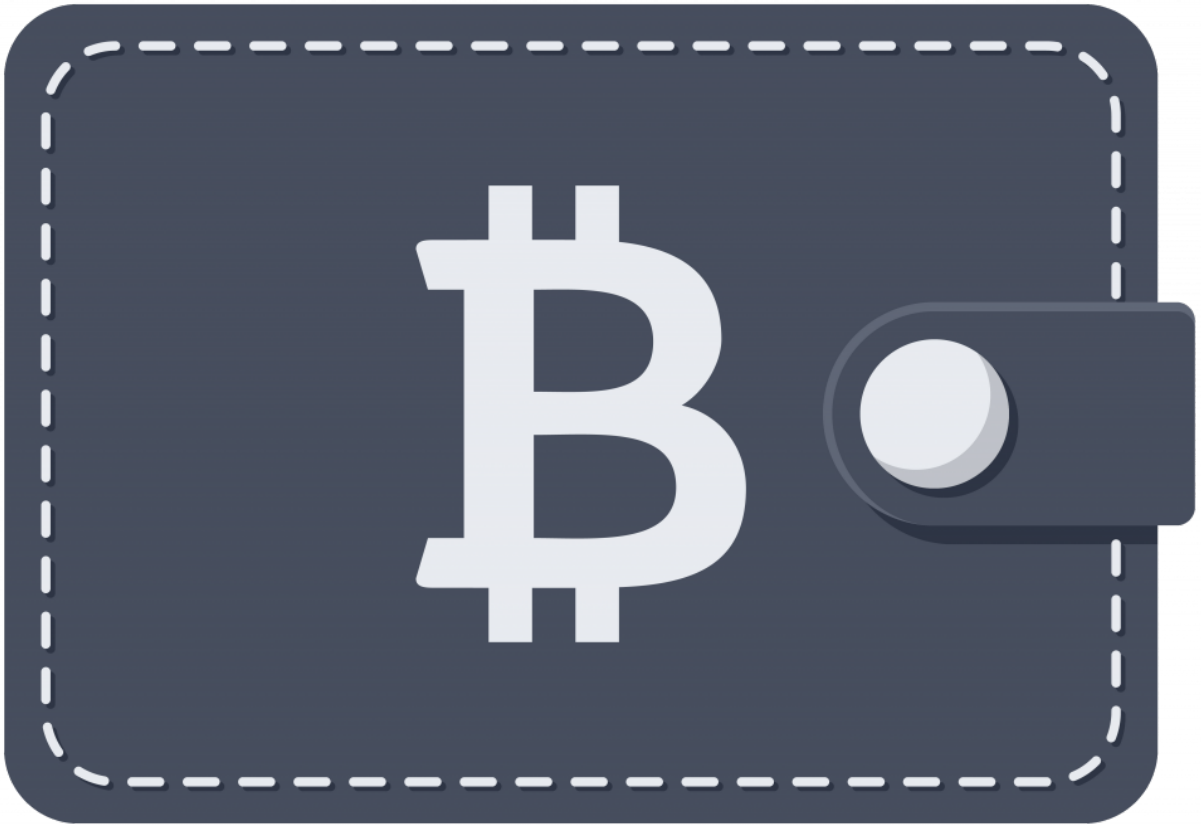
# Bitcoin: A Peer-to-Peer Electronic Cash System



# Para Iniciantes

Escrito por Daniel Marques | Descentralizado| Page Facebook
Revisado por Bruno Neto | Eu Sou Rye | Page Facebook

# AGRADECIMENTOS

Quero agradecer as pessoas que contribuíram diretamente ou indiretamente para que esse projeto se concluísse, são eles:

| | |
|---|---|
| Paula Berman | Democracy.Earth |
| Bruno Neto | Eu Sou Rye |
| Raul Folks | WASS |
| Alexandre de Souza | Bitcoin Brasil |
| Rafael Capaci | Jupiter Coworking Space |


# DÚVIDAS

Qualquer dúvida sobre algum trecho do documento abaixo, ou até sugestões, por favor entre em contato pela site www.descentralizado.com.br ou me envie no email suporte@descentralizado.com.br

Tenham uma boa leitura, esse documento foi feito com muita dedicação, então deixo aqui minha aprovação, para compartilhar ou reproduzi-lo.

como sempre digo:

*“CONHECIMENTO É PRA SER COMPARTILHADO E NÃO COBRADO”*

# INTRODUÇÃO

O Bitcoin nasceu em 2008, criado por Satoshi Nakamoto, um pseudônimo incógnito e de paradeiro desconhecido que sequer poderia ser definido como um grupo ou um único indivíduo.

A primeira transação envolvendo a moeda digital aconteceu em 03 de janeiro de 2009, supostamente realizada por Nakamoto, que até então havia apenas apresentado a proposta inicial do Bitcoin, tratando-se de um white paper batizado como: Bitcoin: a peer-to-peer electronic cash system[1]

Atualmente os sistemas de pagamento online são totalmente dependentes de instituições financeiras, mostrando a total falta de confiança que permeia – com razão – o âmbito das transações digitais. Sendo assim, ao realizarmos alguma transação, algumas dessas instituições assumirão a responsabilidade de auditar as informações intrínsecas à transação em questão, funcionando como um intermediário obrigatório para o funcionamento desse sistema.

Outro ponto importante a ressaltar é em relação à vulnerabilidade destas informações, que, devido à natureza centralizadora do funcionamento do sistema, faz com que as instituições auditoras fiquem à mercê de fraudes e de invasões.

Outro ponto se trata da arbitrariedade que circunda esse tipo de instituição, que para diminuir o volume de transações e de trabalho – consequentemente, de custo –, acabam por limitar o valor mínimo para a execução das transações. Afinal, toda validação e/ ou auditoria possui um custo mínimo inerente a sua manutenção, o que torna micro transações operações demasiadamente custosas para essas instituições. Isto significa que a necessidade de recorrentes auditorias para impedir gastos duplos simplesmente inviabiliza quaisquer transações que se mostrem custosas para o sistema, o que implica na inviabilização de micro transações. Esse dado reflete na divisibilidade da moeda fiduciária Brasileira que impede transações superiores a duas casas decimais.

Entendendo a principal função das instituições financeiras, Nakamoto propôs uma rede de pagamentos [2]P2P, onde todos os pagamentos seriam enfileirados, numerados, datados e inseridos em uma [3]cadeia de blocos. Em seguida, todas as informações seriam criptografadas usando hash e validadas por um mecanismo chamado [4]PoW. Desta forma, esse sistema dispensaria quaisquer intermediários, pois a criptografia que os substituiria impossibilitaria quaisquer adulterações em algum dos dados sem que esta corrupção comprometesse alguma das informações e/ ou alguma das identidades que incidirem no suposto dado adulterado.

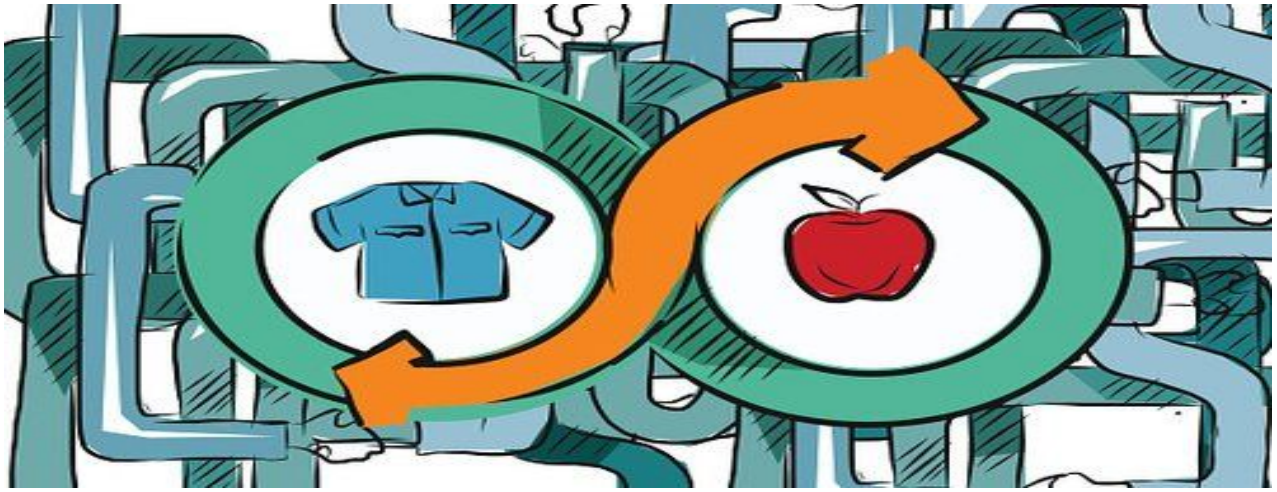
# DINHEIRO

Para entendermos como funcionam as transações, resolvi primeiramente explicar como funciona o sistema de dinheiro para que assim, caro leitor, você possa ter a capacidade de analisar a dimensão do avanço tecnológico e democrático, que essa tecnologia nos proporciona.

> *"O dinheiro é, consequentemente, um sistema de confiança mútua, e não só isso, o dinheiro é o mais universal e mais eficiente sistema de confiança mútua já inventada"*
>
> \- Yuval Harari

O dinheiro deve seguir algumas características padrões:

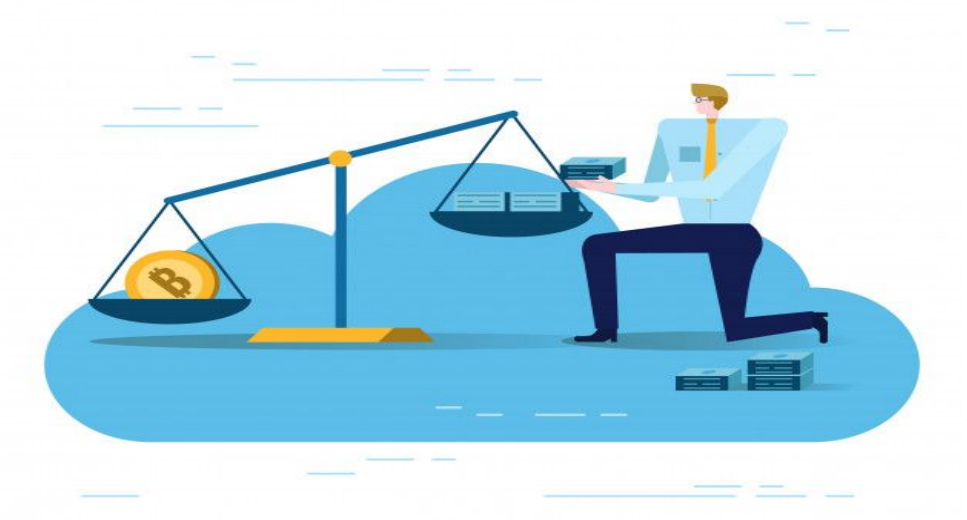
## 1. Deve ser um meio de troca



Para realizar essa função, o ativo deve ser desejado por ambas às partes envolvidas, mas isso nos leva a uma pergunta, porque aceitamos metais, moedas (moeda fiat), ouro, ou qualquer outro objeto como meio de pagamento? A resposta fica fácil de supor, pois sempre soubemos, mas nunca confrontamos a nossa mente para analisar o dinheiro como um bem de troca, a nossa resposta para essa pergunta é, aceitamos ouro ou cédulas de papel como pagamento porque esse ativo garante que ele será aceito em um momento futuro ao atual negociado, onde podemos trocar esse ativo, por outros objetos.

Quando vamos a uma cafeteria e compramos um café por cinco reais, na verdade estamos trocando essa cédula de R$5,00 por uma xícara de café, o comerciante está aceitando a cédula, pois ele acredita que essa mesma cédula poderá ser trocada futuramente por mais pó de café, a fim de ele mesmo continuar o ciclo de vendas de café.

## 2. Deve ser uma Unidade Contábil



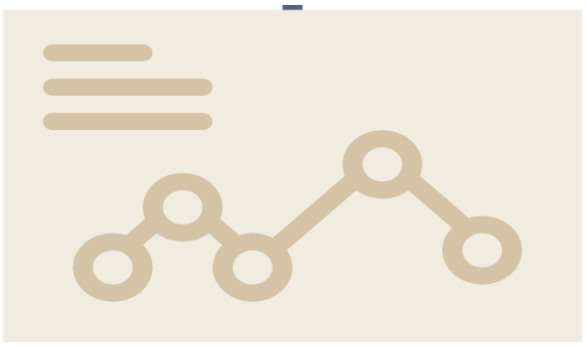
Uma unidade contábil são todos os bens utilizados para expressar débitos e especificar o valor de outros bens, uma moeda como o Real é considerada uma unidade contábil, pois podemos a partir dela medir e dimensionar o valor de um carro, por exemplo, ou dimensionar o valor de uma dívida, estão quando podemos usar esse ativo para dimensionar outros bens, ele são chamados de Unidade Contábil.

## 3. Deve servir para acumular valores

O ativo deve ser capaz de ser acumulado, ou seja, podemos guardar e acumular esse ativo para que em um momento posterior possamos usa-lo. Mas cuidado, nem todos os ativos que tem essas características podemos ser considerados dinheiro.

## 4. Ter o Valor Estável

Ter um valor estável e que não tenha muitas variações no seu valor, permite que cresça uma confiança mútua em aceita-lo e repassá-lo, pois não seria interessante aceitar um ativo como pagamento sabendo que o mesmo, amanhã ou depois, poderá não valer o mesmo valor pelo que foi negociado. O nossa moeda o Real possui um valor estável e é controlado pelo governo.



## 5. Ser de Difícil Falsificação


Antigamente o dinheiro era a representação da assinatura e reputação do Rei, e falsificá-lo era um ato altamente desprezado por seu rei, pois o falsificador estaria manchando e sujando sua [5]reputação. Com base nesse mandamento o rei estabelecia a confiança em sua moeda de troca.

Mas nós dias atuais sabemos que existem diversas outros meio de falsificação, então os Bancos Centrais tentam ao máximo impedir qualquer forma de falsificação das cédulas papel impresso, inserindo marcas d'água, Número de Controle, mudando o tipo de papel a ser impresso, e prendendo os falsificadores como um crime de alta penalidade, pois como na época dos reis, ao falsificar uma nota você está diretamente, falsificando a voz da confiança sujando sua reputação.

## 6. Deve ter um valor padronizado e reprodutível

Duas representações de dinheiro devem ser idênticas, caso refiram-se ao mesmo valor. Toda cédula de dez reais respeita uma padronização, a minha cédula de dez reais e a sua nota, ambas visualmente são iguais, e devem ser iguais sempre que elas representarem o mesmo valor.

## 7. Privacidade

A privacidade e confiança nunca andaram tão juntas como no dinheiro, eu posso trocar meu dinheiro por um copo de suco, e pra isso eu nunca irei precisar me apresentar a parte que receberá esse dinheiro, não preciso conhecer o recebedor, Quantas vezes você já se perguntou, qual é o nome do atendente do mercado? Muitas vezes não sabemos, mas não se sinta mal por isso, ou pense que é falta de respeito, devemos apenas lembrar que uma das propriedades do dinheiro está em ação, desconhecer  o recebedor não implica em realizar um troca que ambas as partes saiam de acordo.

Essa confiança mútua é o que mantém de pé o sistema. Mas que até então nunca foi simulada no meio eletrônico, pois a maior pergunta que tínhamos é como poderíamos sem nenhum dado adicional do usuário ou uma terceira parte confiável, ter a certeza que esse dinheiro é real e será aceito em uma troca futura. A confiança é mantida apenas pelos membros dessa transação, mas ela só está acontecendo porque em um futuro próximo, esses mesmos usuários, terão a certeza que os demais usuários honraram com essa mesma confiança mútua.

## 8. Conclusão

O dinheiro é a representação de confiança mútua com algumas características que devem ser respeitadas, ele tem como principal função ser um ativo de interesse coletivo, para que o mesmo possa ser aceito como moeda de troca e permutado por outros bens e produtos.

# BLOCKCHAIN

Blockchain é um conceito, uma arquitetura banco de dados, sendo assim, não existe uma única arquitetura de dados, existem vários, o mais conhecido é o Blockchain do Bitcoin.

Blockchain como o nome diz na tradução literal, (Block = Bloco) + (Chain = Cadeia) = Cadeia de Blocos, esse protocolo é chamado assim, por registrar em um livro razão todas as transações, eu particularmente gosto de comparar o Blockchain com um grande livro em que a cada página escrita desse livro é feito uma assinatura única, e a pagina posterior irá conter uma assinatura da página anterior e assinatura da página atual.

*“Uma maneira para um usuário da Internet transferir um valor único de propriedade digital para outro usuário da Internet, de modo que a transferência seja garantida para ser fácil e segura, todos sabem que a transferência ocorreu, e ninguém pode duvidar da legitimidade da transferência. As consequências desse avanço são difíceis de exagerar “.*

– Marc Andreessen

## 1. Tipos de Blockchain

- **Publica:**
  Qualquer pessoa pode ler e enviar transações, por exemplo Bitcoin, Ethereum.

- **Privada:**
  Uma organização determina o que cada usuário pode ou não fazer na rede blockchain deles. Por exemplo um Bankchain.

## 2. Propriedades

Para ser considerado um blockchain ele deve respeitar algumas características básicas:

- **Prevenção Gasto Duplo:**
  As moedas (Tokens) enviados não devem permanecer na carteira do usuário que os enviou.

- **Anonimato:**
  Deve-se ser anônimo, nunca deve revelar o seu verdadeiro dono.

- **Transferibilidade:**
  Deve-se transferir o menor valor possível, no exemplo do Bitcoin é possível enviar 0,000000001 isso mesmo 9 casas decimais.

- **Escalabilidade:**
  O Blockchain deve ser escalável, ou seja, quanto mais pessoas entrar na rede, mais ela irá crescer sem comprometer o funcionamento dos demais usuário que já estão na rede.

- **Divisibilidade:**
  O Token deve permitir o maior número possível de divisão, um exemplo clássico é o Bitcoin como foi abordado, em que o mesmo possui 9 casas decimais, no caso dos bancos, quando você irá dividir um valor, você só pode dividir e aceitar até duas casas decimais os valores posteriores a isso você perde.

- **Independência de Hardware:**
  Qualquer usuário pode acessar a rede, independente se o Blockchain é para desktop, Celulares, notebook, tablet, computadores potentes e até os computadores pessoais, isso mesmo aquele computadores velho que você pensou que não seria útil, deve funcionar nele também.

- **Chaves públicas não devem revelar identidades reais:**
  No Blockchain todos possuem uma chave pública, essa chave nunca deve revelar seu verdadeiro dono, ela não deve ser vinculada a números pessoais, Documentos, Foto, Telefone, nada que comprometa ou facilite a identificação do seu verdadeiro dono.

- **Chaves públicas não devem revelar identidades reais :**

No Blockchain todos possuem uma chave pública, essa chave nunca deve revelar seu verdadeiro dono, ela não deve ser vinculada a números pessoais, Documentos, Foto, Telefone, nada que comprometa ou facilite a identificação do seu verdadeiro dono.

## TRANSAÇÕES

Uma transação pode ser resumida pelo ato de transferir valores. No caso do Bitcoin, todas as transações recebem um carimbo de identificação baseado em hashing, assim, para uma transação ser validada, se deve, respeitar as seguintes características:

- **Chave Pública do Destinatário**
- **Chave Pública do Remetente**
- **Data e Hora**
- **Hashing da Transação Anterior**
- **Hashing do Bloco Pertencente**
- **Valor**

O acúmulo de transações na rede em um único local é chamado de blocos e ele possui as suas próprias características para ser um bloco válido, sendo obrigado a ter as seguintes informações: Identificação do bloco, [6]Hashing do bloco anterior, Data/Hora, identificação do minerador, recompensa pelo bloco, tamanho do bloco.

Dominando o conceito de 'bloco', podemos entender o processo por trás de qualquer transação dentro da rede Bitcoin. Basicamente, o procedimento se inicia quando o proprietário transfere o valor desejado para outro membro da rede. Ao realizar a transferência, surge o primeiro problema: a terceira parte verificadora não existe, ou seja, o usuário que recebeu o valor não pode verificar se o valor recebido é fruto de gasto duplo.

Eis que entram em cena os mineradores. Sua principal função é validar quaisquer transações que permeiam o 'bloco', a fim de garantir que os mesmos Bitcoins enviados naquele momento não sejam enviados novamente na posteridade, o que caracterizaria um gasto duplo. Vamos simular uma transação posteriormente para explicar todo o processo por trás de cada transação.

Atualmente existem dois tipos possíveis de transações, sendo:

- **Transações Finais (Somente um [7]Output)**
- **Transações com Troco (Com mais de um Output)**

## 1. Transações Finais

São todas as transações que possuem somente um único output , entendendo isso podemos afirmar que essa transação ou ela está consumindo todo o saldo desta carteira ou está enviando um valor que uma única transação anterior conseguiria liquida-la. Para ilustrar a definição, vamos imaginar um mercado, um cliente está finalizando uma compra e fechou em 100 reais o valor de todos os itens, temos dois tipos de situações e podemos ter a certeza que uma dessas situações irá acontecer:

- Cliente tem uma nota de R$100,00 reais na carteira, ao entregar a nota ao atendente não receberá troco dessa transação, pois o valor pago foi o mesmo do valor solicitado. Ou seja, Uma entrada e uma saída.

- Cliente tem notas diversas como, por exemplo, nota de R$10,00 reais, R$50,00 reais e duas notas de R$20,00 reais na carteira, totalizando os mesmos R$100,00 reais. Esse processo possuem muitas entradas e somente uma saída.

**Modelo de Transação**

|*Origem*|---|*Destino*|----|*Valor*|---|*IDTransacao*|--|*DataHora*|-----|*IDBloco*|-----|*Status*|

## Exemplo:

21e76fac1b1bb4a435075fd637aae2f1edd180dd929bcd8c7ce996153ab8e291

17L2PEt3gy9CSoXoKJWe6UCBXag8tq7q8S ➡ 1NPHasohEGvm6jAivKvmSRNV2sMY3ZqW52

## 2. Transações com Troco

São transações que possuem mais de um output, neste caso o invés da transação gerar somente um output, ele gera duas, a primeira em nome do destinatário com o seu devido valor, e a segunda saída em nome do próprio remetente do montante.

O saldo em uma carteira Bitcoin consiste em pegarmos a soma de todas as transações de entrada, menos as transações de saída. Tem como fórmula Saldo = Entradas – Saídas.

Você deve estar se perguntando, não existe Bitcoin? Irá parecer estranho, mas o saldo que você tem em Bitcoin é na verdade produto da diferença entre muitas outras transações de entradas e saídas para esse endereço. O endereço Bitcoin é realmente validado e visível, somente após esse endereço receber uma transação de entrada, pois é com a chave privada que ele irá reivindicar ser o beneficiário dessa transação e portanto receber esse montante solicitado.

Para entendermos como uma transação de troco funciona, vou tentar explica-lo usando o mesmo exemplo do mercado, mas agora aplicada a essa realidade.

Imaginamos um cliente que faz uma compra em um mercado, a sua compra totalizou R$50,00 reais, mas o cliente só tinha uma nota de R$100,00 reais, o que irá acontecer agora? Com certeza o atendente do caixa irá devolver uma nota de R$50 reais, pois a compra já foi paga e mesmo assim sobrou um “resto” do valor da nota entregue.

Esse mecanismo também é simulado na rede do Bitcoin em suas transações, como vimos uma transação na rede só poderá ocorrer com referência de uma transação anterior a ela, ou seja, na verdade você não possui Bitcoins, você possui transações, e é com elas que você pode enviar Bitcoin, as transações são como as cédulas de papel moeda, se pagarmos uma compra com uma nota superior ao valor total dessa compra, você receberá o troco por essa cédula, a diferença e que na rede Bitcoin, para enviar um pagamento o sistema irá agregar uma ou mais transações de entrada até que chegue no valor solicitado, e caso essa soma ultrapasse o saldo de envio, será realizado uma transação de troco para o remetente.

**Modelo de Transação**

*Output (Transação para o destinatário):*
|Origem|-----|Destino|------|Valor|-------|IDTransacao|--|DataHora|-----|IDBloco|-----|Status|

*Input (Transação para o remetente "Troco"):*
|Origem|-----|Destino|------|Valor|-------|IDTransacao|--|DataHora|-----|IDBloco|-----|Status|

## Exemplo:


## 3. Porque isso ocorre?

Um importante objetivo com esse processo é copiar uma das maiores características do Dinheiro que é evitar o Gasto Duplo, seguindo esse pensamento Satoshi resolveu de forma muito simples o problema, pois a única coisa que o sistema precisa fazer e verificar todas as transações que já saíram e entraram e com isso teremos o nosso saldo.

Com este mecanismo evitamos gasto duplo e todos os usuários da rede são assegurados que o mesmo valor enviado pelo remetente não será enviado para outro destinatário. O sistema sempre reconhece a transação mais antiga como válida, ou seja, a transação que foi minerada primeira é a correta. Para garantirmos que sempre teremos todos os blocos na sequência correta, os usuários que deixarem a rede ou ficarem off-line, deverão realizar a sincronização dos blocos que estão faltando, para que somente depois disso possam enviar as transações que ele fez enquanto estava offline, o usuário que contém uma carteira com todos os blocos dessa rede, geralmente é chamado de [8]"Nó".

Ao se realizar uma transação, será enviada aos mineradores da rede, os responsáveis pelo fechamento do bloco, que se encarrega de validar as transações e de inseri-las no blockchain. Assim, posteriormente permitindo que todos os usuários da rede tenham acesso as informações intrínsecas aos blocos ali inseridos.

## SERVIDOR TIMESTAMP

No blockchain do Bitcoin, existe a necessidade de um servidor ‘data-hora’ para se manter as transações em ordem cronológica, através do registro do momento em que ocorreram.

Esse servidor timestamp se trata da mediana da hora de todos os nós conectados a ele. Para o sistema Bitcoin verificar qual é a hora correta no momento de assinar alguma transação, um timestamp é aceito como válido se for maior que o timestamp mediano dos 11 blocos anteriores e menor que o tempo ajustado na rede + 2 horas. "Tempo ajustado na rede" é a mediana dos carimbos de data / hora retornados por todos os nós conectados a você.

O timestamp prova que os dados foram gerados em algum momento e carrega uma referência dos hashes anteriores. Estes (hashes) são a identificação existente entre os blocos válidos, que, devido a sua natureza, são criptograficamente imutáveis, afinal, qualquer mudança em alguma de suas informações forçará a alteração desta identificação, o que implica na mudança do hash e invalida o bloco, catalisando o reinicio do processo de aquisição do bloco correto. Por fim, resultando no impedimento de ataques de timestamp que poderiam forjar um gasto duplo.

Os ataques de timestamp são realizados para enganar o sistema em função da sua hora, afinal, ao se alterar o horário da transação, tecnicamente se poderia alterar os eventos ocorridos no passado, possibilitando o atacante de manipular as informações de uma transação já processada; ou colocar a transação adulterada em algum outro bloco, confundindo o sistema. O servidor timestamp tem a função de garantir que os ataques de adulteração de transações por meio de modificação de data/hora não ocorram.

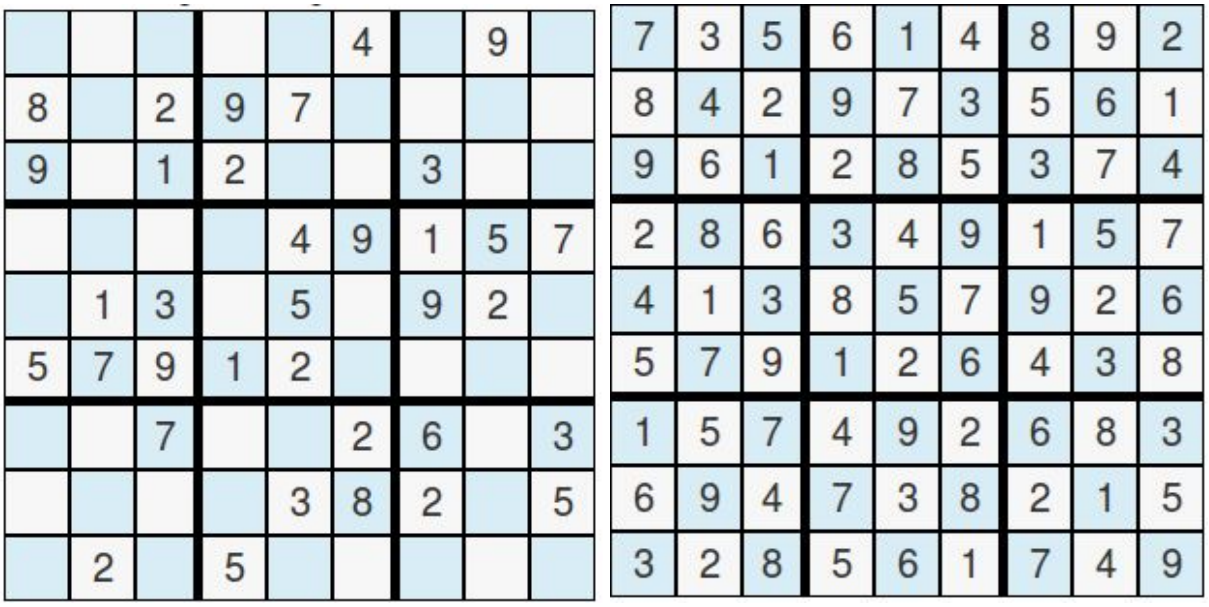
## PROVA DE TRABALHO

Para implementar um servidor de timestamp distribuído em uma base peer-to-peer é necessário um algoritmo que garanta que a prova de trabalho seja refeita sempre que uma nova validação seja requisitada.

A definição de proof-of-work é uma corrida em que todos os mineradores da rede competem entre si para resolver um problema matemático muito complexo, cuja dificuldade se altera a cada quatro anos. O minerador que primeiro resolver o problema terá o direito de validar todas as transações e ficar com a recompensa de todas as taxas.

Esse processo tem como objetivo gerar novos Bitcoins e recompensar o minerador por ceder poder computacional à rede, ocorrendo até que se atinja o limite de 21 milhões de Bitcoins. Chegando ao limite de moedas, não gerar-se-ão novas moedas e o minerador ficará apenas com as taxas pagas pelos usuários.

Vamos Imaginar que o problema matemático é um jogo de Sudoku, a dificuldade é controlado pelo número de quadrados a resolver, onde no algoritmo é representado pelo “0” inicial, podemos ver que o desafio é difícil de resolver, mas é extremamente fácil de verificar se está correta.



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | 4 | | 9 | |
| 8 | | 2 | 9 | 7 | | | | |
| 9 | | 1 | 2 | | | 3 | | |
| | | | | 4 | 9 | 1 | 5 | 7 |
| | 1 | 3 | | 5 | | 9 | 2 | |
| 5 | 7 | 9 | 1 | 2 | | | | |
| | | 7 | | | 2 | 6 | | 3 |
| | | | | 3 | 8 | 2 | | 5 |
| | 2 | | 5 | | | | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 7 | 3 | 5 | 6 | 1 | 4 | 8 | 9 | 2 |
| 8 | 4 | 2 | 9 | 7 | 3 | 5 | 6 | 1 |
| 9 | 6 | 1 | 2 | 8 | 5 | 3 | 7 | 4 |
| 2 | 8 | 6 | 3 | 4 | 9 | 1 | 5 | 7 |
| 4 | 1 | 3 | 8 | 5 | 7 | 9 | 2 | 6 |
| 5 | 7 | 9 | 1 | 2 | 6 | 4 | 3 | 8 |
| 1 | 5 | 7 | 4 | 9 | 2 | 6 | 8 | 3 |
| 6 | 9 | 4 | 7 | 3 | 8 | 2 | 1 | 5 |
| 3 | 2 | 8 | 5 | 6 | 1 | 7 | 4 | 9 |

Digamos que a seqüência de base em que vamos trabalhar é "Olá, mundo!". Nosso objetivo é encontrar uma variação do que [9]SHA-256 que é um valor que começa com '000'. Para isso nós adicionamos um nonce e aumentamos a cada nova tentativa o parâmetro de entrada. Encontrando uma partida para "Olá, mundo!" nos leva 4251 tentativas (mas que inicie com zeros nos três primeiros dígitos):

**"Olá, mundo!" + “0” => 1312af178c253f84028d480a6adc1e25e81caa44c749ec81976192e2ec934c64**

**"Olá, mundo!" + “1” => e9afc424b79e4f6ab42d99c81156d3a17228d6e1eef4139be78e948a9332a7d8**

**"Olá, mundo!" + “2” => ae37343a357a8297591625e7134cbea22f5928be8ca2a32aa475cf05fd4266b7**

...

**"Olá, mundo!" + “4248” => 6e110d98b388e77e9c6f042ac6b497cec46660deef75a55ebc7cfdf65cc0b965**

**"Olá, mundo!" + “4249” => 004190b822f1669cac8dc37e761cb73652e7832fb814565702245cf26ebb9e78**

**"Olá,mundo!" + “4250” => 000c3af42fc31103f1fdc0151fa747ff87349a4714df7cc52ea464e12dcd4e9sed23**

**0000c3af42fc31103f1fdc0151fa747ff87349a4714df7cc52ea464e12dcd4e9sed23 ←---------- VENCEDOR**

Este hash está de acordo com o desafio, mas como testar se ele é a reposta correta para o problema?  Simples. Realiza-se a geração do hash, utilizando dos mesmos PARÂMETROS DE ENTRADA usados, afinal, o desafio está a buscar a palavra que foi usada para chegar ao hash fornecido pelo sistema. Como sabemos, o algoritmo Hash não permite fazer o processo reverso, isto é, saber – a partir dele – a palavra que a gerou. Logo, esse processo força a utilização de processamento computacional para descobrir a palavra, tentando uma a uma até que alguém a descubra.

Uma vez que o bloco seja validado e satisfeito o problema, isto é, quando o minerador encontrar a resposta para o enigma do desafio, o bloco não poderá ser alterado. Caso o seja, dever-se-á calcular o bloco de todas as transações posteriores a ele, considerando que um parâmetro para criar o cabeçalho do bloco é o hash do bloco anterior.
Existem dois tipos de ataques que podem ser causadas nessa parte:

- **Attack Real Time**
- **Attack Block Prev**

Depois de realizado este desafio, o minerador irá validar as transações e assinar o novo bloco. Posteriormente, o bloco será julgado para garantir o bom trabalho do minerador e a não criação de transações fantasmas ou adulteração de alguma transação. Esse tipo de coisa (criação de transações fantasmas e adulteração de transações) é possível, pois todos os ‘nós’ possuem uma cópia de todas as transações.

Os mineradores expressam aceitação de um novo bloco, no momento que os mesmos usarem a identificação do bloco gerado, para resolver um novo problema matemático, e assim reiniciando a competição.

A prova de trabalho também resolve o problema de determinar a representação na tomada de decisão maioritária. Se a maioria fosse baseada em endereço IP-um-voto, a votação poderia ser subvertida por qualquer pessoa que fosse capaz de alocar muitos endereços IP. Por isso, a prova de trabalho é essencialmente CPU-um-voto. A decisão maioritária é representada pela cadeia mais longa, que tem o maior esforço de prova de trabalho investido nele. Para definir se uma transação é válida ou não, deve-se seguir a seguinte linha de pensamento:

[O atacante] + [CPU] > [Questionadores] + [CPU]

Isto é, o atacante deve ter mais poder computacional que toda a rede, incluindo aqueles que são contra o bloco fornecido por ele, somente assim, seu bloco modificado seria aceito. Concluindo, o atacante deverá apresentar 51% do poder computacional total da rede para que a sua falsificação seja validada.

Obs. Com o tempo, os hardwares tendem a ficar obsoletos, assim, para compensar o aumento da velocidade do hardware e o interesse variável em executar os nós, a dificuldade de prova de trabalho varia periodicamente através da média de blocos gerados por hora. Caso sejam gerados muito rápido, a dificuldade aumenta.

### 1. Attack Real Time

Neste caso, o atacante tenta minerar, a fim de gerar um bloco falso. Porém, o atacante deve apresentar um poder computacional gigantesco. Considerando que o Bitcoin é uma pirâmide invertida, quanto maior for a quantidade de mineradores, maior será a quantidade de processamento necessário padrão + ajuste da taxa em relação ao processamento médio. Quantidade de processamento necessário padrão é a média de todo o processamento da rede, ou seja, você não é proibido de minerar, mas se você não tiver pelo menos a média, será quase impossível a mineração solo e o ajuste da taxa em relação ao processamento médio é a taxa de dificuldade que o algoritmo coloca no desafio a fim de controlar a emissão de um novo bloco a cada 10 minutos.

### 2. Attack Block Prev

O atacante tenta forjar um bloco falso, retirando um bloco valido da cadeia de blocos e irá alterar o bloco. Esse processo tem um problema também, pois para esse bloco ser aceito ele terá que refazer a prova de trabalho de todos os blocos posteriores a ele, pois os hash prev de cada bloco estão interligados entre si. caso esse processo seja feito o atacante terá que pediu um consenso do bloco a fim de enviar a todos. Mas lembre de que por ser um TIMESTAMP HASH no cabeçalho do bloco o atacante não sabe a hora certa que o bloco foi gerado e sua prova de comparação de DATETIME com a rede estará divergente fazendo que seus blocos sejam invalidados sempre, o mesmo terá que baixar todos os blocos do consenso da rede já validados para que volte a fazer as transações.

## REDE

A rede possui algumas características que devem ser respeitadas:

1. Novas transações são transmitidas para todos os ‘nós’;
2. Cada ‘nó’ coleta as novas transações em um bloco;
3. Cada ‘nó’ trabalha para encontrar uma prova de trabalho difícil para o seu bloco;
4. Quando um ‘nó’ encontra uma prova de trabalho, ele transmite o bloco para todos os ‘nós’;
5. [8]‘Nós’ aceitam o bloco apenas se todas as transações nele são válidas e não houve gasto-duplo; e
6. ‘Nós’ expressam sua aceitação ao bloco ao começar a trabalhar na criação do próximo bloco da corrente, usando a codificação do bloco aceito como a codificação anterior.

A rede sempre irá aceitar a cadeia mais longa como verdadeira, porém – como visto –, para poder validar as transações, os mineradores devem resolver um problema matemático. Devido à natureza da competição, pode acontecer de dois mineradores encontrarem a mesma solução ao mesmo tempo.

É neste momento que eles escolhem o bloco mais antigo, ou seja, devem escolher um dos blocos que foi construído pelos vencedores como parametro para reiniciar a competição, quando um dos mineradores encontrar o próximo bloco, então todos usaram esse bloco novo como parâmetro anterior e não-mais o bloco que subdividiu a rede.

## INCENTIVO

Os incentivos que o sistema oferece em troca do poder computacional e energia fornecida pelos mineradores derivam das taxas das transações presentes no bloco e da recompensa pelo bloco minerado.

Esse incentivo tem como função disseminar a circulação de novas moedas no sistema, de onde deriva a emissão de Bitcoins.

Quando a geração de novas moedas atingir a quantidade limite de 21 milhões, os mineradores ficarão somente com as taxas das transações presentes nos blocos. As taxas passarão a ser o único incentivo dos mineradores para continuarem fornecendo poder computacional para a rede.

Um ponto interessante em relação a esse incentivo é: pode-se realizar uma transação com 0.000000000 de taxa, porém, será muito difícil confirmá-la, afinal, são os mineradores quem determinam quais transações serão selecionadas para integrar o bloco – de 1mb – e que estes escolherão as transações mais lucrativas (i.e. com as maiores taxas).

O incentivo motiva os nós a permanecerem honestos. Se um atacante ganancioso for capaz de montar mais poder de CPU do que todos os nós honestos, ele terá que escolher entre danificar a rede, o que implicaria em desvalorização da moeda, e portanto, perderá todo seu investimento em hardware, ou usar-lo como um ‘nó’ honesto a fim de se beneficiar com as recompensas oferecidas pelo sistema.

## ESPAÇO EM DISCO

O espaço em disco não é um problema para os usuários que não irão rodar um nó completo em sua máquina, mas esse possível problema que seria a economia de espaço em disco, também foi previsto na sua criação, visto que em 2008 não tínhamos acesso a 1TB (um terabyte) de HD (hard Disk) como temos hoje.

Para um usuário rodar um nó completo da rede Bitcoin, este deverá disponibilizar aproximadamente 180gb (gigabyte) de espaço em disco. Pode parecer muito, mas lembre-se de que o blockchain é uma cadeia de blocos e todos da rede devem acessar um nó completo para poder realizar transações.

As transações são armazenadas em hash de merkle tree. Markle tree se trata de um padrão de organização de blocos hash, onde pequenas informações representam uma vasta

quantidade de dados. Essa organização permite reduzir drasticamente o tamanho de espaço em disco ocupado, pois se salva apenas o cabeçalho do bloco e não o bloco com todas as transações. Este cabeçalho de bloco sem transações tem aproximadamente 80b (bytes).

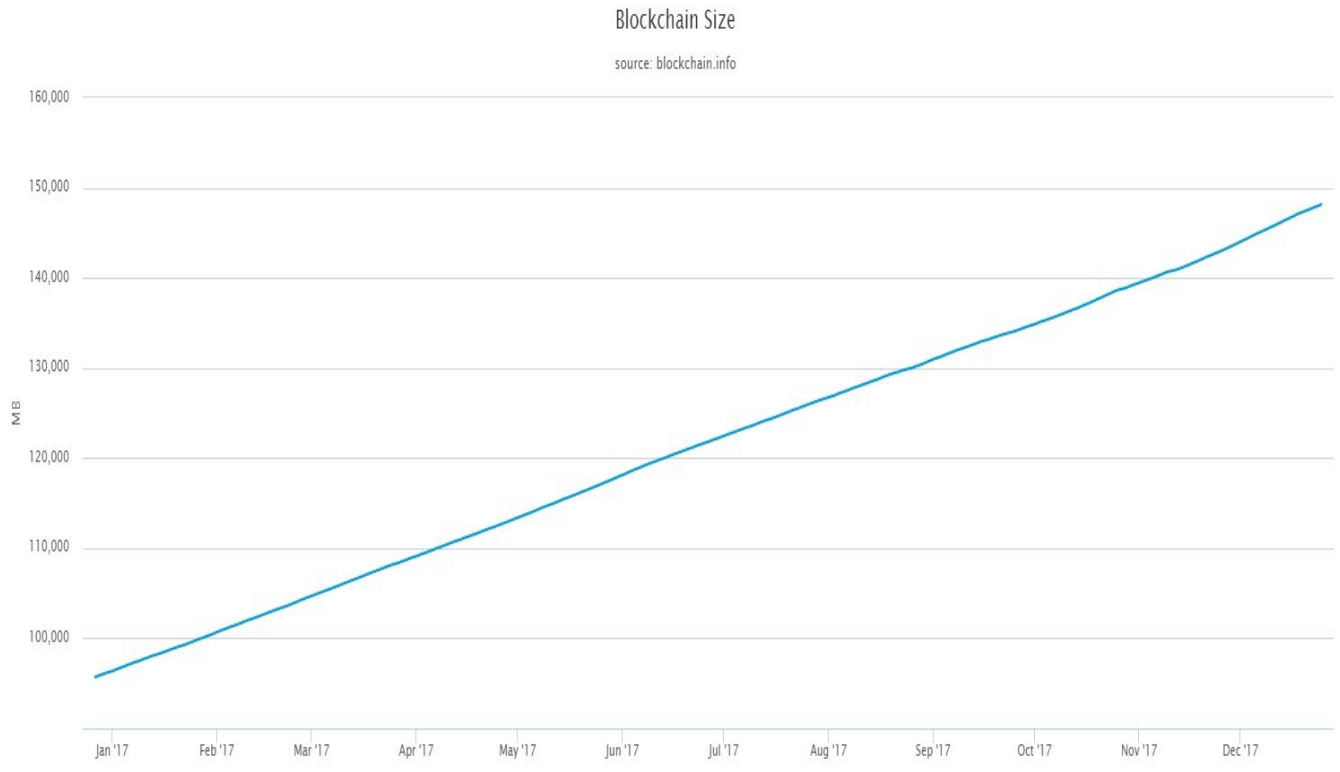
Considerando que os blocos são gerados a cada 10 minutos, podemos deduzir que: 80 bytes * 6 * 24 * 365, resulta em 4.2mb (megabytes)/ ano.

Em 2008, os sistemas de computadores típicos a venda com 2gb (gigabyte) de RAM, e – considerando a lei de Moore prevendo um aumento de 1.2GB por ano –, o armazenamento não se mostrará um problema mesmo se os cabeçalhos dos blocos forem mantidos na memória.

Vamos calcular o espaço utilizado por todos os blocos produzidos em um ano, para entender esse cálculo, temos previamente saber que a cada 10 min um bloco é criado e cada bloco tem 1Mb de tamanho, jogando na fórmula teremos:
[[(1024 Kb de tamanho * 6 * 24 * 365) / 1024 ] /1024] = 51,32 Gb por ano.

A expressão representativa do cálculo:

**[TAM. BLOCO] * [10 MIN/MÉDIA DE NOVO BLOCO] * [UMA HORA (6 * 10 = 60 MIN)] * [1 DIA(24h)] * [365 DIAS (UM ANO)] / [UM MEGABYTE]**




Nesse gráfico podemos confirmar o cálculo, em que a cada ano a rede Bitcoin cresce aproximadamente 51,32Gb. Claro que se formos comparar com o cálculo de economia de espaço onde o sistema guarda apenas o cabeçalho dos blocos, essa diferença fica mais

clara, somente com o cabeçalho dos blocos temos um crescimento de 4.2mb por ano, já um full node utiliza 51,32 GB por ano de espaço em disco.

## VERIFICAÇÃO SIMPLIFICADA DE PAGAMENTO

Verificação simplificada é a verificação pelo cabeçalho do bloco, como vimos anteriormente, quando o sistema enfrenta problemas com espaço em disco, ele irá salvar somente o cabeçalho dos blocos, e também excluirá todas as transações contidas nele, mas sempre que o sistema precisar conferir um pagamento ou consultar uma transação, ele terá que solicitar a outro 'nó' da rede que contenha todas as transações e as forneça ao usuário que está solicitando.

### 1. Para acessar uma transação em um bloco que possui somente o cabeçalho, o que aconteceria nesse caso?

Neste caso o sistema tem a ID do bloco e irá solicitar para outros nós completos que disponibilizem a transação solicitada, assim o sistema poderá realizar o download daquela transação e posteriormente poderá mostrar ao usuário os dados requisitados.

Por isso é extremamente importante garantir a legibilidade dos dados, pois quando um usuário não possuir uma transação, poder-se-á solicitar aos nós completos o seu fornecimento.

As empresas que recebem pagamentos frequentes provavelmente executarão seus próprios nós para obter uma segurança mais independente e uma verificação mais rápida

## COMBINANDO E DIVIDINDO VALORES

Embora seria possível lidar com moedas individualmente, seria custoso criar uma transação separada para cada fração monetária numa transferência.

De modo a permitir fracionamento e combinação de valores, cada transação possui múltiplas entradas e saídas. Normalmente, haverá apenas uma entrada de uma transação anterior mais vultosa, ou múltiplas entradas de transações menores, e no máximo duas saídas: uma para o pagamento e outra devolvendo o troco (se houver) para o pagador.

Podemos ver que nesse parágrafo Satoshi confirma que não existe uma moeda em sí, o que existe são transações registradas no histórico global, onde uma transação determina quem realmente e o possuidor dessa quantidade.

A criação de novas moedas ocorre na transação especial no início de cada bloco, que atribui novas moedas ao criador do bloco. Este tipo de transação é especial justamente porque não tem entrada de recursos; tem apenas uma saída, é análoga a um lançamento contábil de receita.

Note-se que o "espalhamento" onde uma transação depende de múltiplas transações, que por sua vez dependem de muitas outras, não é um problema aqui. Nunca há a necessidade de extrair uma cópia de toda a história de uma transação.

## PRIVACIDADE

No Bitcoin, toda a parte intrínseca às transações são públicas. Por outro lado, graças ao endereço da carteira privada, não existe qualquer vínculo com algum indivíduo per si – e não poderá existir, afinal, ao se criar uma carteira, não há quaisquer solicitações de dados pessoais.

Essa é uma das características que separam o Bitcoin de qualquer outro tipo de transferência digital, pois não precisamos informar nenhum documento ou informação adicional para a rede Bitcoin, para rede, cada endereço é um usuário, e é somente disso que ela precisa, o usuário é seu banco e sua própria agência.

## CÁLCULOS

[Satoshi]

Nós consideramos o cenário em que um atacante tentando gerar uma corrente alternativa mais rápido do que a corrente honesta. Mesmo se isso for conseguido, ela não vai tornar o sistema aberto para mudanças arbitrárias, como criar valor do nada ou tomar o dinheiro que nunca pertenceu ao atacante. Nós não irão aceitar uma transação inválida como pagamento, e nós

honestos nunca aceitarão um bloco contendo-as. Um atacante pode apenas tentar modificar uma de suas próprias transações para pegar de volta o dinheiro que ele gastou recentemente.

[Daniel]

O que ele quer dizer, é que a única forma de atacar a rede é alterar as suas próprias transações, chegamos nessa mesma afirmativa tendo em mente que todas as transações são [6]hasheadas, ou seja, quando uma das informações mudar, a identificação dessa transação é totalmente alterada e por sequencia o bloco também é alterado. Caso o mesmo tente transferir essa transação para os demais 'nós' da rede, eles simplesmente irão rejeitar a transação e o bloco desse usuário, pois todos os usuários possuem todos os blocos, então ficará fácil de saber se uma transação foi adulterada. A única forma desse processo funcionar, é ele vencer a corrida de prova de trabalho para que assim possa ter o direito de minerar o bloco atual, que o mesmo já teria adulterado anteriormente, mas claro, só alterar o bloco e vencer o desafio não fazem uma transação válida, pois depois de cada bloco minerado, ele é enviado a todos da rede para que o mesmo validem e façam a auditoria, e caso alguma transação esteja em desacordo com os registros , será realizada uma votação (1 Voto = 1 CPU como foi explicado no parágrafo Prova de trabalho) levando em conta que para ter sucesso nessa jornada é de extrema importância ter 51% do poder de processamento, pois como foi dito "Se maior parte do poder de processamento é controlado por nós honestos, a corrente honesta irá crescer mais rapidamente e ultrapassar qualquer corrente competidora."

[Satoshi]

A corrida entre a corrente honesta e a corrente do atacante pode ser caracterizada como uma [10]Caminhada Aleatória Binomial. O evento de sucesso é a corrente honesta ser estendida por um bloco, aumentando sua liderança em +1, e o evento de falha é a corrente do atacante ser estendida por um bloco, reduzindo o atraso em -1.

[11]Caminhada Aleatória é um processo matemático que ocorre ao longo de uma série de estados conectado em uma linha. Cada estado é numerado e começamos a partir do estado 0. Lançando uma moeda, no caso de cair o lado cara avançamos uma casa e no caso de cair coroa voltamos uma casa, ao longo de uma série de estados.

A probabilidade de um atacante alcançar de um determinado déficit é análoga ao problema da [12]Ruina do Apostador. Suponha que um apostador com créditos ilimitados comece em déficit e jogue potencialmente um infinito número de vezes para tentar quebrar a banca. Nós vamos calcular a probabilidade de ele nunca conseguir isso, ou que um atacante simplesmente alcance a corrente [13]honesta, como segue :

p = probabilidade de um nó honesto encontrar o próximo bloco q

q = probabilidade de um atacante encontrar o próximo bloco

qz = probabilidade algum dia o atacante alcançar estando z blocos atrás

$$q_z = \begin{cases} 1 & if\ p \leq q \\ (q/p)^z & if\ p > q \end{cases}$$

[Daniel]

A equação de Satoshi é uma variação do modelo Ruína do Jogador, supondo que o apostador joga ao infinito, para chegar ao seu objetivo ou até perder todo seu dinheiro.
Equação:

$$W = \begin{cases} 1 & \text{if } q < p \\ (p/q)^W & \text{if } p < q \end{cases}$$

Voltando a equação do Satoshi onde:

$$q_z = \begin{cases} 1 & if\ p \leq q \\ (q/p)^z & if\ p > q \end{cases}$$

Quer dizer que **se (if) P** (probabilidade de um nó honesto encontrar o próximo bloco) for menor ou igual ao atacante, o atacante irá achar o próximo bloco antes dos nós honestos, representado como **1.**

Agora se **P** (probabilidade de um nó honesto encontrar o próximo bloco) for maior que a do atacante, então temos que calcular as chances dele alcançar o bloco atual, estando **Z** (Número de Blocos) atrás, vamos fazer uma simulação, Imagine que o atacante esteja **2** blocos atras da corrente honesta (bloco atual), e a probabilidade de **p = 0,2** e Probabilidade de **q = 0,1**, então a equação para se calcular é, **Qz = (q/p)** substituindo na fórmula, ficará assim, **$Q2 = (0.1 / 0.2)^2$.**
Vamos calcular: **$Q2 = (0,5)^2$** => **Q2 = 0,25.**

Mas pense que a cada 10 minutos temos um bloco novo no sistema minerado, caso o Invasor esteja a 1 hora tentando, então teremos **60 minutos / 10 minutos = 6 blocos**, jogando na fórmula e acrescentando os 6 blocos ficará assim, **$Q8 = (0,5)^8$.**
Então **Q8 = 0,00390625.**

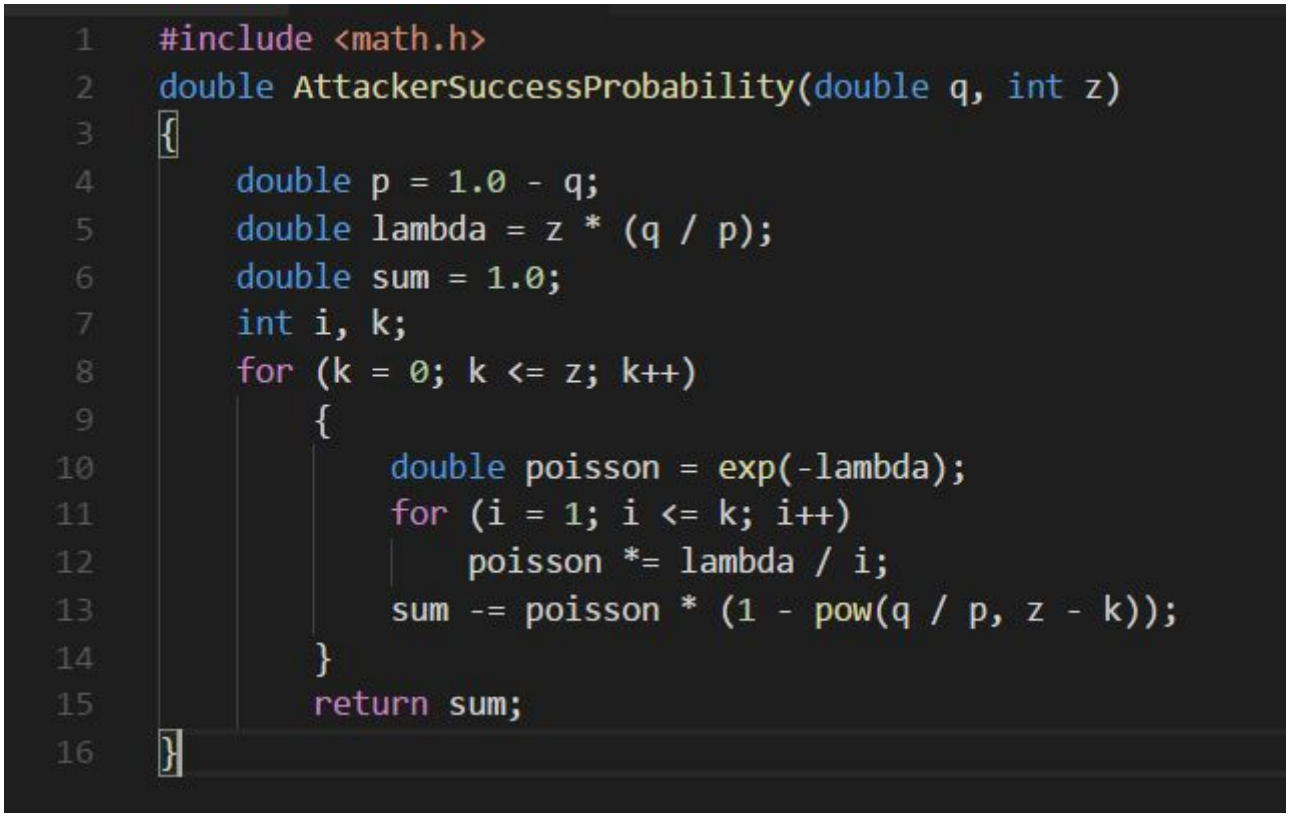
Nesse cálculo, podemos entender que se um atacante estiver competindo pra validar um bloco e não ganhar logo na largada, ele ficará para trás e quanto maior a diferença do bloco que ele está, para o atual, menor a probabilidade de ele chegar novamente a competição.

Você deve estar se perguntando, mas não é só ele tentar competir mais uma vez, a resposta é não, pois ou ele fizer isso, terá que aceitar que a transação dele foi confirmada, e ele quer fazer justamente o oposto, mas se ele não conseguir logo de primeira, terá que refazer a prova de trabalho de todos os blocos que foram criados, após ele alterar a transação. Ele então irá entrar no dilema, em usar o poder computacional para continuar validando os blocos inválidos a fim de alcançar o bloco atual, ou usar seu poder computacional para tentar competir novamente, mas aceitando que sua transação que ele queria burlar já foi concluída e confirmada, sem ter a chance de desfazer ou gastá-la duas vezes.

Aqui temos uma tabela com a [14]probabilidade de um gasto duplo bem sucedido, em função do **hashrate do atacante Q** e o **número de confirmações N.**

| q | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 2% | 4% | 0.237% | 0.016% | 0.001% | ≈ 0 | ≈ 0 | ≈ 0 | ≈ 0 | ≈ 0 | ≈ 0 |
| 4% | 8% | 0.934% | 0.120% | 0.016% | 0.002% | ≈ 0 | ≈ 0 | ≈ 0 | ≈ 0 | ≈ 0 |
| 6% | 12% | 2.074% | 0.394% | 0.078% | 0.016% | 0.003% | 0.001% | ≈ 0 | ≈ 0 | ≈ 0 |
| 8% | 16% | 3.635% | 0.905% | 0.235% | 0.063% | 0.017% | 0.005% | 0.001% | ≈ 0 | ≈ 0 |
| 10% | 20% | 5.600% | 1.712% | 0.546% | 0.178% | 0.059% | 0.020% | 0.007% | 0.002% | 0.001% |
| 12% | 24% | 7.949% | 2.864% | 1.074% | 0.412% | 0.161% | 0.063% | 0.025% | 0.010% | 0.004% |
| 14% | 28% | 10.662% | 4.400% | 1.887% | 0.828% | 0.369% | 0.166% | 0.075% | 0.034% | 0.016% |
| 16% | 32% | 13.722% | 6.352% | 3.050% | 1.497% | 0.745% | 0.375% | 0.190% | 0.097% | 0.050% |
| 18% | 36% | 17.107% | 8.741% | 4.626% | 2.499% | 1.369% | 0.758% | 0.423% | 0.237% | 0.134% |
| 20% | 40% | 20.800% | 11.584% | 6.669% | 3.916% | 2.331% | 1.401% | 0.848% | 0.516% | 0.316% |
| 22% | 44% | 24.781% | 14.887% | 9.227% | 5.828% | 3.729% | 2.407% | 1.565% | 1.023% | 0.672% |
| 24% | 48% | 29.030% | 18.650% | 12.339% | 8.310% | 5.664% | 3.895% | 2.696% | 1.876% | 1.311% |
| 26% | 52% | 33.530% | 22.868% | 16.031% | 11.427% | 8.238% | 5.988% | 4.380% | 3.220% | 2.377% |
| 28% | 56% | 38.259% | 27.530% | 20.319% | 15.232% | 11.539% | 8.810% | 6.766% | 5.221% | 4.044% |
| 30% | 60% | 43.200% | 32.616% | 25.207% | 19.762% | 15.645% | 12.475% | 10.003% | 8.055% | 6.511% |
| 32% | 64% | 48.333% | 38.105% | 30.687% | 25.037% | 20.611% | 17.080% | 14.226% | 11.897% | 9.983% |
| 34% | 68% | 53.638% | 43.970% | 36.738% | 31.058% | 26.470% | 22.695% | 19.548% | 16.900% | 14.655% |
| 36% | 72% | 59.098% | 50.179% | 43.330% | 37.807% | 33.226% | 29.356% | 26.044% | 23.182% | 20.692% |
| 38% | 76% | 64.691% | 56.698% | 50.421% | 45.245% | 40.854% | 37.062% | 33.743% | 30.811% | 28.201% |
| 40% | 80% | 70.400% | 63.488% | 57.958% | 53.314% | 49.300% | 45.769% | 42.621% | 39.787% | 37.218% |
| 42% | 84% | 76.205% | 70.508% | 65.882% | 61.938% | 58.480% | 55.390% | 52.595% | 50.042% | 47.692% |
| 44% | 88% | 82.086% | 77.715% | 74.125% | 71.028% | 68.282% | 65.801% | 63.530% | 61.431% | 59.478% |
| 46% | 92% | 88.026% | 85.064% | 82.612% | 80.480% | 78.573% | 76.836% | 75.234% | 73.742% | 72.342% |
| 48% | 96% | 94.003% | 92.508% | 91.264% | 90.177% | 89.201% | 88.307% | 87.478% | 86.703% | 85.972% |
| 50% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% | 100% |

Logo após, Satoshi nós mostra um script para simular os cálculos,



```c
#include <math.h>
double AttackerSuccessProbability(double q, int z)
{
    double p = 1.0 - q;
    double lambda = z * (q / p);
    double sum = 1.0;
    int i, k;
    for (k = 0; k <= z; k++)
        {
            double poisson = exp(-lambda);
            for (i = 1; i <= k; i++)
                poisson *= lambda / i;
            sum -= poisson * (1 - pow(q / p, z - k));
        }
        return sum;
}
```

Caro leitor, caso seja um Programador pule essa parte, pois vou dar uma pequena orientação sobre programação informando o número da linha e o que esta ocorrendo nela:

**1 -** Temos a função **#Include <Math.h>** está dizendo que ele quer que importe a biblioteca Math, que possuem fórmulas para cálculos matemáticos.

**2-** Aqui estamos criando um novo método para que seja executado, sempre que temos no início do método Double, quer dizer que o seu retorno vai ser em double, os números doubles são números finitos com casas decimais com até 10 casas de precisão., nessa linha temos na ordem [Tipo de retorno] [Nome do Método]([parâmetro de entrada tipo double],[Parâmetro de entrada do tipo int (Inteiros)] representado por **double AttackerSuccessProbability(double q, int z).**

**3-** O carácter **"{"** é representado na linguagem **C**, como abertura do corpo de uma função e tudo que estiver posterior a ele será, claro que depois de abrir uma função temos que ter o caráter de fechamento, que é representado como **"}".**

**4 -** Nesta linha estamos declarando uma nova variável, as variáveis são uma representação de tipo de dados na memória. após declarar o tipo da variável demos o nome de **P** , que irá

receber o numero **1.0** menos (-) o parâmetro de entrada **Q**, sendo representada por **double p = 1.0 - q;**

**5-** Vemos novamente uma declaração de uma variável **double** nomeada de [15]**lambda** que representa o parâmetro de uma distribuição de [16]poisson e irá receber o resultado do cálculo entre o parâmetro de entrada **Z** * (parâmetro de entrada **Q** / pela variável definida anteriormente como P), sendo representada por **double lambda = z *(q / p);**

**6-** Declaramos uma nova variável do tipo **double** com o nome de **SUM** e definimos ela com o valor de **1.0** double sum = 1.0; Representação **double sum = 1.0;**

**7 -** Nesta linha vemos duas variáveis do tipo inteiro, ou melhor, tipo **int** a única função delas é controlar os looping de dados, que veremos nas próximas linhas. Representação **int i,k;**

**8 -** Nós se deparamos com nosso primeiro laço de repetição, conhecidos também como loop, existem muitos tipos de loop em programação, mas vamos nos atentar apenas ao **FOR,** FOR é utilizado sempre que queremos testar alguma condição, exemplo 1 + 1 = 2, caso seja quero que repita o processo. o parâmetro para o FOR são (inicialização ; até quando irá ser executado ; incremento ou decremento). onde a condição para que ele se repita é **K** menor ou igual a **Z.** Representação **for (k= 0; k <= z;k++).**

**9-** Abertura do corpo da função **FOR.**

**10 -** Agora declaramos uma variavel do tipo double com o nome poisson, ele será igual a [17]exp, onde **exp** é a função que calcula o valor exponencial de x, ela recebe como parâmetro a variável **-lambda**, pois ela esta sendo passada como variável negativa. Representação **double poisson = exp(-lambda);**

**11-** Vamos agora criar um novo loop, dentro de outro loop sendo que o **i** está recebendo **1** e não zero como no primeiro laço, sendo que a condição agora é **I** menor ou igual a **K.** Representação **for (i = 1; i <= k; i++).**

**12-** Vemos nesta linha a variável poisson declarada recentemente, sendo multiplicando por ela mesma, ou seja, na programação quando temos +=, -= e *= , quer dizer que a variável antes do = esta fazendo o calculo com ela mesma, um exemplo é **poisson *= ,** o mesmo **poisson = poisson *.** Continuando**,** poisson receberá o resultado de **poisson** * **lambda** / **pela variável de controle I** poisson *= lambda / i;

**13-** Nesta pela última linha temos a variável **SUM** recebendo o cálculo de **(sum - poisson *(1 - pow**(resultado da divisão do parâmetro **Q** , outro parâmetro de entrada para essa função é parâmetro de entrada do método **Z** menos a variável de controle do primeiro loop **K),** a função POW Esta função retorna o resultado de aumentar x para a potência y . Representação **sum -= poisson * (1 - pow(q / p, z - k));**

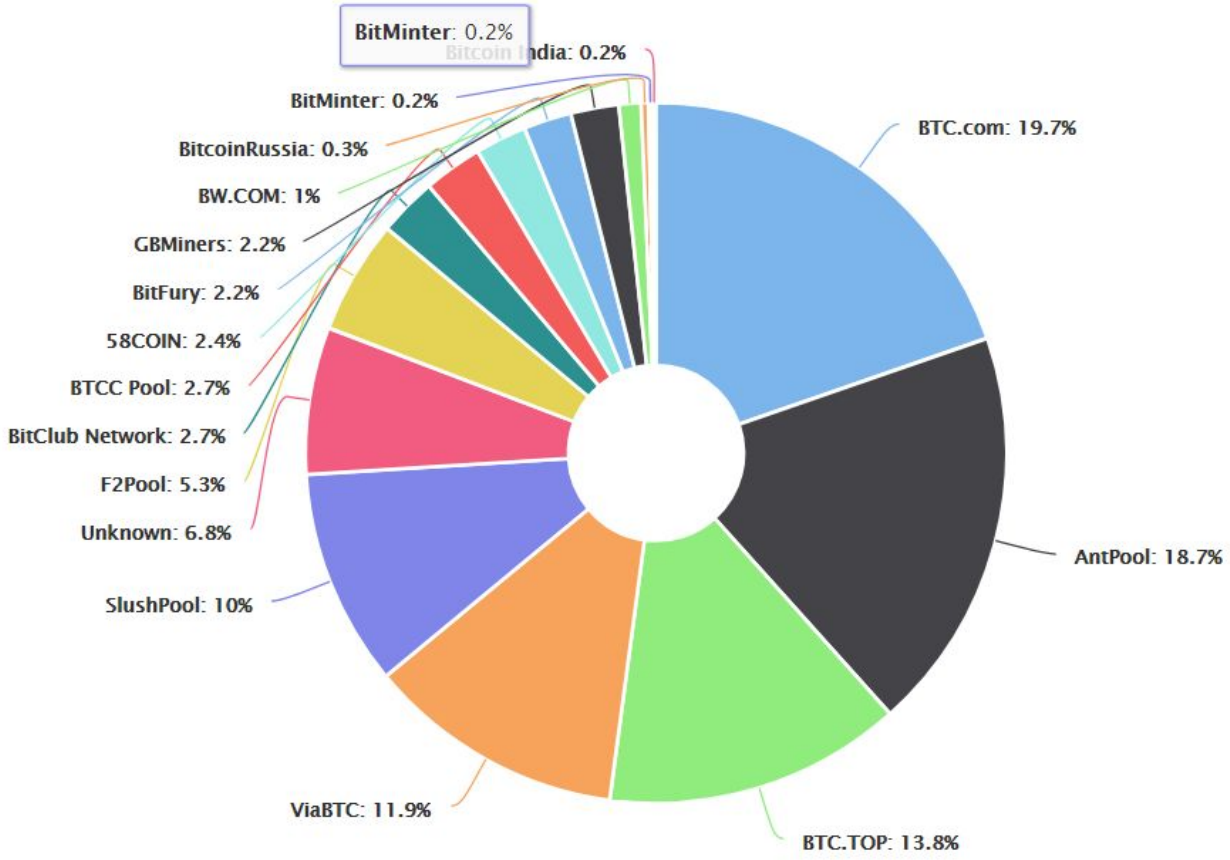
**14-** Fechamento do corpo do Primeiro **FOR.**

**15 -** Retorna a variável **SUM** carregada com o resultado dos cálculos, para o programa que chamar esta função, a **AttackerSuccessProbability.**

**14-** Fechamento do corpo principal.

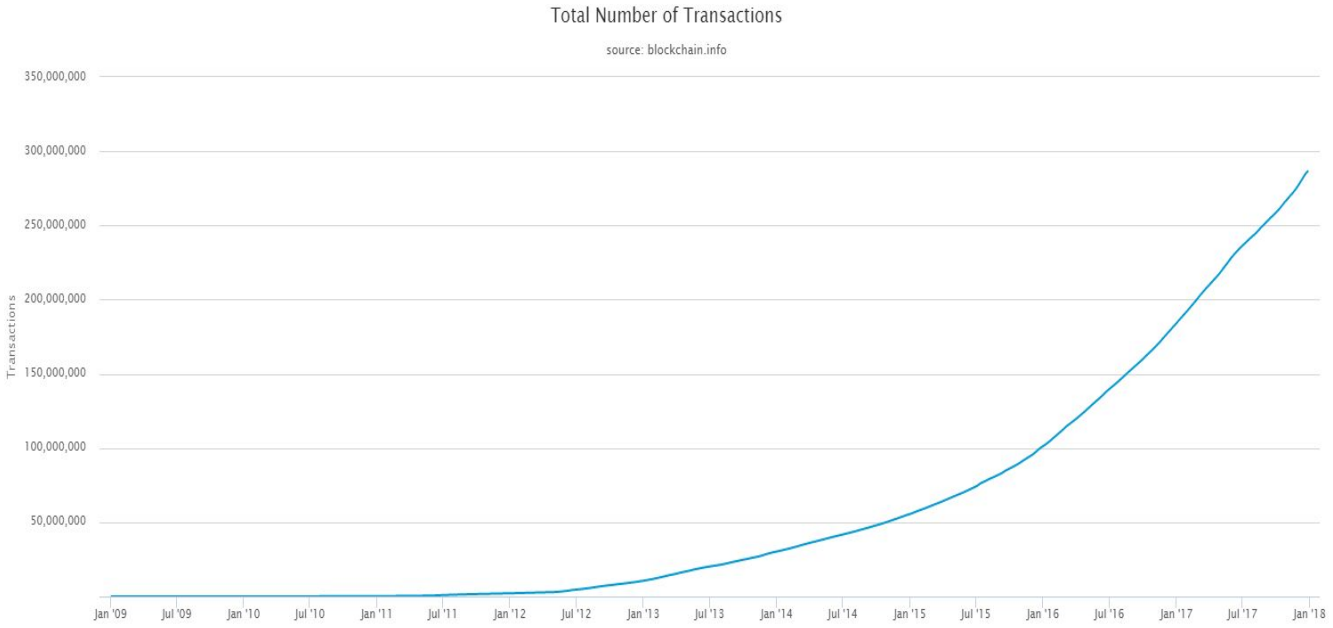
## Conclusão

Com este Whitepaper podemos entender o tamanho da engenhosidade do projeto, sem nenhum órgão regulador ou centralizador das informações, uma gama de projetos foram criadas em cima do Bitcoin, mas na minha opinião o seu maior legado foi deixar uma passo a passo para criar um protocolo como o Blockchain, hoje vemos que o bitcoin já não é tão descentralizado assim, pois as maiores POOL de mineração, que são apena cinco empresas,  ANTPOOL 18.7%, BTC.TOP 13.8%, BTC.COM 19.7%, VIABTC 11.9%, SLUCHPOOL 10%,




juntas elas tem um poder de processamento de  74.1%, então juntas elas tem 74.1% de poder de voto em caso de algumas alterações no sistema outro caso e que a cada dia que passa as transações em BTC estão ficando mais cara, hoje em dia não podemos fazer o que o Bitcoin foi planejado para fazer, isso não é culpa dos usuários e sim porque o hype foi tão grande e um curto espaço de tempo, que os desenvolvedores ainda estão tentando fazer updates para resolver essa questão, os micro-pagamentos já não existe no BTC, tenho a certeza absoluta que nem no seu melhor sonho, Satoshi Nakamoto acharia que chegaria a esse ponto, claro que um dia sim, mas não em tão pouco tempo, hoje vemos que se passaram aproximadamente apenas 10 anos da escrita

desse documento, mas se formos analisar os gráficos de movimentação, somente em Janeiro de 2016 que realmente todos os blocos estão usando toda a sua capacidade de transações, o que quer dizer que existe uma alta demanda de transações para serem mineradas e claro que as transações com maior taxa, serão mineradas primeiros, como mostra o gráfico:




Aqui vemos uma crescente número de transações em 2016, esse número começou crescer de forma absurda, e se manteve no padrão depois disso, mostrando que os blocos estão totalmente carregados de transações, e com isso trabalharão no seu máximo.

## Doações

Caso você queria ajudar o projeto, realizando uma doação, vou deixar logo abaixo o endereços disponíveis.

RAIBLOCKS:
xrb_1yq8y6arot7867nmpi1aq1y9pb8im11fn7oepx4g6bi1o1ji7pq1k48fprtm

ETHEREUM:
0x5a6160a64F113845470948d87f333C6B3dD3A4Bb

BITCOIN:
3LdKteDzboVHbL66yG9or7gzqoX2owY53n

LITCOIN:
LTtrhXTFVLtHQnpfAYkRdw9rgKy3KBWrsu

NEO:
AKsQcdQvYwwx2YTpDig6rMmB6SiT2Yvp1e

DASH:
Xf8xyR3UAJAnCVDR8Nu6PgnreHxBjN3nwD

# REFERÊNCIAS

## 1. MARCADORES


1 - Bitcoin: Um Sistema de Dinheiro Eletrônico Ponto-a-Ponto
2 - P2P = Peer-to-Peer (ponto a ponto) [LINK]
3 - Cadeia de blocos que em inglês tem como nome Blockchain [LINK]
4 - PoW - Proof-of-Work (Prova de Trabalho)
5 - Visão sobre o dinheiro e importância na sociedade - Sapiens - Livro
6 - Hash [LINK]
7 - Output = Transação de Saída
8 - Conhecido em inglês como Node
9 - SHA256 é o tipo de criptografia usada no Bitcoin [LINK]
10 - Caminhada Aleatória Binomial [LINK]
11 - Caminhada Aleatória [LINK]
12 - Explicação Ruína do Apostador [LINK]
13 - W. Feller, "An introduction to probability theory and its applications," 1957.
14 - Tabela DoubleSpend [LINK]
15 - Lambda [LINK]
16 - Poisson [LINK].
17 - Função EXP [LINK]


## 2. LIVROS


- Mastering Bitcoin - Mastering Bitcoin por Andreas M. Antonopoulos LLC [LINK]
- Blockchain Revolution por Don Tapscott & Alex Tapscott [LINK]
- Sapiens – Uma Breve História da Humanidade - Yuval Harari


## 3. ARTIGOS


- WhitePaper Bitcoin - Satoshi Nakamoto [LINK]
- Digital Cash [LINK]
- Minimum Viable Blocks - IGVITA [LINK]
- The Blockchain is a Timestamp Server its Purpose - REDDIT [LINK]
- Why Does Only the Earliest Transaction Matter For Double Spending? [LINK]

- UTC - Universal Time Coordinated [LINK]
- How to Time-Stamp a Digital Document - Stuart Haber & W. Scott [LINK]
- Block Timestamp - Bitcoin Wiki [LINK]
- Hacking Timestamp & Bitcoin [LINK]
- HashCash - Adam Back [LINK]
- Proof of Work - Bitcoin Wiki [LINK]
- Hash Puzzles and Proofs of Work [LINK]
- Lei de Moore [LINK]
- Árvore Markle [LINK]
- Block Size - CoinDesk [LINK]
- WhitePaper Bitcoin - Satoshi Nakamoto [LINK]