

# ETHERVOLTZ: UM SISTEMA DE VOTAÇÃO AUDITÁVEL NO BLOCKCHAIN ETHEREUM

ALENCAR, M. F.<sup>1</sup>; SIMÕES, M. C.<sup>1</sup>

1 – ETEP FACULDADES/FACULDADE DE TECNOLOGIA DE SÃO JOSÉ DOS CAMPOS

mtsalc@ gmail.com; marize.simo@etep.edu.br

**Resumo** – Nos sistemas de votação de primeira, segunda e terceira geração, todas as evidências geradas no momento do voto ficam sob custódia do administrador do processo de votação. Esse tipo de centralização de poderes facilita fraudes, burlas e dificulta auditorias. A adoção em massa da internet permitiu o desenvolvimento de uma nova classe de programas distribuídos que substituem autoridades centrais por um livro razão público para garantir consenso sobre o estado da uma aplicação. Um exemplo conhecido desse tipo de aplicação distribuída é o Bitcoin. Este trabalho apresenta um sistema de votação que transforma o voto do eleitor em uma criptomoeda batizada de VoltToken para descentralizar o destino das provas geradas no momento do voto e dar auditabilidade ao processo eleitoral.

**Palavras-chave:** ethervoltz. blockchain. criptomoeda. sistema. votação. eleições.

**Abstract** – In first, second and third generation voting systems, all of the evidence produced when a voter casts a vote stays under custody of the administrator of the voting process. This kind of centralization results in opacity in the democratic process enables frauds and places barriers on audits. The mass adoption of the internet allowed the development of a new class of distributed applications that replaces trusted central authorities with a public ledger to reach consensus on the current state of an application. A known example of such an application is Bitcoin. This paper presents a voting system that turns the vote into a cryptocurrency to decentralize the destination of audit trails.

**Keywords:** ethervoltz. blockchain. cryptocurrency. voting system. elections.

## I. INTRODUÇÃO

Em sistemas de votação de primeira, segunda e terceira geração (BRUNAZO, 2014), todas as provas geradas pelos votos ficam sobre controle do administrador e precisam ser levadas dos locais de votação para as centrais de apuração dos votos. Além dos custos envolvidos para garantir que estes equipamentos não sejam alterados ou destruídos durante o transporte, o administrador também precisa guardar estes registros após a apuração de votos para auditorias.

Após o período eleitoral, caso um cidadão queira auditar os resultados das eleições, ele precisa interagir com o administrador do processo eleitoral para ter acesso aos registros digitais, aos equipamentos e aos registros independentes de *software* (caso o administrador tenha optado por um sistema eleitoral independente de *software*). Cabe ao administrador decidir se ele tem ou não permissão para realizar a auditoria e quais são condições para a realização da mesma.

Os problemas deste tipo de centralização de poderes ficam evidentes nas eleições do Brasil. Embora não seja objetivo deste artigo discutir esses problemas, quatro casos são listados:

1. O Caso Diadema, SP - 2000: Foram negados a todos os partidos que solicitaram o acesso aos registros digitais dos votos realizados nas urnas eletrônicas. Somente 9 meses após a eleição, os partidos obtiveram acesso, não aos registros dos votos, mas aos Arquivos de LOG das urnas, que demonstravam que todas haviam sido carregadas fora da cerimônia oficial de carga e lacramento (SÉRVULO et al; 2010);
2. O Caso Marília, SP - 2004: Em auditoria, os Arquivos de Espelhos de Boletins de Urna da 400ª Zona Eleitoral indicavam que muitas seções eleitorais tiveram seus resultados recebidos para apuração antes do início da votação (SÉRVULO et al; 2010);
3. O Caso Alagoas - 2006: Diversas irregularidades nos arquivos gerados pelas urnas foram detectadas por auditores externos. Frente as evidências, o administrador negou acesso aos arquivos solicitados pelos auditores e transferiu ao requerente uma cobrança antecipada no valor de R\$ 2 milhões para que fosse desenvolvida uma perícia das urnas. Diante do não pagamento do valor proibitivo, o requerente foi multado e condenado por litigância de má-fé (SÉRVULO et al; 2010);
4. O Caso Itajaí, SC - 2008: Nenhuma urna preparada para a votação passou pelo teste obrigatório prescrito pelo Art. 32 da Res. TSE 22.712/08. Um caso foi o da 97ª Zona Eleitoral onde a urna da seção 236 que foi sorteada para o teste obrigatório foi substituída por outra na hora do teste, preparada exclusivamente para este fim. A urna que foi utilizada para o teste foi posteriormente colocada à parte e recarregada, procedimento que destruiu eventuais provas nela gravadas (SÉRVULO et al; 2010).

Em países como Alemanha, Argentina e Equador, que utilizam urnas mais modernas como as de segunda e terceira geração (BRUNAZO, 2014), o Princípio da Independência de Software em Sistemas Eleitorais (RIVEST; WACK, 2006) é preservado. Entretanto, todas as provas geradas no momento do voto seguem sobre controle do administrador. Auditorias ocorrem apenas com a autorização e condições impostas pelo mesmo. O caminho do voto da urna até a apuração funciona como uma caixa preta em que o eleitor precisa confiar no administrador e em todos envolvidos no processo.

O objetivo do projeto é apresentar um modelo de sistema eleitoral que respeita o Princípio da Independência de Software em Sistemas Eleitorais e que descentralize o destino das provas geradas em cada voto, de forma que os registros físicos ficam sobre custódia do administrador e os registros digitais ficam sobre controle de um programa

autônomo que pode ser auditado sem a necessidade de interagir com o administrador.

Para atingir o objetivo, o artigo propõe a utilização do *blockchain Ethereum* como o *backend* que gerencia e armazena os registros digitais dos votos. Esta estratégia traz ao sistema as seguintes características:

- Decentralização: Como o computador mundial não possui um dono ou entidade responsável por sua administração, nenhuma instituição ou pessoa possui poder de censurar ou de alguma forma impedir que aplicações hospedadas na plataforma se mantenham em execução.
- Distribuição: Diferente de sistemas que utilizam a arquitetura cliente-servidor, aplicações que executam no computador mundial *Ethereum* são resistentes a ataques de negação de serviço distribuídos.
- Todas atualizações na base de dados são registradas permanentemente no *blockchain* estão disponíveis para auditoria por qualquer pessoa com acesso a internet, a qualquer momento.

## II. FUNDAMENTAÇÃO TEÓRICA

Para facilitar a explicação do funcionamento da solução proposta, são definidos nesta seção diversos termos utilizados ao longo do documento.

Esta seção também é uma revisão bibliográfica do estado da arte de aplicações distribuídas no paradigma de programação orientada a contratos inteligentes.

Embora definidos brevemente aqui, o leitor se beneficiará se possuir conhecimento prévio sobre os mecanismos envolvidos em sistemas de votação, programação orientada a contratos inteligentes e criptografia. O leitor se beneficiará principalmente se possuir conhecimento sobre o funcionamento de *blockchains* como o *Bitcoin* e *Ethereum*.

### 2.1 – Princípio da Independência de Software em Sistemas Eleitorais

Segundo Rivest (2006), um dos autores da chave de criptografia RSA e autor do Princípio:

“Um sistema eleitoral é independente do software se uma modificação ou erro não-detectado no seu software não pode causar uma modificação ou erro indetectável no resultado da apuração (RIVEST, R.L; WACK, J.P. p.1, 2006).”

### 2.2 – VICE

O Voto Impresso Conferido Pelo Eleitor ou VICE é um documento em papel que é apresentado ao eleitor no momento da votação e que funciona como um comprovante do candidato que recebeu o voto. O VICE é apresentado para que ele possa confirmar visualmente o voto, mas ao qual ele não tem contato físico e nem leva para casa (BRUNAZO, 2014).

Os termos VICE, registro de voto físico, prova física e voto impresso são utilizados de forma intercambiável neste documento e significam a mesma coisa.

### 2.3 – Livro Razão Público

Livro razão é o nome dado a um documento que agrupa modificações ordenadas do estado de alguma informação.

*Blockchains* são comumente comparados a livros razão públicos, pois são uma sequência de alterações de estado ordenadas uma após a outra e cujas alterações dependem necessariamente do estado anterior. A Tabela 1 apresenta um trecho de um exemplo de livro razão.

Tabela 1 – Exemplo de livro razão.

ID	Remetente	Parâmetros	Destinatário
...	...	...	...
3561	Bob	3 reais	Alice
3562	João	6 reais	Bob
3563	Alice	2 reais	João
3564	Carlos	1 real	Alice
3565	João	2 reais	Carlos
...	...	...	...

Fonte: Autor, 2017.

No exemplo da Tabela 1, a transação número 3564 só é considerada válida e só será incluída no livro razão se Carlos possuir 1 real ou mais. Isto é, se recebeu um 1 real ou mais em uma transação anterior aprovada e inserida no livro razão.

### 2.4 – Estado

O estado de uma aplicação *blockchain* é o conjunto de todas as informações e variáveis da aplicação em um determinado instante e que um sistema deve rastrear (VOGELSTELLER et al, 2017). Em sistemas mais simples isto pode ser apenas o balanço de contas e em sistemas mais complexos, estrutura de dados que fazem parte da aplicação.

### 2.5 – Transação

Uma transação é uma mensagem assinada criptograficamente que autoriza uma ação em particular no *blockchain*. Um exemplo de transação em uma aplicação de criptomoeda é uma que autoriza a transferência de unidades de um endereço a outro (VOGELSTELLER et al, 2017). Uma transação também pode incluir parâmetros da chamada de um método de uma aplicação distribuída.

Sempre que uma transação é confirmada, um *hash* que identifica esta transação unicamente é gerado a partir do conjunto de informações contidas nela e na história do *blockchain* até o momento. Com este *hash* é possível obter informações sobre a transação. Este recurso é explorado no EtherVoltz para garantir a rastreabilidade das operações realizadas na aplicação durante uma auditoria.

### 2.6 – Hash

Uma função *hash* (ou algoritmo *hash*) é um processo pelo qual um dado de qualquer e de tamanho arbitrário como um programa, imagem, filme ou mesmo uma lista de *hashes* é processado e gera uma pequena quantidade de dados, que parece completamente aleatório, e a partir do qual não podem ser recuperados dados significativos sobre o documento que o gerou (VOGELSTELLER et al, 2017).

Por exemplo, o *hash* SHA3-224 da palavra “EtherVoltz” (sem as aspas) é   
f0b7aa13021d863d540c5d37655c348f34  
94dcb008b69bca9f724498.

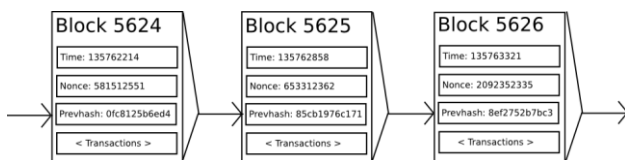
### 2.7 – Bloco

Um bloco *b* é um pacote de dados contendo uma lista de transações *T*, uma referência à um bloco anterior *d* e opcionalmente, mais informações (VOGELSTELLER et al,

2017). A inclusão de um bloco no *blockchain* implica na atualização do estado, mas o bloco só é incluído se for considerado válido (NAKAMOTO, 2009). Um bloco  $b$  é considerado válido se:

- Todas as transações pertencentes a  $T$  listadas nele são válidas.
- O bloco  $d$  ao qual ele faz referência é válido.
- As condições impostas pelo protocolo de consenso são atingidas.

Figura 1 – Sequência de Blocos.



Fonte: Ethereum White Paper, 2017.

A Figura 1 apresenta um exemplo de sequência de blocos e dados que podem possuir. Note que cada bloco referencia o *hash* das informações do bloco precedente, de forma que uma alteração mínima no conteúdo de qualquer bloco na história do *blockchain* causaria uma mudança em todos os blocos seguintes.

## 2.8 – Blockchain

Um *blockchain*  $B=[b_0, b_1, b_2, b_3, \dots]$  é uma sequência de blocos  $b_n$  em que cada item faz referência ao *hash* do item precedente até o bloco gênese " $b_0$ ". Um *blockchain* é dito válido se cada bloco  $b_i$  pertencente a  $B$  for válido (VOGELSTELLER et al, 2017).

Em discussões sobre desenvolvimento de aplicações distribuídas empoderadas por tecnologia *blockchain* - e em alguns trechos neste documento - é comum se referir ao mesmo como um banco de dados distribuído, visto que existem cópias dele armazenadas nos computadores dos milhares de clientes e mineradores espalhados pelo mundo. Neste documento o termo *blockchain* será utilizado para se referir especificamente ao *blockchain* utilizado pelo computador mundial *Ethereum*, mas vale lembrar que esta estrutura de dados também é utilizada em outros sistemas como Bitcoin e Litecoin (NAKAMOTO, 2009).

## 2.9 – Nó, Nó Completo.

Neste documento a palavra "nó" se refere a um computador conectado à rede *Ethereum* através de algum cliente como *geth* ou *pyeth* e que possui uma cópia completa ou parcial do *blockchain*.

O termo "nó completo" é utilizado para especificar que o nó em questão possui uma cópia do *blockchain* completa e válida, já que existem nós ditos leves, que possuem apenas cópia parcial do *blockchain* para execução em dispositivos com poucos recursos (e.g. *smartphones*, relógios, IoT).

## 2.10 – Ethereum, Computador Mundial

*Ethereum* ou Computador Mundial é, em um sentido técnico, um computador mundial que pode ser utilizado e programado por qualquer pessoa que tenha um computador com acesso a internet. Possui apenas um processador e um *thread* para executar programas, mas tanta memória quanto for necessária.

Qualquer pessoa pode escrever programas que a máquina virtual *Ethereum* é capaz de interpretar, implantá-los na rede e fazer requisições ao programa. Por isto, a pilha de tecnologias formada pela máquina virtual *Ethereum*, *blockchain* e rede *peer-to-peer* é frequentemente chamada de Computador Mundial.

A rede *peer-to-peer* é formada por pessoas anônimas que dedicam hardware e eletricidade para executar estes programas e recebem em troca uma recompensa financeira para isto.

Outra característica importante é a de que, em um sentido técnico, cada programa possui seu próprio armazenamento que persiste entre execuções. Enquanto houver demanda, o computador mundial e todos os programas estarão disponíveis (VOGELSTELLER et al, 2017).

Programas no computador mundial executam exatamente como programados e são ditos autônomos. A implicação disto é de que um desenvolvedor pode escrever um programa que só pode receber requisição de certas pessoas, podendo inclusive revogar o direito do próprio criador do programa de interagir com ele para garantir transparência a terceiros. Este é um recurso utilizado no núcleo do projeto EtherVotz em que o administrador do processo eleitoral revoga parte do próprio poder de interação com o programa para dar transparência e imutabilidade ao processo.

## 2.11 – Contrato Inteligente

*Smart contract*, contrato inteligente ou simplesmente contrato, é um termo utilizado para referir a código que executa no computador mundial. O núcleo da prova de conceito concebida no projeto EtherVotz é um contrato inteligente escrito na linguagem de programação *Solidity*.

Formalmente, são contas que contém e são controladas por código. Por padrão, contratos só podem ser controlados diretamente por chaves privadas se isto for definido em código. O efeito disto, é que um contrato não possui "dono" após *deployment* no *blockchain*.

O criador de uma aplicação distribuída pode revogar o próprio poder de tirar uma aplicação do ar, de forma que nenhuma instituição, governo ou pessoa tenha o poder coagir o criador da aplicação a tirar o programa e conteúdo do ar.

## 2.12 – Criptomoeda, Token, VoltToken

Uma criptomoeda é um bem digital projetado para servir como meio de troca utilizando criptografia para assegurar transações e controlar a emissão de novas unidades da moeda (CHOHAN, 2017).

VoltToken é a criptomoeda proposta e implementada na prova de conceito deste projeto, para servir como um meio do eleitor expressar sua intenção de voto.

Neste documento, a palavra *token* é propositalmente utilizada para evidenciar que diferente de criptomoedas comuns, não pode ser transferida ou comercializada livremente de uma pessoa para outra como uma moeda comum. Possui regras definidas em código que restringem a transferência em tempo, quantidade e destinatários.

## 2.13 – Carteira

Embora incorreto tecnicamente, o termo "carteira" é utilizado em discussões sobre criptomoedas como uma boa analogia para explicar o efeito de possuir um par de chaves

pública e privada capaz de autorizar e receber transações (VOGELSTELLER et al, 2017).

A chave pública funciona como um "endereço" que pode ser utilizado para receber transações.

A chave privada pode ser vista como uma assinatura que deve ser mantida em segredo, capaz de autorizar o envio de transações.

### III. METODOLOGIA

#### 3.1 – Requisitos

Para o desenvolvimento do projeto, foram estabelecidos os seguintes requisitos:

1. Velocidade de Apuração – O sistema precisa entregar velocidade de apuração igual ou superior a dos sistemas de votação atuais;
2. Disponibilidade – Os registros digitais de votos precisam estar armazenados em um sistema confiável, que seja resistente a ataques de negação de serviço e devem estar sempre disponíveis para auditorias;
3. Integridade – Os registros digitais dos votos precisam ser imunes a alteração não autorizada e logs de alterações executadas devem imutáveis e permanentes;
4. Descentralizado e Autônomo – Parte do software e a infraestrutura utilizada não devem estar sobre controle ou custódia de uma autoridade central;
5. Independência de Software – Erros ou alterações não detectados no software do sistema não devem poder causar modificações ou erros indetectáveis no resultado final.

#### 3.2 – Pilha de Tecnologias e Arquitetura

Para garantir os requisitos 1,3 e 4, foi adotada a estratégia de transformar cada voto em uma criptomoeda batizada de *VoltToken*, cujas regras de emissão e transferência são definidas em um contrato inteligente programado na linguagem *Solidity*. O contrato é uma implementação parcial do padrão ERC20, com funções adicionais e regras que regulamentam através de “listas brancas” e através do tempo, quando e quem pode interagir com a aplicação para realizar transferências, incluir ou remover candidatos e máquinas de votar.

A biblioteca *zeppelin* foi utilizada para atribuir um proprietário ao contrato inteligente durante a fase de testes. A motivação disto, é dar poder ao administrador de excluir o contrato após *deployment* na rede. Este recurso seria removido em uma situação real para dar transparência e imutabilidade ao processo.

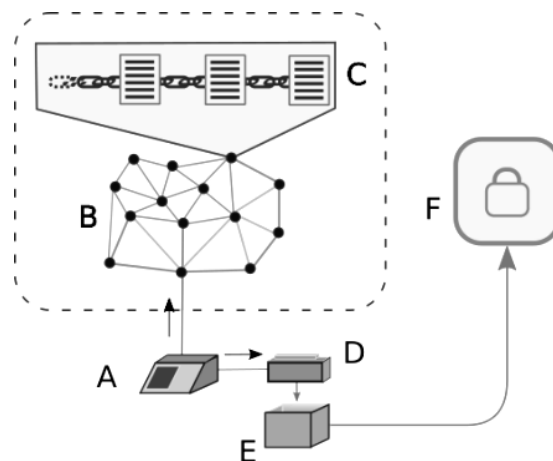
A linguagem *Solidity* é uma linguagem de alto nível similar ao *javascript* e ao *python*, orientada a contratos inteligentes e que é compilada para *bytecode* que a máquina virtual *Ethereum* é capaz de interpretar. (REITWIESSNER et al., 2017).

Na Figura 2, as letras representam respectivamente:

- A. Um computador sob controle do administrador que o eleitor usa para votar e que possui a chave privada da carteira associada à urna, capaz de assinar transferências de VoltTokens para carteiras associadas a candidatos;
- B. A rede *peer-to-peer* formada pelos milhares de nós e mineradores da rede *Ethereum*;

- C. A cópia do *blockchain Ethereum* em um nó na rede;
- D. Uma impressora para realizar a impressão de Votos Impressos Conferidos Pelos Eleitores (VICES) (BRUNAZO, 2014);
- E. Uma caixa lacrada para a coleta dos VICES;
- F. Um cofre sob controle do administrador para armazenamento dos VICES.

Figura 2 – Arquitetura do sistema EtherVoltz



Fonte: Autor, 2017.

Embora o *backend* seja resistente a alterações e infecções, no EtherVoltz as chaves privadas que assinam cada voto estão sob controle do administrador, estão sujeitas a ataques internos que podem causar o vazamento das mesmas. De maneira similar, o *frontend* utilizado para interagir com a aplicação hospedada na rede *Ethereum* pode conter erros ou estar infectado com código malicioso capaz de fraudar votos.

Para garantir que não seja possível este tipo de vulnerabilidade ser explorada para a emissão de votos fraudulentos e válidos, é necessário que o sistema atenda ao Princípio da Independência de Software em Sistemas Eleitorais, que é o item 5 dos requisitos. A estratégia utilizada é a emissão de uma versão modificada da prova auditável pelo eleitor utilizada em urnas de 2ª geração. Inclui informações adicionais que criam um laço entre ela e o registro digital do voto no *blockchain*. Uma sugestão dessa prova é apresentada na Figura 3.

Cada voto é uma transferência de uma carteira associada a uma determinada urna, para uma carteira associada a um determinado candidato. A apuração dos votos é um procedimento instantâneo e consiste em solicitar o balanço das carteiras associadas a cada candidato, atingindo assim o item 1 dos requisitos.

Como o código do contrato é aberto e as transações são asseguradas por criptografia de chaves assimétricas (VOGELSTELLER et al; 2017), um auditor pode verificar as regras de negócio do sistema e verificar a origem de todas as transações realizadas.

Figura 3 – VICE do EtherVoltz



Fonte: Autor, 2017.

Cada transação confirmada é incluída em um bloco na rede *Ethereum*, produz um *hash* que a identifica de forma única (VOGELSTELLER et al; 2017). Uma busca com um explorador de blocos usando este *hash* permite saber a chave pública que a assinou a transação e a chave pública da carteira destino. O EtherVoltz explora este fato para criar um laço entre as provas impressas e os registros digitais ao incluir este *hash* nos VICES. Assim, auditando o *hash* de uma prova impressa, um auditor pode se descobrir de qual carteira de urna um determinado voto saiu e qual candidato recebeu o voto. Se a alguma informação retornada pela máquina virtual divergir do que está impresso no VICE, está configurada a fraude.

#### IV. RESULTADOS

Um sistema de votação independente de software que grava os registros digitais de votos numa base de dados autônoma e resistente a censura. Foi escrito um contrato inteligente na linguagem *Solidity* utilizando o framework *Truffle*. Cada função criada possui testes unitários desenvolvidos utilizando *javascript*, as suites de asserção *mocha.js* e *chai.js* e o cliente RPC *testrpc* para a execução dos testes automatizados.

O contrato inteligente define as seguintes regras, que são imutáveis após hospedagem na rede:

- A transferência de VoltTokens para candidatos só pode ocorrer durante o período eleitoral;
- Apenas endereços de carteiras que representam candidatos podem receber VoltTokens;
- O número total de VoltTokens em circulação é finito e sua quantidade é definida em código;
- Nenhuma nova unidade da moeda pode ser emitida após a criação do sistema.

- Cada urna recebe precisamente o número de VoltTokens correspondente ao número de eleitores que devem votar naquela urna;
- O administrador só pode definir candidatos e urnas antes do período eleitoral;
- O administrador só pode distribuir VoltTokens às urnas antes do período eleitoral.

As regras anteriores são auditáveis por qualquer pessoa através do código fonte do contrato disponibilizado pelo administrador e através do código fonte do contrato inteligente no *blockchain*. O *bytecode* da aplicação em execução, e o *bytecode* resultante da compilação do código fonte anunciado pelo administrador devem ser idênticos.

Todos os VoltTokens são rastreáveis desde o momento de sua emissão através de um explorador de blocos. Roubos de votos ficam registrados permanentemente no *blockchain* assim como a origem e destino dos mesmos e a chave pública responsável pelo ato.

Após a execução de testes em ambiente controlado, a aplicação distribuída foi lançada na rede pública *Rinkeby* para simular uma eleição em funcionamento na internet. Foram criados pares de chaves pública e privada para o Administrador (1), Candidato A (2), Candidato B (3), Urna A (4) e Urna B (5) que são apresentados na Tabela 2. A chave (6) é o endereço e chave pública da aplicação.

Tabela 2 – Chaves públicas das carteiras utilizadas.

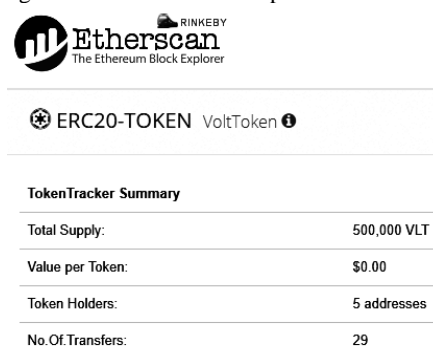
	Chave Públicas
(1)	0x51e6dd45486b5fafeda75595b7501891c9fc54e7
(2)	0x2ec72e4e7846e33bd0cc88cbaecdf4bb01bcd3ff
(3)	0x2330d7654399d22a750bd22b8fc8501a347b7547
(4)	0x0caa969e554a35f1176d739e384045691d21ee64
(5)	0xdc2b8ea73104807285a3fad17c35dcc80e54ba46
(6)	0x063407a72493c8058b415f50076bc990c3927958

Fonte: Blockchain Rinkeby, 2017.

Após a criação da aplicação na rede, o único endereço reconhecido pelo programa é o do administrador e, portanto nenhum endereço de urna ou candidato possui permissão para receber transferências.

Foram enviadas transações assinadas com a chave privada do administrador para adicionar os endereços das urnas e dos candidatos, para que possam emitir e receber votos, respectivamente.

Figura 4 – VoltToken no Explorador Etherscan



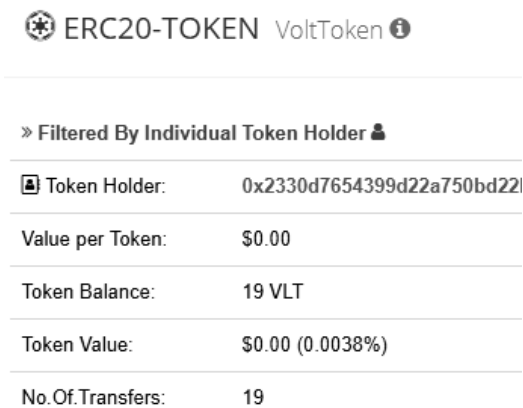
Fonte: Autor, 2017.

A Figura 4 e 5 mostram informações sobre a aplicação no *blockchain* após o período eleitoral. Informações incluem

o número total de *tokens* em circulação, quantos endereços possuem tokens e o número de transferências realizadas.

De maneira similar, após o lançamento da aplicação, o administrador possui custódia de todos os VoltTokens em circulação. Dos 500.000 VoltTokens criados, vinte foram transferidos para a Urna A e 5000 para a Urna B para que as urnas possam assinar transferências de suas próprias carteiras para as carteiras dos candidatos. Dentro do período eleitoral, as duas carteiras assinaram votos para cada candidato, como ocorreria em uma eleição real.

Figura 5 – Total de Votos do Candidato B



» Filtered By Individual Token Holder	
Token Holder:	0x2330d7654399d22a750bd221
Value per Token:	\$0.00
Token Balance:	19 VLT
Token Value:	\$0.00 (0.0038%)
No.Of.Transfers:	19

Fonte: Autor, 2017.

Com o explorador de blocos *rinkeby.etherscan.io*, foram adquiridas informações das transferências de VoltTokens realizadas. Sete destas transferências estão listadas na Tabela 3 em ordem cronológica, sendo que as transferências mais recentes estão no topo. As duas primeiras são a distribuição de VoltTokens realizada pelo administrador antes do período eleitoral e as demais são os votos realizados pelos eleitores dentro do período eleitoral.

Tabela 3 – Transferências de VoltTokens.

Tx Hash	De	Para	Quantidade
0xe7b3...	0x0caa...	0x2330...	1
0x53ff...	0xdc2b...	0x2330...	1
0xad73...	0xdc2b...	0x2330...	1
0x4b41...	0xdc2b...	0x2330...	1
0x3a13...	0xdc2b...	0x2330...	1
0x4d2c...	0x51e6...	0x0caa...	20
0xcd46...	0x51e6...	0xdc2b...	5000

Fonte: Autor, 2017.

A coluna “Tx Hash” apresenta o *hash* de cada transação realizada e incluída em um bloco por um minerador da rede *Rinkeby*.

## V. DISCUSSÃO

### 4.1 – O Procedimento do Voto

Do ponto de vista do eleitor, o voto ocorre da mesma forma que em uma urna de segunda geração comum, exceto que a impressão do VICE ocorre em duas etapas.

A seguir são detalhados eventos relevantes que ocorrem durante a emissão de um voto.

1. O eleitor, após ser autorizado pelo mesário, digita o código do candidato.

2. A impressora imprime o VICE que fica visível para que o eleitor possa conferir seu voto.
3. Se os dados estão corretos o eleitor pressiona “confirma”.
4. A urna envia uma transação à aplicação que está no endereço publicado na cerimônia oficial para transferir 1 VoltToken da carteira da urna para a carteira que representa o candidato.
5. Após a confirmação da transação, a urna recebe um *hash* que identifica unicamente este voto no *blockchain*.
6. A impressora imprime este *hash* no VICE.
7. O VICE é cortado e cai em uma caixa lacrada.

Após o período eleitoral existem dois registros de cada voto contado. Um deles é o registro digital do voto que está gravado no *blockchain* da máquina virtual *Ethereum*. O outro é o voto impresso conferível pelo eleitor.

Os registros digitais dos votos podem ser solicitados por qualquer pessoa a qualquer momento para auditoria e possuem as seguintes informações:

- De qual urna o voto foi emitido.
- Qual candidato recebeu o voto.
- O *hash* que identifica unicamente esta transação no *blockchain*.

Já os registros físicos do voto, ficam sobre controle do administrador da eleição e possuem as seguintes informações:

- De qual urna o voto foi emitido.
- Qual candidato recebeu o voto.
- O *hash* que identifica unicamente esta transação no *blockchain*.

A apuração dos votos é instantânea e consiste em apenas solicitar ao computador mundial, o balanço das carteiras criadas para receberem os votos dos candidatos.

### 4.2 – Auditorias

Em sistemas eleitorais de primeira, segunda e terceira geração, todo o processo de auditoria precisa necessariamente envolver o administrador, já que este tem custódia de todas as provas do processo eleitoral. Já no sistema proposto, o administrador do sistema eleitoral não possui custódia nem do programa que realiza a transferência dos VoltTokens que representam os votos e nem do banco de dados que armazena os registros digitais de voto. Consequentemente, parte da auditoria pode ocorrer sem a necessidade do envolvimento do administrador. De fato, utilizando apenas um navegador de blocos, qualquer pessoa pode analisar as transferências de VoltTokens em busca de endereços e carteiras que não foram anunciados em cerimônia oficial pelo administrador.

Uma auditoria de uma urna poderia ser conduzida da seguinte forma:

- A. O auditor utiliza um explorador de blocos como o *rinkeby.etherscan.io*, para analisar todas as transações realizadas pela urna que está sendo auditada;
- B. O auditor solicita ao administrador da votação, a caixa contendo os votos impressos conferíveis pelo eleitor, da mesma urna;
- C. O auditor compara as duas provas, em busca de provas inconsistentes.

Alguns exemplos de inconsistências nas provas, e que caracterizam fraudes são listados:

- O número de VICE's na caixa entregue pelo auditor é diferente do número de registros de voto digital retornados pela máquina virtual *Ethereum*;
- Algum registro digital de voto retornado pela máquina virtual *Ethereum* não possui seu VICE associado na caixa que o administrador entregou ao auditor;
- Algum VICE não possui um *hash* válido.

Um hash impresso no VICE é considerado válido se:

- Ele existir no *blockchain Ethereum*;
- O endereço do remetente for igual ao da urna que está sendo auditada;
- O endereço da carteira do candidato for o mesmo que o impresso no VICE correspondente;
- O endereço do contrato da criptomoeda for o mesmo que o publicado em cerimônia oficial.

## VI. CONCLUSÃO

EtherVoltz é uma proposta que visa descentralizar o processo eleitoral e de auditorias ao transferir a responsabilidade de gerenciar dos registros digitais de votos a um programa que executa em uma máquina virtual que não possui autoridade central, é imutável e resistente a censura. Ao transformar o voto do eleitor em uma criptomoeda, o sistema imediatamente ganha todas as propriedades de segurança do protocolo de consenso e de disponibilidade da rede *peer-to-peer* nativa da plataforma.

Votos são finitos e cada voto é rastreável desde a sua emissão e distribuição para as carteiras das urnas até a carteira que representa o candidato.

Embora ainda exista a necessidade da emissão e controle das provas impressas para garantir o Princípio da Independência de Software ao sistema, a separação do destino das duas provas produzidas no momento do voto dá aos eleitores poder de auditoria com limitada necessidade do envolvimento de intermediário.

## VII. REFERÊNCIAS BIBLIOGRÁFICAS

BRUNAZO, A.F. Modelos e Gerações dos Equipamentos de Votação Eletrônica. Disponível em: <<http://www.brunazo.eng.br/voto-e/textos/modelosUE.htm>>. Acesso em: 27 jul. 2017.

CHOHAN, U.W. Criptocurrencies: A Brief Thematic Review. 2017. Disponível em: <[https://papers.ssrn.com/sol3/papers.cfm?abstract\\_id=3024330](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3024330)>. Acesso em: 11 jul. 2017.

NAKAMOTO, S. Bitcoin: A Peer-to-Peer Electronic Cash System. Disponível em: <<https://bitcoin.org/bitcoin.pdf>>. Acesso em: 4 mai. 2017.

REITWIESSNER, C. et al. Solidity Documentation. 2017. Disponível em: <<https://solidity.readthedocs.io/en/latest/>>. Acesso em: 6 jun. 2017.

RIVEST, R.L.; WACK, J.P. On the notion of “software independence” in voting systems. Cambridge, MA. MIT. 2006. Disponível em: <<https://people.csail.mit.edu/rivest/pubs/RW06.pdf>>. Acesso em: 22 jul. 2017.

SÉRVULO, S.S. et al. 1º Relatório do Comitê Multidisciplinar Independente. 2010. Disponível em: <<http://www.votoseguro.org/textos/CMind-1-Brasil2010.pdf>>. Acesso em: 15 jul. 2017.

VOGELSTELLER, F. et al. What Is Ethereum. 2017. Disponível em: <<https://github.com/ethereum/wiki/wiki/FAQ>>. Acesso em: 8 set. 2017.

VOGELSTELLER, F. et al. Ethereum White Paper. 2017. Disponível em: <<https://github.com/ethereum/wiki/wiki/Glossary>>. Acesso em: 9 jul. 2017.

VOGELSTELLER, F. et al. What Is Ethereum. 2017. Disponível em: <<https://github.com/ethereum/wiki/wiki/What-is-Ethereum>>. Acesso em: 3 jun. 2017.

VOGELSTELLER, F. et al. Ethereum Glossary. 2017. Disponível em: <<https://github.com/ethereum/wiki/wiki/White-Paper>>. Acesso em: 20 ago. 2017.

WOOD, G. Ethereum: A Secure Decentralised Generalised Transaction Ledger. p. 2-6. Disponível em: <<http://gavwood.com/paper.pdf>>. Acesso em: 27 jul. 2017.

YUAN B; LIN, W; MCDONNELL, C. Blockchains and electronic health records. p. 3. 2017.

## VIII. COPYRIGHT

Direitos autorais: O(s) autor(es) é(são) o(s) único(s) responsável(is) pelo material incluído no artigo.