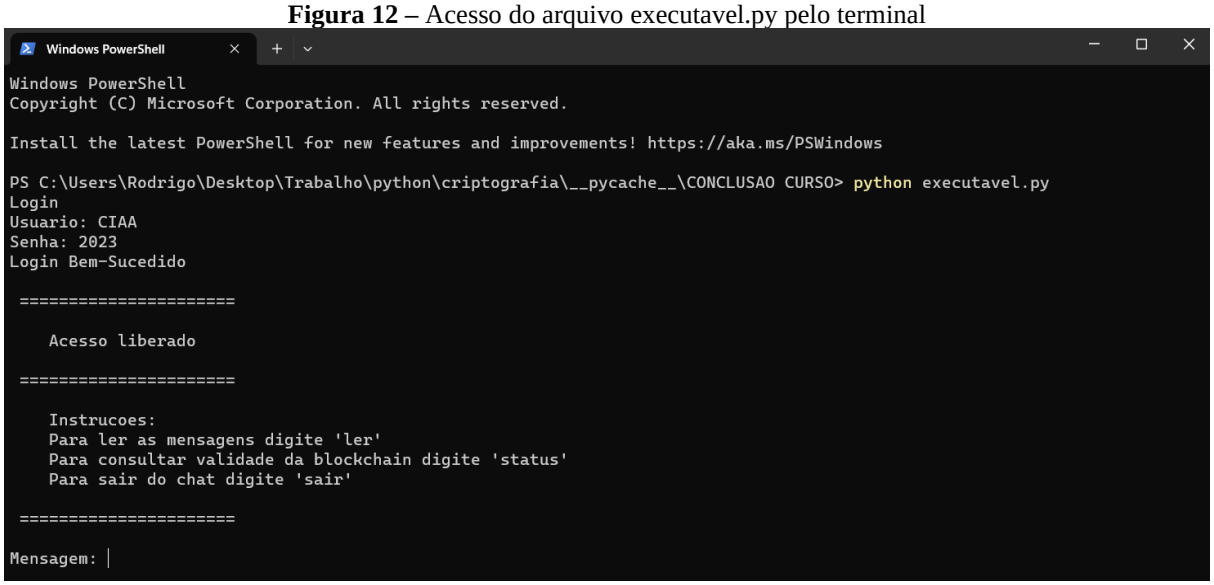
Figura 11 - criar_usuario.py executado pelo terminal



Fonte: Própria 2023.

Na figura 12, é executado o programa executavel.py. Quando executado, o programa solicita que seja informado nome de usuário e senha, caso sejam fornecidas as credenciais corretas, o programa dá acesso a área de mensagens. Além disso, na área logada do programa, são exibidas mensagens de instruções para utilização. As instruções permitem que o usuário leia o que já foi inserido na *blockchain*, permite que o usuário verifique a validade da *blockchain* e permite que o usuário feche o chat.

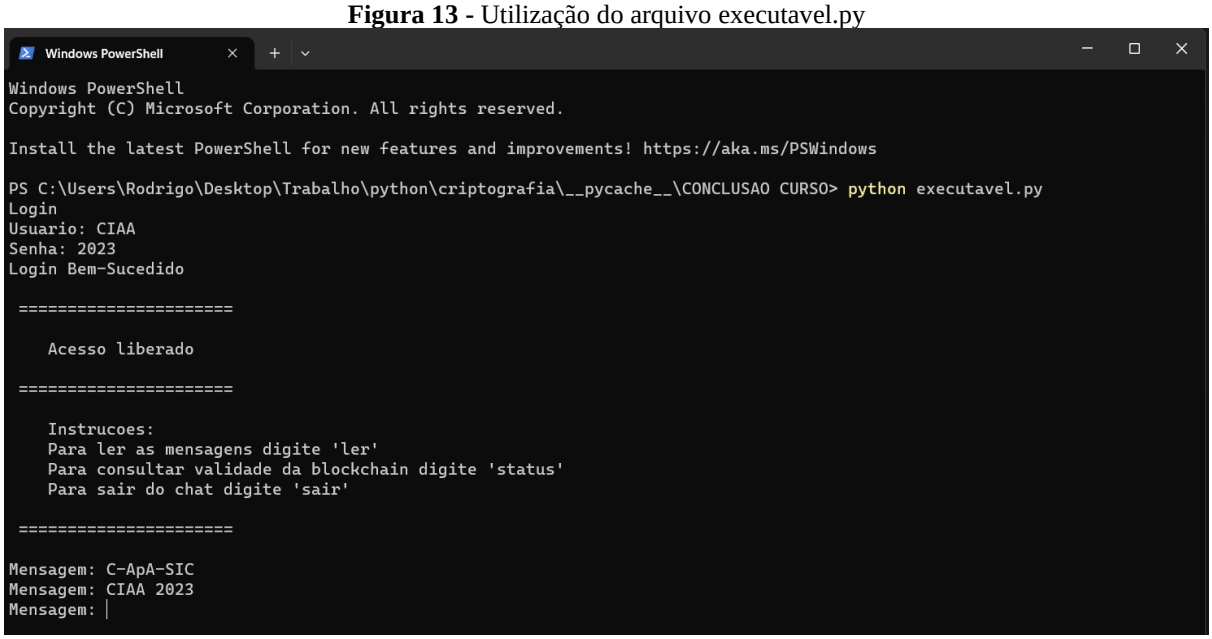
Figura 12 – Acesso do arquivo executavel.py pelo terminal



Fonte: Própria 2023.

A figura 13 demonstra a utilização da aplicação. Nessa figura, o usuário digitou duas mensagens que são adicionadas a blocos da *blockchain*.

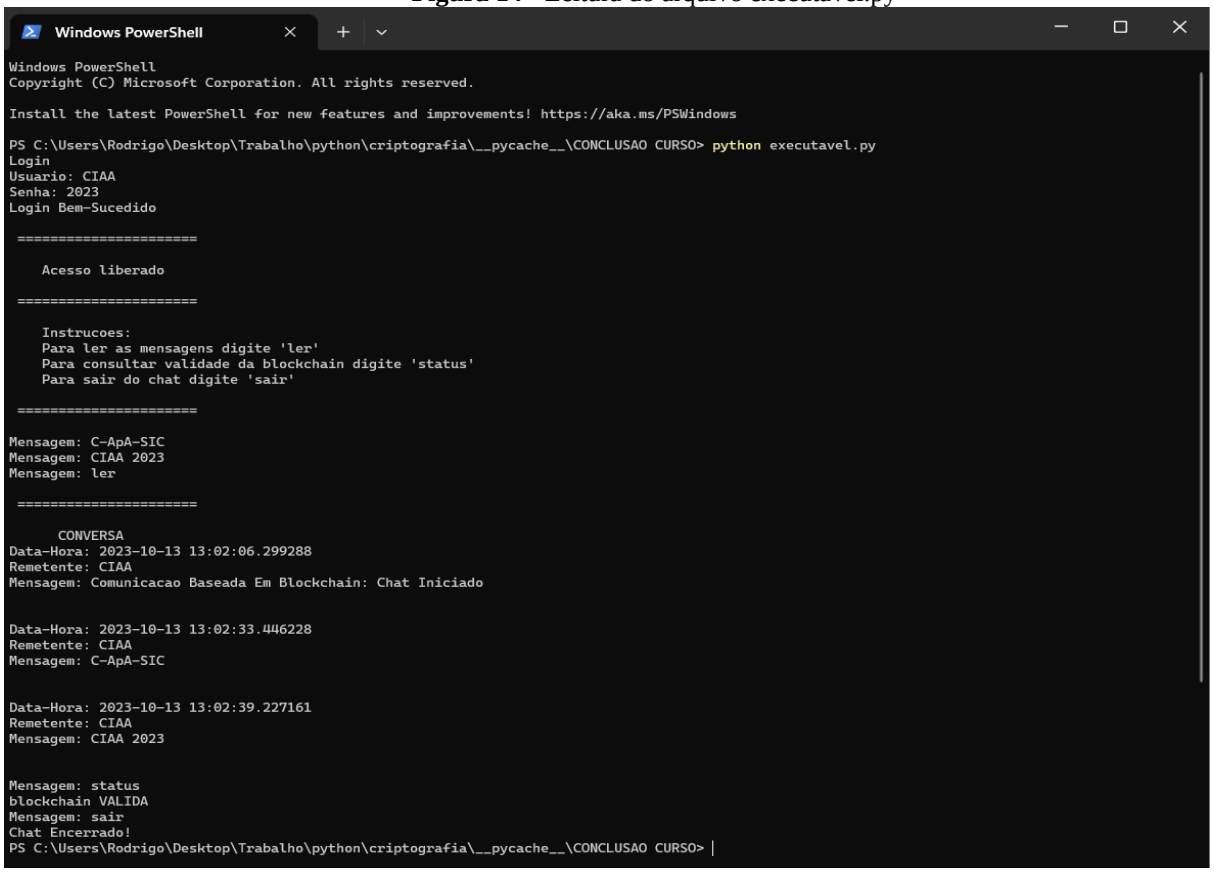
Figura 13 - Utilização do arquivo executavel.py



Fonte: Própria 2023.

Na figura 14, o usuário digitou a instrução ‘ler’ para verificar as mensagens digitadas, em seguida verificou a validade da *blockchain* e após isso digitou a instrução ‘sair’ para encerrar o chat.

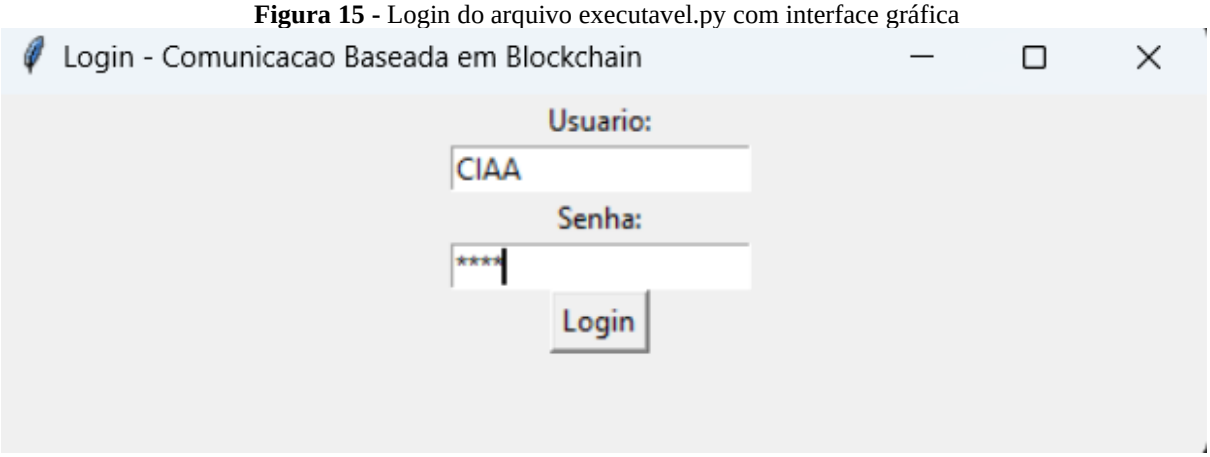
Figura 14 - Leitura do arquivo executavel.py



Fonte: Própria 2023.

Para facilitar a experiência do usuário, foi desenvolvido um terminal gráfico utilizando a mesma lógica de programação da aplicação executada no terminal. De fato, a implementação gráfica chama em seu código os mesmos arquivos usados na implementação do terminal. Para construir a interface gráfica, foi utilizada a biblioteca Tkinter. Esta biblioteca é adequada para uma grande variedade de aplicações e possui algumas vantagens relevantes, tais como ser estável e com poucas exceções, está disponível em qualquer lugar que o Python esteja instalado. (MOORE, 2018) O arquivo contendo a aplicação em terminal gráfico foi nomeado como terminal_grafico.py e em sua implementação, ele importa para utilização o arquivo valida_login.py. Quando executado, ele é exibido conforma figura 15 abaixo.

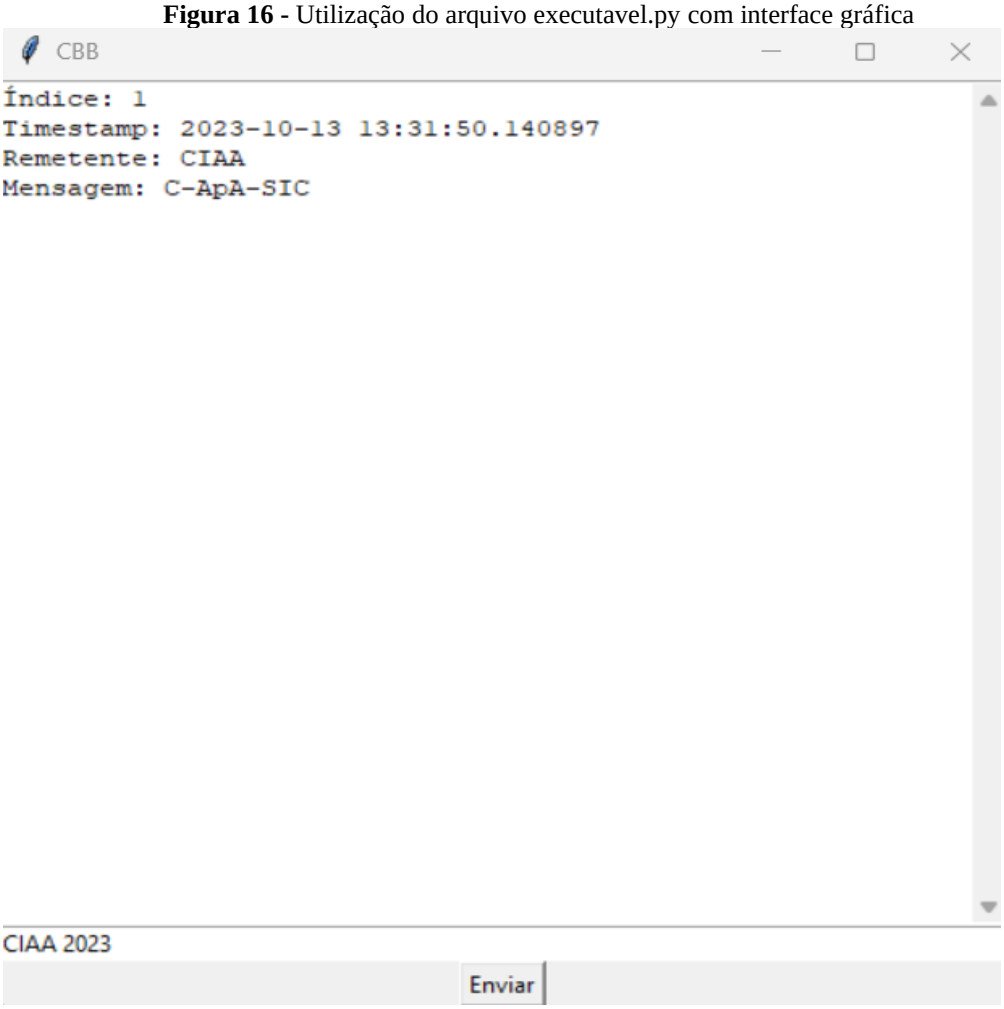
Figura 15 - Login do arquivo executavel.py com interface gráfica



Fonte: Própria 2023.

Após informadas as credenciais corretas, a tela a será exibida de conforme figura 16. Nesse ponto, o usuário pode digitar a mensagem que desejar enviar. A tela exibirá as informações de cada bloco, incluindo o índice do bloco e o data-hora que foi enviada a mensagem na rede.

Figura 16 - Utilização do arquivo executavel.py com interface gráfica



Fonte: Própria 2023.

4 DESCRIÇÃO E ANÁLISE DOS RESULTADOS

Diante dos programas desenvolvidos, foi possível verificar a existência dos conceitos mais relevantes descritos na parte teórica deste trabalho. Como exemplo disso, pode-se citar:

1. da arquitetura da *blockchain*:
    a. No exemplo do programa dados.py, propositalmente foram exibidos os *hashes* dos blocos, de forma que fosse possível notar a questão do apontamento do *hash* de um bloco i, para um bloco i-1.
    b. Em ambas *blockchains* apresentadas, é possível verificar a questão da marca temporal característica da arquitetura.
    c. Em cada bloco, corroborando com a estrutura descrita no referencial teórico, foram inseridos os dados, de forma a dar uma finalidade para o uso da rede.
2. da funcionalidade proposta:
    a. Analisando um meio naval como um sistema de sistemas. Onde devem ser consideradas a todo momento, diversas variáveis que se relacionam intimamente com a segurança do pessoal e do material. Além disso, tendo em vista as ameaças atuais decorrentes da cibersegurança, a implementação de uma *blockchain* para trafegar dados de sensores importantes traz consigo um grande implemento à segurança do meio.
    b. Como forma de garantir uma comunicação segura e robusta para sistemas de comando e controle, uma rede de *blockchain* para comunicação permite garantir a segurança desejada.

A implementação da *blockchain* dedicada a leitura de sensores, confere uma camada adicional de segurança que garante a integridade dos dados gerados. Cada leitura do sensor é encapsulada em um bloco imutável, dessa forma é impossível a adulteração dos registros. Essa característica é particularmente valiosa em setores críticos como em sistemas de controle de meios navais.

A *blockchain* empregada para comunicação segura, estabelece um ambiente seguro, que proporciona uma rede descentralizada para trocar informações relevantes e confidenciais. Isso elimina as vulnerabilidades associadas a intermediários e minimiza o risco de manipulação de dados sensíveis durante a transmissão. Diante disso, as implementações de *blockchain* se destacam na garantia de que a comunicação seja protegida e autenticada.

5 CONCLUSÃO

No decorrer do presente trabalho de conclusão, a tecnologia do *blockchain* foi explorada analisando seus princípios teóricos os aplicando na prática. Para demonstração prática do que fora explanado, foram desenvolvidas *blockchains* funcionais usando Python a fim de demonstrar a aplicação real dos conceitos abordados na parte teórica do estudo. Dessa forma, foi possível compreender de forma mais ampla e profunda os conceitos relevantes da tecnologia *blockchain* e suas possíveis aplicações no mundo real.

Tal tecnologia, foi utilizada como solução ao problema de ameaças cibernéticas no contexto da MB, sobretudo na operação dos meios navais. O problema em questão foi exemplificado, com o intuito de trazer ao leitor, a importância do assunto, através de explanações sobre o malware Stuxnet. Sendo citados, para isso, inclusive números de hosts infectados por esse programa malicioso.

No capítulo 2, foram explanados maiores detalhes sobre a tecnologia, dentre os quais destacam-se sua arquitetura, conceitos relevantes e quais objetivos a utilização do *blockchain* permite ao usuário alcançar. Dentre os conceitos relevantes, destacam-se os *smart contracts* e os mecanismos de consenso, que são ferramentas imprescindíveis para o correto compreendimento da tecnologia.

No desenvolvimento da *blockchain* própria em Python, foi possível constatar a importância do arcabouço teórico adquirido durante a pesquisa. Tais conhecimentos serviram como base para que fosse factível implementar de forma prática duas *blockchains* que, apesar de não funcionarem em uma plataforma específica, são funcionais. Ressalta-se ainda, que as *blockchains* desenvolvidas, podem servir como base para uma implementação real e funcional no contexto da MB.

Em resumo, este trabalho fornece uma visão abrangente da tecnologia *blockchain*, abordando tanto princípios teóricos quanto a implementação prática. Ao passo que a tecnologia *blockchain* continua a se expandir e evoluir, é relevante considerar as diversas maneiras pelas quais ela pode transformar os sistemas e processos existentes na Marinha do Brasil. Este trabalho é o início de uma jornada contínua da tecnologia *blockchain* no âmbito da MB.

Diante do exposto acima, é notório que os objetivos delimitados para o presente trabalho foram alcançados. O primeiro macro objetivo enumerar características da tecnologia blockchain mencionando e o segundo macro objetivo a abordagem pratica que comprove o que fora previamente dito na parte teórica. Atingindo os objetivos maiores, os objetivos menores como citar possíveis utilizações da tecnologia na MB também foram atingidos.

5.1 Considerações Finais

Finalizar este trabalho de conclusão de curso foi uma jornada que permitiu explorar um novo universo de conhecimento e desafios. Durante o processo de pesquisa e desenvolvimento prático, foi possível aprofundar-me nos conceitos da tecnologia *blockchain*. Por meio desta pesquisa, ficou constatado que a tecnologia *blockchain* é sem dúvida, uma força transformadora do cenário atual da segurança das informações e comunicações. Essa tecnologia proporciona boa segurança, transparência e eficiência em diversas áreas de atuação.

A pesquisa proporcionou a compreensão de que a pesquisa e o desenvolvimento nessa área ainda podem evoluir sobremaneira. À medida que novas aplicações e funcionalidades surgem, o uso de *blockchains* continuará a evoluir. Dito isso, sigo confiante que meu trabalho poderá ser útil para aplicações reais no contexto em propus.

5.2 Sugestões para Futuros Trabalhos

Tendo em vista os benefícios e potenciais utilizações para a tecnologia *blockchain* e a sua baixa implementação prática nas atividades das forças armadas, sugere-se dar continuidade a pesquisa da tecnologia e fomentar o ensino e desenvolvimento de aplicações baseadas em *blockchain*. Para tal, é possível mencionar algumas possibilidades de serem desenvolvidas em trabalhos futuros, dentre as quais destacam-se:

- Dar continuidade ao projeto apresentado neste trabalho para que seja possível integrar a lógica utilizada redes interligadas de IoT.
- Analisar a viabilidade do projeto desenvolvido nos diversos sensores utilizados a bordo de navios
- Analisar a viabilidade de uma comunicação baseada em *blockchain* entre estações dos navios para troca de mensagens relevantes
- Desenvolvimento de sistemas de controles de estoques baseados em *blockchain*.
- Desenvolvimento de novas possíveis soluções que utilizam a tecnologia *blockchain* para oferecer novas camadas de segurança às atividades desenvolvidas pela MB.

REFERÊNCIAS

AKTER, R.; BHARDWAJ, S.; LEE, J. M.; KIM, D.-S. **Highly Secured C3I Communication Network Based on Blockchain Technology for Military System** International Conference on Information and Communication Technology Convergence (ICTC); Jeju, Korea (South), 2019. p. 780-783. DOI: 10.1109/ICTC46691.2019.8939813.

ANDROULAKI, E. et al. **Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains**. Proceedings of the 13th EuroSys Conference, EuroSys 2018, n.1, 2018.

BRASIL. Plano Nacional de Defesa - PND. p. 81, 2020.

BRITANNICA.Science & Tech. In: **Destroyer**. [S. l.], 24 fev. 2023. Disponível em: https://www.britannica.com/technology/destroyer. Acesso em: 5 out. 2023.

BUTERIN, V. **A next-generation smart contract and decentralized application platform.** Etherum, n. Janeiro, p. 1–36, 2014.

CHRISTIDIS, K.; DEVETSIKIOTIS, M. **Blockchains and Smart Contracts for the Internet of Things.** IEEE Access, v. 4, p. 2292–2303, 2016.

COELHO, Flavio C. **Computação Cientifica com Python**. [S. l.: s. n.], 2007.

CORNELLA, Alessia; ZAMENGO, Linda; DELEPIERRE, Alexandre; CLEMENTZ, Georges. **Blockchain in defence: a breakthrought**. European Army Interoperability Center, [S. l.], p. 1-28, 1 set. 2020.

DEFESANET (ed.). **Fragata Classe Tamandaré – Tablets, estações de computadores e realidade aumentada: os novos navios da Marinha estão sendo construídos sem o uso de desenhos impressos em papel**. [S. l.], 25 jan. 2023. Disponível em: https://www.defesanet.com.br/cct/noticia/1046840/fragata-classe-tamandare-tablets-estacoes-de-computadores-e-realidade-aumentada-os-novos-navios-da-marinha-estao-sendo-construidos-sem-o-uso-de-desenhos-impressos-em-papel/. Acesso em: 25 set. 2023.

DJORDJEVIC-KAJAN, Slobodanka Djordjevic-Kajan; KAJAN, Ejub; MITROVIC, Dejan Mitrovic. Towards **Active C3I Systems**. Telsiks, [*s. l.*], Outubro 1997.

FALLIERE, Nicolas; MURCHU, Liam O; CHIEN, Eric. **Stuxnet Dossier**. Symantec, [S. l.], p. 1-68, 11 fev. 2011.

FREIRE, Warlley Paulo. **Cibersegurança Naval: Navegando em águas turbulentas na era da Guerra Cibernética**. Revista nas Sombras, [s. l.], 2023.

FREIRE, Warlley Paulo; MELO, Wilson S.; NASCIMENTO, Vinicius D. do; SA, Alan Oliveira de. **Blockchain-based Maritime Monitoring System**. International Workshop on Metrology for the sea, [S. l.], p. 1-6, 25 nov. 2021.

FREITAS, Glênio Descovi de. **Metodologia De Modelagem E Implementação De blockchain Em Processos De Negócio**. 2023. 56p. Dissertação de Mestrado, 2023. Disponível em: https://repositorio.ufsm.br/bitstream/handle/1/28664/DIS_PPGCC_2023_FREITAS_DL%c3%8aNIO.pdf?sequence=1&isAllowed=y. Acesso em: 10 set. 2023.

GIL, A. **Como Elaborar projetos de pesquisa, 5a Edição, editora Atlas**. São Paulo, p. 184, 2010.

IDREES, Sheikh Mohammad; NOWOSTAWSKI, Mariusz; JAMEEL, Roshan; MOURYA, Ashish Kumar. **Security Aspects of Blockchain Technology Intended for Industrial Applications.** MDPI Journal Electronics, [S. l.], p. 1-24, 16 abr. 2021.

INTERNATIONAL MARITIME ORGANIZATION. **Resoluton MSC.428(98) Maritme Cyber Risk Management in Safety Management Systems**. Web site IMO, v. 428, n. Junho 2017, p. 2017, 2017.

K. Fan, S. Wang, Y. Ren, H. Li, Y. Yang. **MedBlock: Efficient and Secure Medical Data Sharing Via Blockchain**. Journal of Medical Systems, Vol. 42, No. 8, Article 136, Agosto, 2018.

KAKAVAND, H.; KOST DE SEVRES, N.; CHILTON, B. **The blockchain revolution: an analysis of regulation and technology related to distributed ledger technologies**, 2017a.
LINUX FOUNDATION. **Hyperledger Fabric: Documentation Release Master**. 13 jan. 2021, p. 441, 2019.

M, Platt; P, McBurney. **Sybil in the Haystack: A Comprehensive Review of Blockchain Consensus Mechanisms in Search of Strong Sybil Attack Resistance.** Department of Informatics, King's College London, [S. l.], p. 3, 6 jan. 2023.

MIERS, C. et al. **Análise de mecanismos para consenso distribuído aplicados a blockchain.** SBC, 2019.

MITOLA, J. **software radio architecture**. [S.l.]: Wiley, 2000.

MOHANTA, Bhabendu Kumar; PANDA, Soumyashree S; JENA, Debasish. **An Overview of Smart Contract and Use cases in Blockchain Technology**. International Conference on Computing, Communication and Networking Technologies (ICCCNT), [s. l.], julho 2018.

MOORE, Alan D. **Python GUI Programming with Tkinter**. [S. l.: s. n.], 2018.
NUNES, Paulo Fernando Viegas. **A Definição de uma Estratégia Nacional de Cibersegurança**. NAÇÃO E DEFESA, [s. l.], ano 113, p. 127, 2012.

O QUE é QoS? Entenda para que serve a tecnologia em roteadores. [*S. l.*], 23 jul. 2018. Disponível em: https://www.techtudo.com.br/noticias/2018/07/o-que-e-qos-entenda-para-que-serve-a-tecnologia-em-roteadores.ghtml. Acesso em: 19 out. 2023.
POPOVSKI, Lewis; SOUSSOU, George. **A Brief History of Blockchain**. Legaltech News, [s. l.], 14 maio 2018. Disponível em: https://pbwt2.gjassets.com/content/uploads/2018/05/010051804-Patterson2.pdf. Acesso em: 5 out. 2023.

RICHARDSON, Ronny; NORTH, Max M. Ransomware: Evolution, Mitigation and Prevention. **International Management Review**, [*s. l.*], v. 13, p. 10-21, Mar 2019.

SOBTI, Rajeev; GEETHA, G. **Cryptographic Hash Functions: A Review**. International Journal of Computer Science Issues, [s. l.], v. 9, Mar 2012.

SZABO, N. (1997) **The Idea of Smart Contracts**.

TRELEAVEN, P.; GENDAL BROWN, R.; YANG, D. **Blockchain Technology in Finance in Computer**. IEEE Access, vol. 50, no. 9, pp. 14-17, 2017. Disponível em: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8048631.

ZHANG, Jingyu; ZHONG, Siqi; WANG, Tian; CHAO, Han-Chieh; WANG, Jin. **Blockchain-based Systems and Applications: A Survey**. Journal of Internet Technology, [S. l.], v. 21, n. 1, p. 1-14, 15 jan. 2020.

ZHENG, Z. et al. **Blockchain challenges and opportunities: A survey**. International Journal of Web and Grid Services, Inderscience Publishers (IEL), v. 14, n. 4, p. 352–375, 2018.

APÊNDICES

A. Programa Arduino para leitura de temperatura

```
int sensorPinTemp = A4;
float temperaturaC;

void setup() {
  Serial.begin(9600);
}

void loop() {
  int valorSensorTemp = analogRead(sensorPinTemp);

  temperaturaC = (valorSensorTemp / 1024.0) * 500 - 50;

  Serial.print("Temperatura: ");
  Serial.print(temperaturaC);
  Serial.print(char(176));
  Serial.print("C");
  Serial.print("\n");

  delay(1000);
}
```

B. Programa blockchain_dados.py

```
import hashlib
import datetime as date
import csv


lista_password = []
usuarios = ['CIAA']
senhas = [2023]


usuario = str(input("Informe seu usuario: ")).upper()


class Block:
    def __init__(self, index, timestamp, data, previous_hash):
        self.index = index
        self.timestamp = timestamp
        self.data = data
        self.previous_hash = previous_hash
        self.hash = self.calculate_hash()


    def calculate_hash(self):
        sha = hashlib.sha256()
        sha.update(str(self.index).encode('utf-8')+
                str(self.timestamp).encode('utf-8')+
                str(self.data).encode('utf-8')+
                str(self.previous_hash).encode('utf-8'))
        return sha.hexdigest()
```

```
class Blockchain:
    def __init__(self):
        self.chain = [self.create_genesis_block()]

    def create_genesis_block(self):
        return Block (0, date.datetime.now(), 'Genesis Block: [data-hora ,
        temperatura]', '0')

    def add_block(self, new_block):
        new_block.previous_hash = self.chain[-1].hash
        new_block.hash = new_block.calculate_hash()
        self.chain.append(new_block)

    def is_valid(self):
        for i in range (1, len(self.chain)):
            current_block = self.chain[i]
            previous_block = self.chain [i-1]

            if current_block.hash != current_block.calculate_hash():
                return False
            if current_block.previous_hash != previous_block.calculate_hash():
                return False

        return True

    def imprimir_blocos(self):
        for bloco in self.chain:
            print(f"Bloco: {bloco.index}")
            print(f"Timestamp: {bloco.timestamp}")
            print(f"Dados: {bloco.data}")
            print(f"Hash anterior: {bloco.previous_hash}")
            print(f"Hash: {bloco.hash}")
            print("\n")
```

```
for i in range (0, len(usuarios)):
        if usuario == usuarios[i]:
            senha = int(input('Informe sua senha: '))
            if senha == senhas[i]:
                print('Acesso liberado')
                print('======================')

                my_blockchain_sensor = Blockchain()

                with open("data.csv", "r")  as f:
                    reader = csv.reader(f, delimiter=",")

                    index = 1
                    for row in reader:
                        # Criando um novo bloco
                        data = row[0]

                        new_block = Block(index, date.datetime.now(), row,
                        my_blockchain_sensor.chain[-1].calculate_hash())

                        # Adicionando o novo bloco à blockchain
                        my_blockchain_sensor.add_block(new_block)
                        index += 1

                print(f'blockchain valida? {str(my_blockchain_sensor.is_valid
                ())}\n')

                my_blockchain_sensor.imprimir_blocos()

            else:
                print('Acesso negado')
```

C. Programa blockchain.py

```
import hashlib
import datetime

class Bloco():
    def __init__(self, indice, ultimo_hash, usuario, msg):
        self.indice = indice
        self.ultimo_hash = ultimo_hash
        self.usuario = usuario
        self.msg = msg
        self.datahora = datetime.datetime.now()
        self.hash = self.calcula_hash()

    def calcula_hash(self):
        dados_concatenados = str(self.datahora) + str(self.indice)  + str(self.
        ultimo_hash) + str(self.msg)
        return hashlib.sha256(dados_concatenados.encode()).hexdigest()


class Blockchain:
    def __init__(self, usuario):
        self.corrente = [self.cria_bloco_genesis(usuario)]
        self.usuario = usuario
        self.msg = ''

    def cria_bloco_genesis(self, usuario):
        return Bloco(0,'0', usuario, 'Comunicacao Baseada Em Blockchain: Chat
        Iniciado')

    def adiciona_bloco(self,msg):
        indice = len(self.corrente)
        ultimo_hash = self.corrente[-1].hash
        novo_bloco = Bloco(indice, ultimo_hash, self.usuario, msg)
        self.corrente.append(novo_bloco)

    def verifica_validade_bloco(self):
        for i in range(1,len(self.corrente)):
            bloco_atual = self.corrente[i]
            bloco_anterior = self.corrente[i-1]

            if bloco_atual.hash != bloco_atual.calcula_hash():
                return False
            elif bloco_atual.ultimo_hash != bloco_anterior.calcula_hash():
                return False

        return True

    def le_msg(self):
        return self.msg
```

D. Programa criar_usuarios.py

```
import hashlib

def criar_ususario():

    usuario = input('Informe o nome desejado para o usuario: ').upper()
    senha = input('Informe a senha desejada: ')

    hash_senha = hashlib.sha256(senha.encode()).hexdigest()

    with open('usuarios.txt', 'a') as arquivo_de_usuarios:
        arquivo_de_usuarios.write(f'{usuario}:{hash_senha}\n')

    print('Usuario Criado!')

criar_ususario()
```

E. Programa vallida_login.py

```
import hashlib

def valida_login(usuario, senha):

    with open('usuarios.txt', 'r') as arquivo_de_usuarios:

        for linha in arquivo_de_usuarios:
            usr,passwd = linha.strip().split(':')

            hash_senha = hashlib.sha256(senha.encode()).hexdigest()

            if usuario == usr and hash_senha == passwd:
                return True
    return False
```

F. Programa cria_blockchain.py

```
import blockchain

def estrutura_chat():
    print('\n ====================== \n')
    print('    Acesso liberado')
    print('\n ====================== \n')
    print("    Instrucoes:")
    print("    Para ler as mensagens digite 'ler'")
    print("    Para consultar validade da blockchain digite 'status'")
    print("    Para sair do chat digite 'sair'")
    print('\n ====================== \n')

def estrutura_conversa():
    print('\n ====================== \n')
    print('      CONVERSA')

def define_blockchain(usuario_login):
    comunicacao_baseada_em_blockchain = blockchain.Blockchain(usuario_login)

    estrutura_chat()

    while True:
        msg = input('Mensagem: ')

        if msg == 'sair':
            print('Chat Encerrado!')
            break

        if msg == 'status':
           status = comunicacao_baseada_em_blockchain.verifica_validade_bloco()
           if status == True:
               print('blockchain VALIDA')
           else:
               print('blockchain INVALIDA')

        elif msg == 'ler':
            estrutura_conversa()
            for bloco in comunicacao_baseada_em_blockchain.corrente:
                print(f"Data-Hora: {bloco.datahora}")
                print(f"Remetente: {bloco.usuario}")
                print(f"Mensagem: {bloco.msg}")
                print("\n")

        comunicacao_baseada_em_blockchain.adiciona_bloco(msg)

if __name__ == '__main__':
    pass
```

G. Programa executável.py

```
import valida_login
import cria_blockchain

if __name__ == '__main__':
    print('Login')
    usuario_login = input('Usuario: ').upper()
    senha_login = input('Senha: ')

    if valida_login.valida_login(usuario_login,senha_login):
        print('Login Bem-Sucedido')
        cria_blockchain.define_blockchain(usuario_login)
    else:
        print('Usuario/senha incorretos.')
```

H. Programa terminal_grafico.py

```
import tkinter as tk
import valida_login
import datetime

blockchain = []

def fazer_login():
    usuario = entry_usuario.get().upper()
    senha = entry_senha.get()

    if valida_login.valida_login(usuario, senha):
        janela_login.destroy()
        abrir_executavel(usuario)
    else:
        mensagem_login.config(text='Usuario/senha incorretos.')

def abrir_executavel(usuario):
    janela_executavel = tk.Tk()
    janela_executavel.title('CBB')
    janela_executavel.geometry('500x500')

    texto_saida = tk.Text(janela_executavel, height=10, width=10, wrap=tk.WORD)
    texto_saida.pack(side=tk.TOP, fill=tk.BOTH, expand=True)

    scrollbar = tk.Scrollbar(texto_saida)
    scrollbar.pack(side=tk.RIGHT, fill=tk.Y)

    texto_saida.config(yscrollcommand=scrollbar.set)
    scrollbar.config(command=texto_saida.yview)

    caixa_texto = tk.Entry(janela_executavel)
    caixa_texto.pack(fill=tk.X)

    def processa_texto():
        texto = caixa_texto.get()
        if texto.lower() == 'ler':
            exibir_blockchain(texto_saida)
        else:
            atualizar_blockchain(usuario,texto)
            exibir_blockchain(texto_saida)
            caixa_texto.delete(0, tk.END)

    botao_processar = tk.Button(janela_executavel, text='Enviar',
    command=processa_texto)
    botao_processar.pack()

    caixa_texto.bind('<Return>', lambda event=None: processa_texto())

    janela_executavel.mainloop()
```

```
def exibir_blockchain(text_widget):
    text_widget.delete('1.0', tk.END)
    for bloco in blockchain:
        text_widget.insert(tk.END, bloco + '\n')

def atualizar_blockchain(usuario, texto):
    indice = len(blockchain) + 1
    timestamp = datetime.datetime.now()
    novo_bloco = f"Índice: {indice}\nTimestamp: {timestamp}\nRemetente:
    {usuario}\nMensagem: {texto}\n"
    blockchain.append(novo_bloco)



janela_login = tk.Tk()
janela_login.geometry("500x150")
janela_login.title("Login - Comunicacao Baseada em Blockchain")

label_usuario = tk.Label(janela_login, text="Usuario:")
label_usuario.pack()
entry_usuario = tk.Entry(janela_login)
entry_usuario.pack()

label_senha = tk.Label(janela_login, text="Senha:")
label_senha.pack()
entry_senha = tk.Entry(janela_login, show="*")
entry_senha.pack()


botao_login = tk.Button(janela_login, text="Login", command=fazer_login)
botao_login.pack()


mensagem_login = tk.Label(janela_login, text="")
mensagem_login.pack()


janela_login.mainloop()
```