BlockSigner, uma Malha de Confiança para Documentos

Gabriel Marques, Murilo B. Flor, Brivaldo Junior

[1]Faculdade de Computação (Facom) – Universidade Federal de Mato Grosso do Sul (UFMS)
79.070-900 – Campo Grande – MS – Brazil

{brivaldo}@facom.ufms.br

{murilo.flor, gabriel-marques}@aluno.ufms.br

Abstract. *With the constant evolution of technology and the use of electronic devices, the use of digital signatures has also grown in recent years.* Blockchain *with its wide range of applications, from property proofing, legal decisions, to logistics processes, has been providing an alternative to handle the high flow of documents and increasing concern about security and also paper spending, the basis of the documents. Security has been the main target of studies and in an attempt to mitigate possible fraud, current algorithms use various techniques to improve client-side security. The use of* blockchains *as a server-side security mechanism brings real benefits to the storage and use of digital signatures. However, there are still some barriers to the consolidation of this technology.*

Resumo. *Com a constante evolução da tecnologia e utilização de aparelhos eletrônicos, o uso de assinaturas digitais também tem crescido nos últimos anos. A* Blockchain *com sua ampla gama de aplicações, desde comprovações de propriedade de imóveis, decisões jurídicas, até processos de logística, vem mostrando uma alternativa para lidar com o alto fluxo de documentos e crescente aumento da preocupação com segurança e também dos gastos com papel, matéria base dos documentos. A segurança tem sido o principal alvo de estudos e na tentativa de mitigar possíveis fraudes, algoritmos atuais utilizam várias técnicas para melhorar a segurança no lado do cliente. A utilização de* blockchains *como um mecanismo de segurança do lado servidor traz benefícios reais para o armazenamento e utilização das assinaturas digitais. Contudo, ainda existem algumas barreiras para a consolidação desta tecnologia.*

1. Introdução

A crescente demanda pela validação de documentos e a autenticidade das informações digitais cresceu muito nos últimos anos. Mecanismos criptográficos como DES [Thakur and Kumar 2011], 3DES [Singh 2013] e RSA [Mahajan and Sachdeva 2013] são padrões utilizados para validar se um usuário pode acessar determinados serviços. No início, apenas chaves (senhas) eram utilizadas para realizar esta validação e hoje isso já se provou insuficiente. Tecnologias como SSO [Cohen et al. 2001] servem para criar camadas adicionais de validação do lado do usuário (exige que sejam informadas senhas pessoais, dados pessoais ou outros mecanismos como o OTP [Rubin 1996]).

Contudo, embora a tecnologia tenha evoluído no lado do cliente, no lado servidor, as senhas continuam sendo armazenadas em *hashes* criptográficos que são utilizados no

processo de autenticação. Ou seja, o usuário digita sua senha em uma tela de autenticação, o sistema calcula o *hash* e transmite o par *hash* e usuário ao servidor que valida se esta informação está correta.

O problema surge, quando um administrador mal intencionado usa informações dos usuários para modificar arquivos ou assinar dados sem seu consentimento. Uma vez que a informação local do servidor pode ser modificada, não existe garantia para o usuário que sua assinatura não foi forjada.

Para tornar uma aplicação mais segura, uma possibilidade é criar um histórico compartilhado de arquivos assinados de forma distribuída. Desta maneira, quando um servidor que armazena documentos assinados alterar, maliciosamente, um arquivo assinado, ele deverá obrigatóriamente submeter a nova moficação para todos os outros nós da rede. Por se tratar de uma nova assinatura, é simples para o usuário determinar que um novo arquivo, possivelmente fraudulento, foi assinado.

Por outro lado, se a mudança ocorrer em um arquivo previamente assinado e, maliciosamente, o atacante mudar a *blockchain* [Greve et al.] local, isso também será detectado pelos outros nós da rede, invalidando o bloco. Esta camada adicional de segurança garante ao usuário que um administrador mal intencionado não possa modificar ou assinar documentos sem deixar rastros.

A forma mais usual de *blockchains* é na manutenção de carteiras digitais para armazenamento e troca financeira. Ele serve como um livro caixa distribuído de transações digitais. Este trabalho utiliza a ideia de livro caixa, mas não distribui para todos os nós todos os arquivos (forma tradicional no uso de *blockchains*). Apenas os *hashes* assinados dos arquivos são distribuídos a cada um minuto entre todos os nós. Isso reduz drasticamente o tamanho e a quantidade de informações trocadas na malha de servidores.

Embora o ideal para larga escala seja o uso de P2P para reduzir o número de nós que um servidor está se comunicando, este trabalho usou uma estrutura de conexão NxN (todos para todos), pois é esperado que esse valor não seja superior à 50 servidores (um para cada Estado da federação brasileira). Essa restrição existe, pois cada Estado possui, geralmente, apenas uma ou duas fundações de apoio. Uma versão futura com suporte a P2P já está em andamento.

Este trabalho está organizado da seguinte maneira: Na Seção 2 será abordada a fundamentação teórica e principais técnicas utilizadas em sistemas distribuídos em *blockchain*. Na Seção 3 serão apresentados trabalhos relacionados. Na Seção 4 os modelos e a implementação realizada. E na Seção 5 a conclusão e trabalhos futuros.

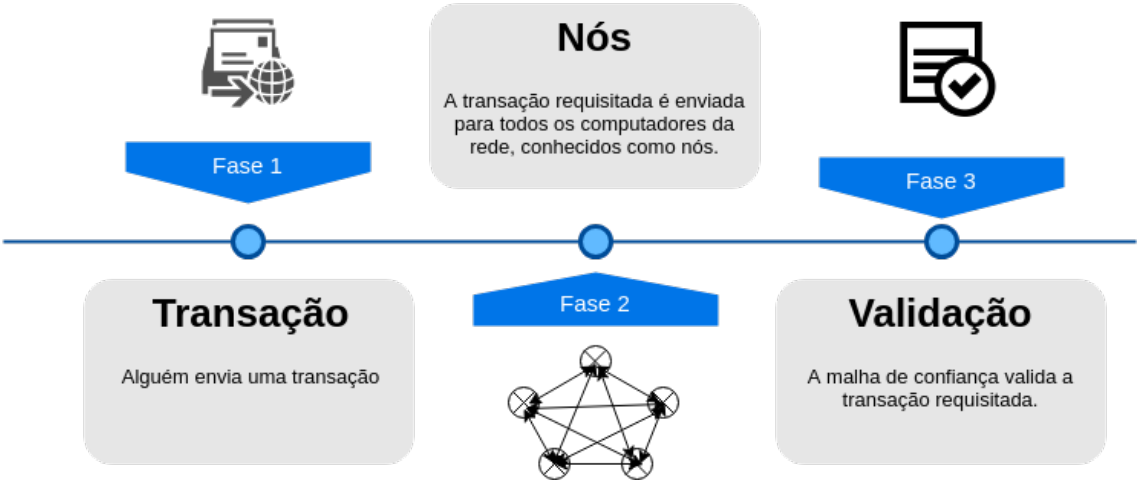
2. Fundamentação Teórica

2.1. Blockchain

A *blockchain* (também conhecido como "o protocolo de confiança") é um registro digital de transações distribuído que usa a descentralização como medida de segurança. O nome deriva de sua estrutura, na qual registros individuais chamados blocos, são ligados em uma única lista (cadeia). Funciona como um livro-razão, só que de forma pública, compartilhada e universal entre os participantes, que cria consenso e confiança na comunicação direta entre duas partes, ou seja, sem o intermédio de terceiros.

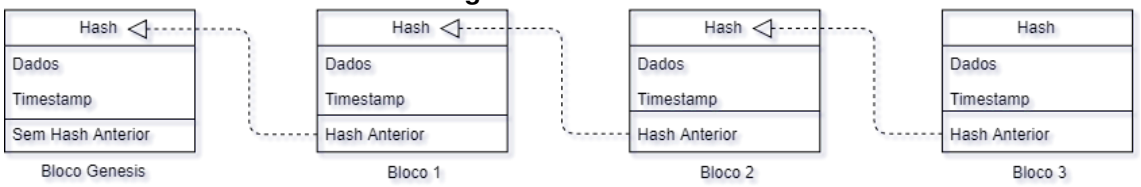
Cada transação adicionada a uma *blockchain* é validada por vários nós (computadores conectados a rede) pertencentes a rede, veja a Figura 1. Estes sistemas, que são configurados para monitorar tipos específicos de transações, formam uma rede *peer-to-peer*. Eles trabalham juntos para garantir que cada transação seja válida antes de ser adicionada a *blockchain*. Essa rede descentralizada de computadores garante que nenhum usuário possa adicionar blocos inválidos à malha.

Figura 1. Estágios de um transação



Quando um novo bloco é adicionado a *blockchain*, ele é vinculado ao bloco anterior, veja a Figura 2, usando um hash criptográfico gerado a partir do conteúdo do bloco anterior. Isso garante que a corrente nunca seja quebrada e que cada bloco seja permanentemente gravado. Também é intencionalmente difícil alterar informações passadas na *blockchain*, uma vez que todos os blocos subsequentes devem ser alterados primeiro em mais da metade dos nós que estão executando na rede. Portanto, a cadeia está em constante crescimento, dificultando ainda mais sua invasão pelo tempo.

Figura 2. *Blockchain*



Os blocos são adicionados à *blockchain* de modo linear e cronológico. Cada nó da rede tem a tarefa de validar e repassar transações, além de obter uma cópia da *blockchain* após o ingresso na rede. A *blockchain* possui informação completa sobre endereços e conteúdos diretamente do bloco gênese (Primeiro bloco da cadeia, normalmente codificado manualmente pelos desenvolvedores no código das aplicações) até o bloco mais recentemente concluído.

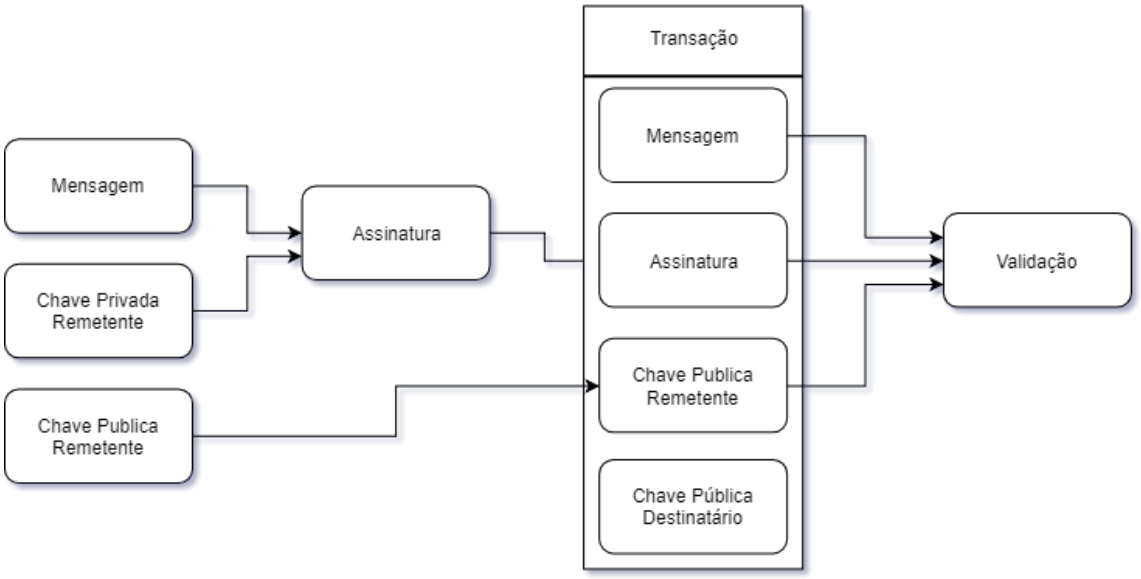
2.1.1. Mineração

Todos os nós da malha criam um novo bloco localmente com as transações pendentes, que foram enviadas entre si a cada inserção em qualquer nó da malha. Estes computadores, conhecidos como mineradores, competem entre si para ver se o seu bloco de transações em aberto, ou seja, que precisam ser validadas por todos os nós da rede, se tornará o próximo bloco da lista para toda a malha. A validação pode ser feita de maneiras diferentes de acordo com a lógica de cada rede. Essas variações são nomeadas como protocolos de consenso:

- ***Proof of Work* (POW)**: É utilizado para a prevenção de ataques cibernéticos como DDOS e Spam. Ele surgiu como uma tentativa de reduzir os efeitos desses ataques utilizando de funções hash, posteriormente explicado. Para um usuário realizar alguma ação, ele deve ser capaz de provar que realizou alguma tarefa, essa prova é a garantia de que o usuário gastou tempo para gerar uma resposta que satisfaça algum requisito do avaliador. Para esse sistema funcionar, tal prova deve ser trabalhosa de ser criada, mas facilmente verificada pelo avaliador.
  O POW é muito criticado devido ao alto consumo de energia. Mas devido a dificuldade deste processo, é sem dúvida alguma a *blockchain* mais segura;
- ***Proof of Stake (POS)*** : A forma de mineração POS (ou Prova de Participação) usa um sorteio aleatório para decidir quem será o criador do próximo bloco. Nesse modelo o potencial criador já deve contar com ativos na moeda específica e quem tiver mais moedas tem mais chances de ser o criador/sorteado. É necessário alocar uma quantidade de moedas para este processo e caso tente comprometer ou alterar o bloco perderá suas moedas. Isto em teoria garante a integridade dos participantes.
  Existem diversas formas de aplicar este algoritmo de consenso. Em algumas o rateio de novas moedas é feito de forma proporcional as moedas existentes. Em outras, todas as moedas foram pré-criadas assim cada bloco novo não dá recompensa. O criador apenas recebe as taxas das transações processadas naquele bloco;
- ***Proof of Authority* (POA)**: O Proof of authority (ou prova de autoridade) é um mecanismo de consenso em *blockchains* privadas. Essencialmente este método oferece a um cliente (ou um número específico de clientes) o direito de mineirar todos os blocos da *blockchain*;
- ***Proof of Capacity* (POC)**: Por fim, existe a forma de mineração Proof of Capacity (ou prova de capacidade), muito menos conhecida e utilizada. Neste caso, é usado espaço no HD do minerador como prova de capacidade. É semelhante ao PoS por ser um sorteio, porém nesse caso, quanto mais espaço no HD disponível maior a chance de ser o sorteado para minerar o próximo bloco. Devido ao baixo custo de energia, é considerado a opção mais sustentável.

Porém, em BlockSigner, não foi utilizado nenhum protocolo de consenso, uma vez que foi desenvolvido com o intuito de ser utilizado em redes privadas, que teoricamente, possuem nós confiáveis e controlados. Nesta API, a validação de transações é feita pelo cálculo da verificação de chaves públicas e privadas das transações. Quem o fizer primeiro, será o nó a publicar o novo bloco na lista. Ainda sobre BlockSigner, o novo bloco só será publicado após todos os outros nós verificarem se o novo bloco está correto. Assim, é permitido a adição deste bloco em toda a malha da *blockchain*.

Figura 3. Diagrama de uma Transação.



Cada um destes novos blocos é adicionado na *blockchain* a cada 1 minuto. Uma vez que um bloco é adicionado a lista, este se torna imutável, sendo impossível sua deleção ou alteração. Portando, todo participante da rede é também chamado de minerador, os quais são responsáveis por criar os blocos com as novas transações em aberto como descrito anteriormente.

2.1.2. Propriedades de uma Blockchain

As principais propriedades da tecnologia *blockchain* que contribuem de forma inovadora para o desenvolvimento de aplicações e sistemas são as seguintes:

- **Descentralização**: As aplicações e sistemas são executados de maneira distribuída, por meio do estabelecimento de confiança entre as partes, sem a necessidade de uma entidade intermediária confiável;
- **Disponibilidade e Integridade**: Todo o conjunto de dados e transações são replicados em diferentes nós de maneira segura, de forma a manter o sistema disponível e consistente;
- **Transparência e Auditabilidade**: Todas as transações registradas na *blockchain* são públicas, podendo ser verificadas e auditadas. Além disso, os códigos da tecnologia costumam ser abertos, passíveis de verificação;
- **Imutabilidade e Irrefutabilidade**: As transações registradas são imutáveis. Uma vez registradas não podem ser refutadas. Atualizações são possíveis a partir da geração de novas transações e realização de novo consenso;
- **Privacidade e Anonimidade**: É possível oferecer privacidade aos usuários sem que os terceiros envolvidos tenham acesso e controle dos seus dados. Na tecnologia, cada usuário gerencia suas próprias chaves e cada nó servidor armazena apenas fragmentos criptografados de dados do usuário. Transações são até certo ponto anônimas, com base no endereço dos envolvidos na *blockchain*;

- **Desintermediação**: A *blockchain* possibilita a integração entre diversos sistemas de forma direta e eficiente. Assim, é considerada um conector de sistemas complexos (sistemas de sistemas), permitindo a eliminação de intermediários de maneira a simplificar o projeto dos sistemas e processos [Croman et al. 2016];
- **Cooperação e Incentivos**: Oferta de modelo de negócios à base de incentivos, à luz da teoria dos jogos. O consenso sob demanda passa a ser oferecido como serviço em diversos níveis e escopos;
- **Transações mais rápidas**: Transações interbancárias podem potencialmente levar dias para serem compensadas e terem acordo final, especialmente fora do horário de trabalho. Transações com *blockchain* podem reduzir o tempo de transações para minutos e são processadas 24 horas por dia e 7 dias por semana.

Todos essas características são reais e tangíveis. Porém, há 3 grandes desafios para implementar uma *blockchain* de forma sólida:

1. A seleção de funções hash criptográficas fortes e resilientes ao tempo ou o projeto de técnicas de atenuação de vulnerabilidades de função hash;
2. O design e a implementação correta e segura do código que implementa uma *blockchain*;
3. A seleção e implementação adequadas de algoritmos de eleição de líder eficientes, robustos e confiáveis.

Detalhando um pouco mais sobre os desafios. As funções de criptografia hash e criptografia assimétrica são baseadas no que é chamado de funções unidirecionais. Alguns problemas matemáticos e seus cálculos correspondentes são computacionalmente fáceis em uma direção, mas computacionalmente inviáveis na direção inversa. Um número composto poderia ser resultado de diferentes combinações de números sendo multiplicados. Os algoritmos de hash seguro (SHA-2) usam números de 224 a 512 dígitos binários de comprimento [Oliveira 2012]. Em Blocksigner recomenda-se o SHA-256.

Um desafio para o design da *blockchain* é que a propriedade de computacionalmente inviável não é uma propriedade estática de uma determinada função hash, mas muda com o avanço da matemática e da computação. Um algoritmo de hash que é criptograficamente seguro (computacionalmente inviável para encontrar o inverso) hoje pode não ser criptograficamente seguro daqui a 20 anos. Isso dificulta nossa capacidade de projetar algoritmos que possa resistir ao teste do tempo.

2.2. Criptografia

Quando falamos de informação e transportamos este conceito para o meio digital, particularmente na utilização das redes públicas de computação como a internet, é relevante ao ser humano à credibilidade nos sistemas computacionais, estes que inseridos nos fundamentos da segurança da informação, são definidos pela disponibilidade, integridade, controle de acesso, autenticidade, não-repudiação e finalmente a privacidade. Neste cenário apresentam-se os dois tipos básicos de criptografia: a simétrica e a assimétrica.

2.2.1. Criptografia Simétrica

O modelo mais antigo de criptografia, em que a chave, isto é, o elemento que dá acesso à mensagem oculta trocada entre duas partes, é igual (simétrica) para ambas as partes e

deve permanecer em segredo (privada). Tipicamente, esta chave é representada por uma senha, usada tanto pelo remetente para codificar a mensagem numa ponta, como pelo destinatário para decodificá-la na outra.

O principal problema residente na utilização deste sistema de criptografia é que quando a chave de ciframento é a mesma utilizada para deciframento, ambas precisam ser compartilhadas previamente entre origem e destino, antes de se estabelecer o canal criptográfico desejado, e durante o processo de compartilhamento a senha pode ser interceptada, por isso é fundamental utilizar um canal seguro durante o compartilhamento, este independente do destinado à comunicação sigilosa, uma vez que qualquer um que tenha acesso à senha poderá descobrir o conteúdo secreto da mensagem. Outras lacunas são interpostas a este sistema:

- Como cada par necessita de uma chave para se comunicar de forma segura, para uma rede de n usuários precisaríamos de algo da ordem de n^2 chaves, quantidade esta que dificulta a gerência das chaves;
- A chave deve ser trocada entre as partes e armazenada de forma segura, o que nem sempre é fácil de ser garantido;
- A criptografia simétrica não garante os princípios de autenticidade e não-repudiação.

2.2.2. Criptografia Assimétrica

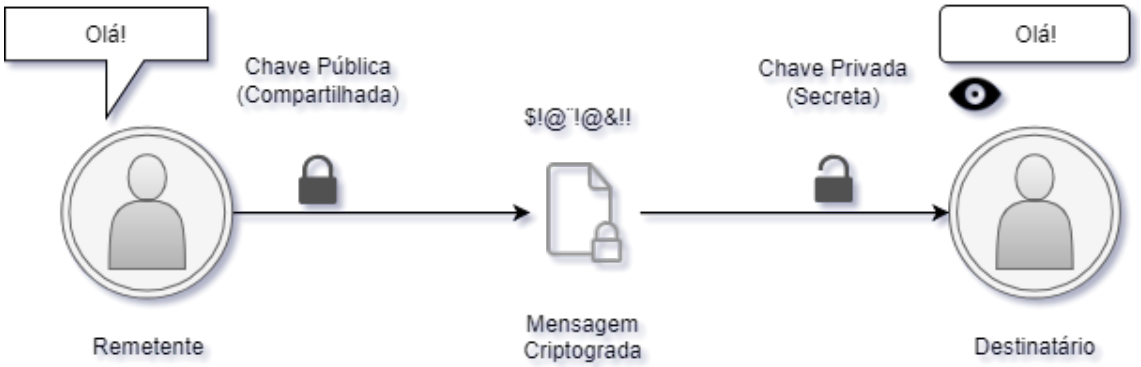
A criptografia assimétrica é qualquer tipo de sistema de criptografia que utiliza um par de chaves [Oliveira 2012]: chaves públicas que podem ser amplamente divulgadas e chaves privadas, que são conhecidas apenas pelos seus respectivos donos. Este método realiza duas funções:

- **Autenticação**: A chave pública consegue verificar se o detentor da chave privada associada enviou a mensagem;
- **Criptografia**: Apenas o detentor da chave privada associada pode descriptografar a mensagem criptografada com a chave pública.

O uso desta tecnologia é relevante ao notar-se que proteger um arquivo somente por uma senha ainda seria vulnerável. Se, por exemplo, um determinado usuário X criptografa um documento com uma senha própria, outros usuários teriam de ter a senha do usuário X para descriptografar, além do fato de que esta senha pode ser interceptada por usuários maléficos durante o compartilhamento desta senha. A utilização do par de chaves cria, de forma abstrata, uma relação como a de uma caixa de correios, onde todos sabem a sua localização para o envio de mensagens, porém somente o dono desta consegue abri-la e ver as mensagens. De forma análoga, quando um usuário X envia uma mensagem assimetricamente criptografada com a chave pública de um usuário Y, somente o usuário Y consegue descriptografar a mensagem, com sua chave privada.

A força e segurança da criptografia assimétrica dependem de como os usuários mantém suas respectivas chaves privadas seguras, pois se um usuário malicioso roubar a chave privada do usuário X, esta poderá ser usada para descriptografar todas as mensagens que tinham como endereço final o próprio usuário X. Entretanto, o usuário malicioso não conseguirá descriptografar as mensagens que foram enviadas pelo usuário X, pois para isso seria necessário a chave privada do destinatário.

Figura 4. Par de Chaves Assimétrica.



2.2.3. Função de *Hash*

A aplicação primaria da função de *hash* na criptografia é a integridade da mensagem. O valor hash gera uma impressão digital do conteúdo da mensagem passada como entrada. Esta impressão garante que a mensagem não foi modificada por um usuário malicioso, virus ou outros males. A força de um algoritmo de *hash* está em sua resistência a colisões (duas entradas não geram a mesma saída) e a impossibilidade de derivar a mensagem de entrada usando apenas o *hash* de saída.

Por conta destas características, a função de hashing é o principal algoritmo usado na *blockchain*. A saída desta rotina não pode ser descriptografada de volta para o texto original, ou seja, é uma função hash criptográfica unidirecional. Independente do tamanho da entrada, esta função retorna sempre um hash com o mesmo número de caracteres. Note que, a omissão de um único caracter na entrada modifica totalmente a saída da função.

2.3. Documentos e Assinaturas Digitais

A assinatura digital é um *token* eletronico que cria uma relação entre uma entidade e um registro. Seu propósito é validar e autenticar documentos digitais. A validação refere-se ao processo de certificação do conteúdo do documento, enquanto a autenticação relata o processo de certificação do remetente relacionado ao documento.

Pode-se dizer que a assinatura digital é uma versão eletrônica de uma assinatura de próprio punho. O processo de assinar é implementado com a ajuda da criptografia assimétrica. O assinante utiliza sua chave privada para assinar o hash do documento. Esta assinatura é utilizada para garantir que o conteúdo da mensagem ou documento não foi alterado.

Sua natureza variada forneceu um mecanismo fácil, rápido, preciso e conveniente para criar, armazenar, transmitir e recuperar dados sem envolver formalidades tradicionais baseadas em papel. Cada vez mais os negócios, comunicação, dados oficiais e transações comerciais estão sendo realizados no ciberespaço.

A assinatura digital pode ser usada, por exemplo, na documentação de estágio dos alunos, o que inclui termos de compromisso, relatórios de atividades e comunicados de desligamento. Comprovantes de matrícula também podem ser assinados eletronicamente, bem como a liberação de documentos acadêmicos e certificados de participação

em congressos.

Pesquisadores também podem usar a plataforma para assinar pedidos de bolsa ou garantir a documentação para uma iniciação científica. Por fim, vale lembrar de outros processos burocráticos que utilizam o papel em grande quantidade , como requisições de trancamentos, de mudanças de disciplina ou de transferência de cursos.

Além disso, a assinatura digital pode ser uma aliada em tarefas mais administrativas, como nos contratos com fornecedores e prestadores de serviços ou na assinatura de convênios e parcerias com empresas ou outras instituições de ensino.

Os objetivos da assinatura digital podem ser listados como:

- **Criar autenticidade do remetente**: A assinatura digital permite ao destinatário de uma mensagem ou documento verificar o remetente. Ela é única para cada usuário e, portanto, uma assinatura digital válida é usada para afirmar que uma mensagem foi originada de um usuário específico. Assim, a qualquer momento após a criação de qualquer material digital, a autenticidade do criador pode ser verificada.
- **Criar autenticidade do documento**: Uma mensagem ou documento assinado digitalmente não pode ser alterado sem invalidar sua assinatura, seja o documento criptografado ou não. Além disso, esta assinatura pode confirmar se o documento foi ou não alterado. O mecanismo também garante ao remetente que ninguém diferente dele será capaz de modificar, alterar ou adulterar o documento.
- **Não repúdio**: Como uma assinatura digital é equivalente a uma assinatura escrita à mão, seu uso é tomado como um sinal de reconhecimento de uma mensagem ou documento. Assim, se alguém assinou digitalmente um documento, ele não pode negar a responsabilidade decorrente de tal documento.

3. Trabalhos Relacionados

Com o objetivo de entender as arquiteturas distribuídas, foram analisadas algumas iniciativas e sistemas para a criação de aplicações de assinatura digital baseadas em *blockchain*:

- ***Original My*** [JR.2015] - Efetua provas de autenticidade para documentos digitais. A plataforma efetua os registros de autenticidade em 4 *blockchains* públicos e outros privados, sendo os mais relevantes Bitcoin [Nakamoto 2008] e Ethereum [Wood 2014].
  Desta maneira, a plataforma se mostra bastante flexível inclusive na precificação, efetuando os registros de acordo com a necessidade dos clientes que fazem integração com o ambiente de APIs. A aplicação recebe documentos em formato PDF e o publica em *blockchains* de terceiros.
  Suas funcionalidades que podem ser relacionadas com este projeto são:
  - Registro de Autenticidade: Qualquer documento digital pode ter sua existência comprovada através de um carimbo de tempo fornecido por um *blockchain* público. Conteúdos como obras de arte, declarações, propostas, relatórios e qualquer outro tipo de documento.
  - Identidade Blockchain: Através de seu app, é feito a validação de identidade dos usuários. Após um cadastro completo, o sistema cria uma Identidade Blockchain que fica em posse do usuário. Através dessa identidade

única e exclusiva, o usuário pode efetuar ações na plataforma, como assinatura de contratos pelo app ou através do site, por exemplo.

- ***Signatura*** - Permite que várias partes assine documentos conjuntamente, juridicamente vinculativas e autenticadas, de tal forma que ninguém possa repudiar a data, o conteúdo ou as assinaturas.
  Os documentos são criptografados, enviados e os assinantes exigidos selecionados, concedendo-lhes acesso imediato aos arquivos. Os participantes se autenticam, baixam, descriptografam, revisam e assinam digitalmente a documentação. Finalmente, quando totalmente assinado, os documentos são autenticados.
  Além disso, Signature também se baseia no conceito do uso do *blockchain* Bitcoin como um esquema de assinatura digital.
- **Exonum / BestSign** - O Exonum é uma framework que permite criar aplicações de *blockchains* com permissão segura. Isso signifca que um número limitado de nós podem publicar novas transações para o *blockchain*. Essa abordagem se torna interessante uma vez que se tem um ou um grupo de pessoas responsáveis por manter o controle do *blockchain* (por exemplo, definir e atualizar as regras do processamento das transações). Isso não significa que o controle dos mantenedores é irrestrito. Existem regras bem definidas de processamento de transações que nem todos os administradores da rede juntos podem burlar.
  BestSign é um dos *blockchains* desenvolvidos por Exonum. Seu principal objetivo é fornecer aos usuários finais um produto que atenda aos critérios de validade judicial, conveniência, segurança e facilidade de uso. Tais critérios antes mantidos de forma física pela empresa, agora mantidos de forma segura e digital via *blockchain*.

4. Implementação e Experimentação

4.1. Blockchain

A Blocksigner foi desenvolvida baseando-se na busca por maior segurança para tais documentos assinados digitalmente.

Sua implementação foi fortemente baseada na estrutura de *blockchain* do Bitcoin, com algumas particularidades:

- ***Moedas***: Diferente de outras *blockchains* que possuem valor econômico agregado ao *blockchain*, Blocksigner é uma rede que não remunera os nós conectados a ela, uma vez que não existe um protocolo de consenso como *proof of work*, portanto o seu intuito é unicamente para armazenamento de registros de documentos assinados;
- ***Rede privada*** Como Blocksigner foi desenvolvida com o intuito de ser implementada em um ambiente privado, como por exemplo a RNP, tornou-se então necessário especificar quais serão os endereços de rede a se conectarem com a malha (detalhado no manual da API);
- ***Custo computacional ao minerar***: Outro fator que diferencia Blocksigner do Bitcoin, é que em nesta *Blockchain* ao invés do nó que está minerando o novo bloco, gastar esforço computacional para valida-lo, somente é necessário aguardar o tempo de mineração e se houver transações em aberto, um novo bloco será minerado;

- ***Chaves ocultas***: Não existe necessidade de o usuário final conhecer e armazenar seu par de chaves assimétrica, uma vez que Blocksigner foi desenvolvido para integrar um sistema consolidado e seus vários usuários finais. Portanto, foi definido o uso do CPF para simplificar o seu uso. Note que, há uma correlação interna entre o CPF e o par de chaves assimétrica.

4.2. Dependências

Esta API foi desenvolvida usando a linguagem Python e ao todo foi necessário incluir 15 dependências [Murilo B. Flor 2018]. Dentre elas, as mais importantes são:

- ***Flask*** - Esta biblioteca é o coração de Blocksigner e possui várias funções para desenvolvimento web, como recebimento e tratamento de requisições HTTP;
- ***Requests*** - Envio e tratamento de requisições HTTP;
- ***PyCrypto*** - Esta é uma coleção de funções hash seguras (como SHA256) e vários algoritmos de criptografia (AES, DES, RSA, etc.).

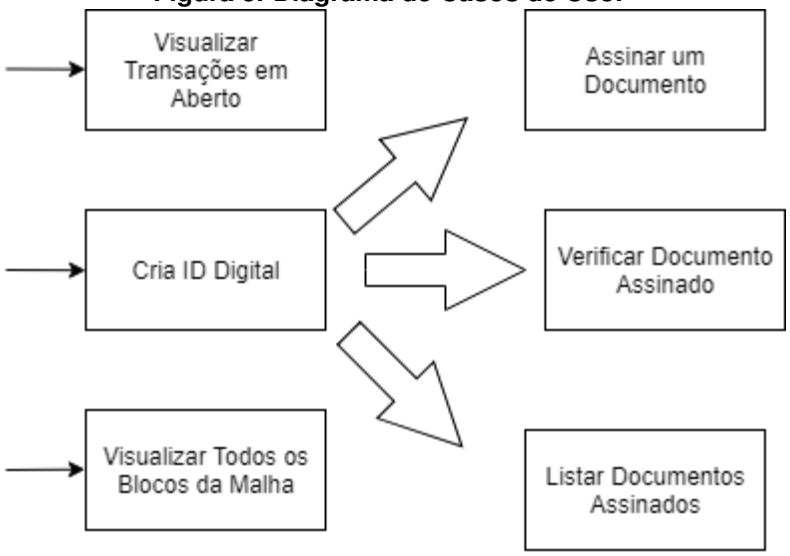
4.3. Casos de Uso

Foi implementado portanto um *blockchain* para armazenamento dos hashes assinados dos arquivos de uma determinada rede de usuários. Esta API fornece todos benefícios já citados, vistos na seção 2.1.2, em conjunto do armazenamento distribuídos de tais hashes assinados. De maneira mais prática, podemos citar seus casos de uso. São eles:

- **Criação do ID Digital** - A API possibilita ao usuário a criação de uma ”carteira”. Esta identidade é criada a partir de um CPF válido. Com ela, é possível assinar hashes de documentos, salvando tal transação permanentemente no *blockchain*.
- **Assinar um Documento** - Após possuir uma ID Digital, pode-se criar uma transação. Esta transação requer o hash do documento a ser assinado e o CPF do usuário assinante. Assim, caso as informações sejam válidas, o documento é assinado e guardado no *blockchain*.
- **Transações em Aberto** - É possível verificar quais transações ainda estão em espera para serem validadas pelos nós da rede.
- **Verificar Hash de um Documento** - Com o registro feito no *blockchain*, a assinatura digital é salva permanentemente na cadeia. Isso possibilita a verificação independente do próprio objeto. Assim, é possível provar se um documento foi assinado, sem a necessidade de revelar o documento para terceiros ou usuários não autorizados. Para isto, basta ter o hash do documento assinado e o CPF associado a esta assinatura.
- **Listar Documentos Assinados** - Pode ser feito uma varredura por toda a lista de blocos, buscando todos os documentos assinados por um CPF específico. Basta informar a API o CPF do usuário a ser buscado, e todos os seus documentos serão mostrados.
- **Listar Blocos** - É possível também visualizar toda a lista de blocos, desde o bloco genêse, até o mais atual.

Para uma análise mais detalhada e técnica sobre a API, é recomendado a leitura da documentação para desenvolvedores, hospedada no GitLab da FACOM-UFMS [Murilo B. Flor 2018].

Figura 5. Diagrama de Casos de Uso.



4.4. Ambiente Computacional

Ao ser inicializada, a API carrega em sua execução uma lista de nós da rede. Note que este arquivo pode ser alterado e que, por padrão, vem configurado no ambiente de teste com 3 endereços locais de máquina, nas portas 5000, 5001 e 5002. Após isso, é feito uma tentativa de atualização entre esses nós. Ou seja, caso algum nó esteja mais atualizado (tenha uma cadeia maior e válida), sua própria *blockchain* local é substituida pela mais atualizada dentre os outros nós da malha. A partir disso, a malha já estará em execução e pronta para aceitar as requisições detalhadas acima.

A API também oferece suporte para a técnica de internacionalização, dando suporte a mais de um idioma e/ou região.

4.4.1. Hardware de Teste

Testar é um processo que tem como objetivo encontrar as falhas em um sistema. Pode ser para eliminar erros ou por motivos de aceitação das funcionalidades [Ben-Menachem 2005]. Embora os participantes de um processo de desenvolvimento de um sistema concordem que é muito melhor impedir falhas a procurá-las e corrigi-las, a realidade é que ainda não se é capaz de produzir sistemas livres de falhas, portanto o teste é um elemento essencial no desenvolvimento do sistema em questão. A API foi desenvolvida para ser executada em um ambiente linux, provido de apache e python 3. Os testes realistas foram feitos em ambiente de containers Linux (LXC) [Helsley 2009] e também em hardware físico:

- **Hardware Simulado:**
  - Debian 9 64bits
  - Processador Xeon Dual Core 2.0Ghz
  - 64MB de memória RAM
- **Hardware Físico:**
  - Ubuntu 18.04 LTS

- Xeon dual core 2.0Ghz
- 8GB RAM

É importante notar que este hardware não é necessariamente o mínimo requerido para executar a API, e somente é a título de informação.

5. Conclusão e Trabalhos Futuros

Neste trabalho é proposto a criação de uma blockchain para armazenamento de assinaturas digitais e suas aplicações. A proposta do trabalho inclui o uso de uma tecnologia nova em uma área que ainda não possui muitas referências téoricas ou práticas. Seu desenvolvimento foi desafiador e exigiu bastante dos envolvidos. Utilizar o conceito de *blockchains* para armazenamento distribuído parece ser uma tendância não somente em ambientes privados, bem como em instituições governamentais ou acadêmicas.

A API BlockSigner está em sua versão inicial e seu desenvolvimento está em constante evolução. Os próximos passos da aplicação giram em torno da escalabilidade e usabilidade.

Tornar a aplicação intrinsicamente *peer-to-peer* com o suporte do Chord [Stoica et al. 2001] será uma das próximas diretivas do projeto. Na versão atual, sempre que um bloco é gerado e validado, ele é transmitido para todos os outros nós da rede, porém via requisições HTTP. A utilização do Chord transformaria a estrutura de comunicação entre os nós da API de NxN (todos para todos) para o logaritmo do número de nós. Os resultados finais de avaliação do Chord se mostram altamente escaláveis, com baixo custo de comunicação e a manutenção do seu estado feita por cada nó. Esta alteração tornaria a aplicação mais escalável, permitindo seu uso não somente em redes com um número de nós limitados.

Outra possível melhoria, seria não apenas armazenar o hash do documento, mas também guardar de forma distribuída o arquivo do documento. Essa implementação pode ser otimizada com o uso do IPFS (Inter Planetary File System), sistema descentralizado web para armazenamento de dados [Benet 2014]. Similar a maneira como o BitTorrent troca seus dados pela internet, o IPFS se mostra independente de um servidor centralizado para requisições. Uma internet construída de forma totalmente descentralizada tem o potencial de acelerar a transferência e a transmissão de dados. Tal melhora não é só conveniente como necessária, uma vez que os sistemas web de entrega de conteúdo atualmente estão sobrecarregados.

Mesmo diante de algumas possíveis melhorias, Blocksigner se mostrou suficientemente madura para a experimentação preliminar. A segurança e validade dos arquivos aqui salvos são garantidos, bem como sua estabilidade em conjunto de grandes servidores utilizando suas funções.

Referências

Ben-Menachem, M. (2005). Review of "testing embedded software by bart broekman and edwin notenboom"; addison wesley.; 2003. *SIGSOFT Softw. Eng. Notes*, 30(2):30–30.

Benet, J. (2014). Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561*.

Cohen, R. J., Forsberg, R. A., Kallfelz Jr, P. A., Meckstroth, J. R., Pascoe, C. J., and Snow-Weaver, A. L. (2001). Coordinating user target logons in a single sign-on (sso) environment. US Patent 6,178,511.

Croman, K., Decker, C., Eyal, I., Gencer, A. E., Juels, A., Kosba, A., Miller, A., Saxena, P., Shi, E., Sirer, E. G., et al. (2016). On Scaling Decentralized Blockchains. In *International Conference on Financial Cryptography and Data Security*, pages 106–125. Springer.

Greve, F., Sampaio, L., Abijaude, J., Coutinho, A., Valcy, Í., and Queiroz, S. Blockchain e a revolução do consenso sob demanda.

Helsley, M. (2009). Lxc: Linux container tools. *IBM devloperWorks Technical Library*, 11.

JR.2015, E. www.originalmy.com.

Mahajan, P. and Sachdeva, A. (2013). A study of encryption algorithms aes, des and rsa for security. *Global Journal of Computer Science and Technology*.

Murilo B. Flor, Gabriel Marques, B. J. (2018). Blocksigner GitLab Repository. Online; accessed 2018-11-10.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

Oliveira, R. R. (2012). Criptografia simétrica e assimétrica-os principais algoritmos de cifragem. *Segurança Digital [Revista online]*, 31:11–15.

Rubin, F. (1996). One-time pad cryptography. *Cryptologia*, 20(4):359–364.

Singh, G. (2013). A study of encryption algorithms (rsa, des, 3des and aes) for information security. *International Journal of Computer Applications*, 67(19).

Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., and Balakrishnan, H. (2001). Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication Review*, 31(4):149–160.

Thakur, J. and Kumar, N. (2011). Des, aes and blowfish: Symmetric key cryptography algorithms simulation based performance analysis. *International journal of emerging technology and advanced engineering*, 1(2):6–12.

Wood, G. (2014). Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151:1–32.