ISSN 1677-8464

Foto:Pexels

COMUNICADO TÉCNICO

130

Campinas, SP
Dezembro/2018

Embrapa

Modelo de rastreamento bovino via *Smart Contracts* com tecnologia Blockchain

Inácio Henrique Yano
Edgard Henrique dos Santos
Alexandre de Castro
Ivan Bergier
Patrícia Menezes Santos
Stanley Robson de Medeiros Oliveira
Urbano Gomes Pinto de Abreu

Modelo de rastreamento bovino via Smart Contracts com tecnologia Blockchain[1]

A Embrapa estabelece em seu documento Visão 2014-2034 que:

> O avanço nas tecnologias da informação e comunicação (TICs) reduziu as barreiras físicas, políticas e culturais entre as nações. Globalizou o acesso às matérias-primas, aos bens e aos serviços e deu a todas as pessoas o poder para influenciar os rumos do desenvolvimento tecnológico e da formatação de bens e serviços. Munida de equipamentos e sensores, sem limites de conexão, a população mundial exerce seu poder de escolha em escala global, consubstanciando a realidade *Big Data*, em que um grande volume de dados e informações sobre tendências e demandas reflete, entre outros, manifestações de caráter cultural e psicossocial. Investir intensivamente em ferramentas e processos que apoiem previsões sobre as necessidades tecnológicas e sobre a demanda futura por bens e serviços, cada vez mais difusa e dinâmica, é essencial para as organizações de pesquisa e inovação. (Embrapa, 2014, p.11).

Em especial, o documento descreve um contexto agropecuário marcado pela Era do *Big Data*, com a geração de grandes volumes de dados que necessitam ser organizados, armazenados e processados para a geração de novos conhecimentos.

> O mundo já vive a era *Big Data*, com a possibilidade de gerar, medir, coletar e armazenar assombrosas quantidades de dados, que são a matéria-prima do conhecimento. Uma vasta gama de tecnologias emergentes ajuda as organizações a extraírem valor desses grandes conjuntos de dados, o que torna possível, por exemplo, inferir padrões de comportamento e de consumo e ajustar o design e a logística de entrega de produtos e de serviços para cada indivíduo, com enormes ganhos de eficiência operacional e econômica. Daqui para o futuro, o setor privado vai usar *Big Data* para multiplicar o acesso a serviços e bens de consumo. O setor público vai usá-lo para suporte à formulação, melhoria e implementação de

[1] Inácio Henrique Yano, tecnólogo em Processamento de Dados e Economista, doutor em Engenharia Agrícola, analista da Embrapa Informática Agropecuária, Campinas, SP. Edgard Henrique dos Santos, analista de sistemas, analista na Embrapa Informática Agropecuária, Campinas, SP. Alexandre de Castro, físico, doutor em Ciências, pesquisador da Embrapa Informática Agropecuária, Campinas, SP. Ivan Bergier, biólogo, doutor em Ciências, pesquisador da Embrapa Pantanal, Corumbá, MS, Patrícia Menezes Santos, engenheira agrônoma, doutora em Ciência Animal e Pastagens, pesquisadora da Embrapa Pecuária Sudeste, São Carlos, SP. Stanley Robson de Medeiros Oliveira, bacharel em Ciência da Computação, Ph.D. em Ciência da Computação, pesquisador da Embrapa Informática Agropecuária, Campinas, SP. Urbano Gomes Pinto de Abreu, médico veterinário, doutor em Zootecnia, pesquisador da Embrapa Pantanal, Corumbá, MS.

> políticas públicas em áreas sensíveis tais como medicina, saúde pública, produção de alimentos e meio ambiente. Na agropecuária, a Era *Big Data* ainda irá impactar o melhoramento genético, a previsão de clima, a agricultura de precisão, o entendimento da dinâmica dos mercados, entre outros aspectos. (Embrapa, 2014, p. 52-53).

No âmbito do agronegócio, a realidade *Big Data*, associada às novas tecnologias como cadeia de blocos – *Blockchain*, poderá facilitar o registro distribuído de operações de rastreamento de produtos agrícolas e transações de *commodities* visando à descentralização como medidas de segurança. O advento do *blockchain* como uma ferramenta para o agronegócio pode facilitar a adoção dos chamados Contratos Inteligentes (*Smart Contracts*), com critérios de qualidade e de conformidade socioambiental que seriam certificados ao longo de todo o processo, o que, do ponto de vista dos mercados consumidores interno e no exterior, poderá impulsionar a adesão do Sistema Brasileiro de Identificação e Certificação de Bovinos e Bubalinos (Sisbov) do Ministério da Agricultura, Pecuária e Abastecimento (Mapa) por produtores rurais, que hoje ainda ocorre de forma voluntária (Brasil, 2017).

Blockchain

Blockchains são bases de registros e de dados distribuídos e compartilhados que têm a função de criar um índice global para todas as transações que ocorrem em um determinado mercado. Essa tecnologia está por trás de criptomoedas (moedas digitais) como *bitcoin, ether, ripple,* entre tantas outras, mas seu uso como conduíte transacional vai muito além. Por conceito e desenho, trata-se de uma tecnologia que permite – e pede – a formação de parcerias colaborativas entre uma rede de participantes, em um ou mais mercados, e até mesmo em diferentes geografias.

Merkle (1990) patenteou sua ideia de estrutura de dados chamada de *Merkle tree* ou árvore de dispersão. Essa

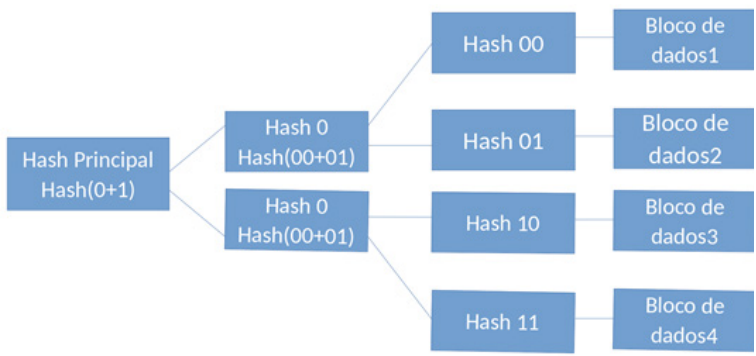

Figura 1. Árvore de Merkle.

estrutura permite verificar erros ou alterações em blocos de dados transmitidos em redes ponto a ponto por meio de uma função de dispersão ou hash (Figura 1).

A função *hash*, ou função de dispersão, é uma fórmula matemática que possibilita o mapeamento de dados de tamanho arbitrário para um conjunto de tamanho fixo. Dessa forma é possível verificar alterações mínimas em um bloco de dados volumoso, examinando a *hash* gerada.

Utilizando a fórmula SHA256 (Penard; Werkhoven, 2008), por exemplo, no texto "Bloco de dados 1", obtêm-se a *hash* "CEB232941...". Alterando para o texto "Bloco de dados 2" produz-se uma nova hash. "BFB13C6D8..." (Figura 2).

O aperfeiçoamento do uso de árvores de dispersão foi proposto por Haber e Stornetta (1991), como solução para a certificação da data de criação e de alteração de arquivos de mídia, cujo incremento de produção foi impulsionado pela popularização dos computadores pessoais no início dos anos 90. A ideia primitiva de *blockchain* surge com a geração de hashes não apenas para documentos de forma individual, mas para o conjunto de blocos.

A forma atual de *blockchains* tem origem no trabalho de Nakamoto (2009), um recôndito cientista que ampliou e publicou o conceito contemporâneo de criptomoeda. Nessa concepção, os blocos de dados trazem informações de transações de compra e venda, semelhantes a um livro caixa, que é disponibilizado para qualquer pessoa. As inscrições nesse livro são chamadas de blocos e o conjunto delas é chamado de *blockchain*.

As validações das transações são feitas a partir de assinaturas digitais e posteriormente o conjunto de dados passa pela função geradora da *hash* e a sequência criada é inserida no próximo bloco (Figura 3).

A validação dos dados, no caso específico de criptomoedas, apresenta ainda uma etapa chamada prova de trabalho ou PoW (Proof-of-work). A PoW foi criada por Dwork e Naor (1993) como forma de se evitar o envio de spam de e-mail. Consiste em solicitar uma certa tarefa com custo computacional elevado, porém de fácil verificação.

A PoW de *bitcoins* é a inserção de um número de zeros no início da hash a partir da inclusão de um número no final do bloco, chamado de "nonce". Ao alterar o "nonce", toda a *hash* é modificada. Pelo método de tentativa e erro, vários números são testados até que se obtenha a quantidade de zeros especificada para o início da *hash*. A solução da PoW autoriza a criação de um novo bloco. Como recompensa, o realizador do prova de trabalho recebe um conjunto definido de criptomoedas.

Atualmente o uso da tecnologia *blockchain* vai além das criptomoedas, sendo utilizado em infraestruturas públicas, processos eleitorais (Money, 2018)

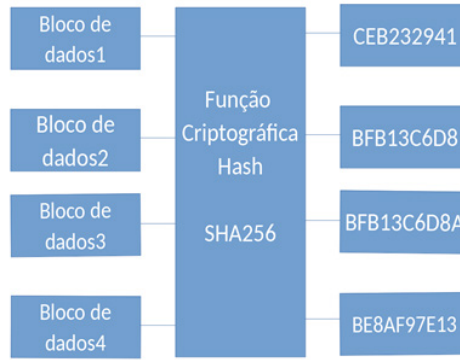

Figura 2. Função *hash* de blocos de dados.

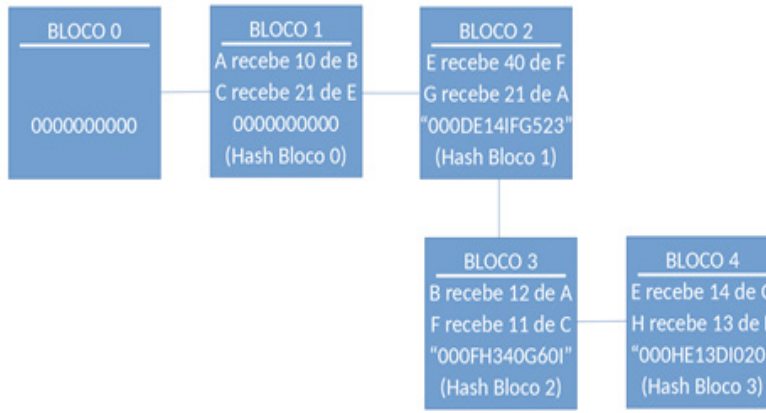

Figura 3. Exemplo de *Blockchain.*

e cadeia de alimentos de supermercados (Carrefour, 2018).

O *blockchain* reduz o custo e a complexidade de transações entre empresas por meio da criação de redes eficientes e altamente seguras na qual praticamente qualquer coisa de valor pode ser monitorada e negociada, sem a dependência de um ponto de controle centralizado. No mundo das finanças, as redes de blockchain podem permitir que comércios de valores mobiliários sejam liquidados em minutos em vez de dias. No mundo do comércio, essas redes podem facilitar o gerenciamento da cadeia de suprimentos e permitir que o fluxo de mercadorias e de pagamentos seja rastreado e registrado em tempo real (Embrapa, 2018).

Vale ressaltar que a tecnologia distribuída do *blockchain* tem potencial para minimizar a assimetria de informação,

responsável por imperfeições de mercado. A pequena produção vende seus produtos por preço muito inferior ao da grande produção, e compra os insumos por preço mais elevado e, portanto, deixa de adotar novas tecnologias pela falta de lucratividade. Sem a adoção de tecnologia não há como resolver o problema da pobreza rural pela agricultura (Alves et al., 2016). Países em desenvolvimento caracterizam-se por apresentarem maior grau de incerteza e de risco, sendo frequentemente sujeitos a choques e a mudanças de diversas ordens — de política econômica, de política, marcos regulatórios, etc. Ademais, a debilidade dos arranjos institucionais nessas economias dificulta o desenvolvimento adequado dos mercados e a atenuação de suas ineficiências (Stiglitz, 1989). Sob esse prisma, temas como desemprego, contratos de seguro, racionamento de crédito, regime de parceria na agricultura, e crises financeiras, são questões cujos efeitos das informações privadas sobre as decisões financeiras, e a existência de instituições financeiras, corroboram para a assimetria da informação (Aldrighi, 2006). O *blockchain*, portanto, pode ser uma alternativa de integrar a pequena e a grande produção mitigando essas distorções e assimetrias.

Neste comunicado técnico, a implementação de um sistema baseado em *blockchain* para rastreamento bovino é apresentada. A proposta visa a utilização da plataforma *Ethereum*, cujo ambiente de desenvolvimento *Remix* permite montar as transações envolvidas via linguagem de programação *Solidity*. Na plataforma *Ethereum* (Ethereum project, 2018), os blocos encadeados carregam um determinado conteúdo junto a uma assinatura *hash* digital, de forma que o bloco seguinte sempre contém a função *hash* do bloco anterior e seu próprio conteúdo gerando, assim, sua própria *hash:*

Bloco 1 = conteúdo = hash 0
Bloco 2 = conteúdo + hash 0 = hash 1
Bloco 3 = conteúdo + hash 1 = hash 2

Todas as informações dos blocos são escritas e gravadas em um livro-razão digital (*ledger*) e depois de escritas não podem mais ser apagadas. Eventualmente, se qualquer conteúdo do bloco for alterado, a função *hash* também será alterada. É importante destacar que em uma cadeia padrão de blocos existem nós mineradores constituídos de participantes que verificam se o bloco escrito é válido. Sempre que um participante validar um bloco, recebe uma recompensa na forma da moeda digital corrente da plataforma, no caso da *Ethereum*, o ether. Essa recompensa é o pagamento pelo custo de processamento dos cálculos matemáticos que asseguram que o *hash* criptográfico do bloco é válido. Contudo, nada impede que a validação seja realizada por um sistema hospedado em uma infraestrutura governamental que não exija recompensa para validar e armazenar as transações.

O código (contrato) é armazenado na cadeia de blocos e é executado como parte de cada transação, em que todos os dados armazenados são funções *hashes* criptográficas, uma função

matemática considerada praticamente impossível de inverter, ou seja, de recriar o valor de entrada a partir de seu valor de saída. Por segurança criptográfica, o *Ethereum* utiliza a função Secure Hash Algorithm V. 3 (SHA3) em substituição ao padrão SHA-2 cipher.

Exemplo de rastreamento bovino com *Smart Contract*

Este exemplo visa ilustrar e compreender o funcionamento de um *Smart Contract*. Consequentemente, pode ainda não retratar a realidade de um rastreamento bovino da melhor maneira possível. No entanto, permite vislumbrar e testar regras e funções, de forma que no futuro seja viável desenvolver contratos mais complexos e que se aproximem mais de condições ótimas de funcionamento. O código foi escrito na linguagem *Solidity*, que é uma das linguagens mais utilizadas para este fim e pode ser visto na Figura 4.

```
1  pragama solidity ^0.4.25;
2  contract BoiZ {
3
4  address faz_cria_A;
5  address faz_engorda_B;
6  address frigorifico_C;
7  bool public venda_magro =
true;
8  bool public venda_gordo =
false;
9  bool public contrato_finalizado
= false;
10 uint constant unidade_moneta-
ria = 1 ether;
11 uint public preco_magro_wei =
0;
12 uint public preco_gordo_wei =
0;
13 string public dados_cadastrais;
14 string[] public eventos = new
string[](10);
15 uint public proximo_evento = 0;
16
17 constructor() public payable {
18 faz_cria_A = msg.sender;
19 dados_cadastrais = "BoiZ
nascido em 29/02/2016, identficação
0649, filho de ..."
20 ;
21 }
22
23 function eventos(string da-
dos_eventos) public {
24 require(contrato_finalizado ==
false);
25 eventos[proximo_evento] =
dados_eventos;
26 proximo_evento += 1;
27 }
28
29
30 function precoBoiMagro(uint
amount) public {
31 require(venda_magro == true);
32 require(faz_cria_A == msg.
sender);
33 preco_magro_wei = amount *
unidade_monetaria;
34
35 }
36
37
```

```
38  function compraBoiMagro()
public payable {
39  require(preco_magro_wei > 0);
40  require(msg.value ==
(preco_magro_wei));
41  require(venda_magro == true);
42
43  faz_engorda_B = msg.sender;
44
45  require(faz_cria_A.send(msg.
value));
46


47  venda_magro = false;
48  venda_gordo = true;
49
50  }
51
52  function precoBoiGordo(uint
amount) public {
53  require(faz_engorda_B == msg.
sender);
54  require(venda_gordo == true);
55  preco_gordo_wei = amount *
unidade_monetaria;
56
57  }
58
59  function compraBoiGordo()
public payable {
60  require(preco_gordo_wei > 0);
61  require(msg.value ==
(preco_gordo_wei));
62  require(venda_gordo == true);
63
64  frigorifico_C = msg.sender;
65
66  require(faz_engorda_B.sen-
d(msg.value));
67
68  venda_gordo = false;
69  contrato_finalizado = true;
70  }
71
72  }
```

Figura 4. Linhas de Código do *Smart Contract* BoiZ.

Neste exemplo, o *Smart Contract* BoiZ foi dividido em sete atividades conforme a Estrutura Analítica do Contrato da Figura 5.

O contrato inicia com a versão do compilador e na segunda linha com o nome do contrato, neste caso BoiZ, nas linhas seguintes seguem as atividades, sendo que da primeira até a sexta atividade, a execução ocorre sequencialmente. A primeira atividade é para definição das principais variáveis do contrato, e a segunda até a sétima atividade são funções do contrato, ou seja, modificam o estado ou os valores das variáveis.

As variáveis da primeira atividade (“Define Variáveis”) estão relacionadas abaixo:

1) as variáveis das linhas 4, 5 e 6 servem para guardar o endereço dos participantes do contrato ou *Externally owned account* (EOA), neste exemplo: fazenda de cria, fazenda de engorda e frigorífico;

2) as variáveis das linhas 7 e 8 foram criadas para garantir que o boi tenha o preço estabelecido e seja vendido somente pela fazenda que detém a sua posse. Já a variável da linha 9 foi criada para garantir que o boi não possa ser

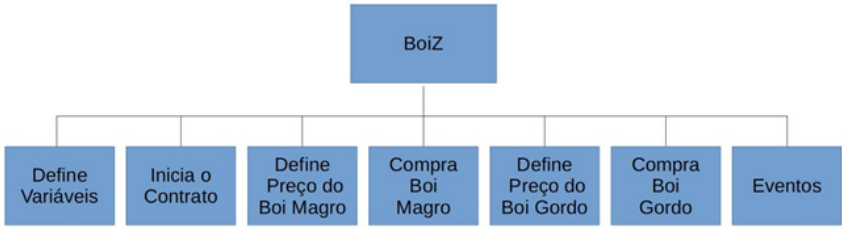

Figura 5. Estrutura Analítica do Smart Contract BoiZ.

mais vendido depois de ter sido adquirido pelo frigorífico, momento em que este exemplo de contrato termina;

3) a variável da linha 10 serve para conversão de moedas para estabelecimento do preço do boi, de ether, que é maior unidade de valor de uma rede Ethereum, para wei, que é a menor unidade de valor, porque o contrato somente trabalha em unidades de wei;

4) as variáveis das linhas 11 e 12, servem para guardar o preço do boi em wei;

5) a variável na linha 13 existe para tornar público os dados cadastrais do boi no início da vigência do contrato (data de nascimento, código de identificação, sexo, etc);

6) e, finalmente, as variáveis das linhas 14 e 15, foram criadas para armazenar e para controlar eventos que não fazem parte das transações comerciais, servindo apenas para qualificar o boi (vacinações, tratamentos veterinários, doenças, acidentes, etc).

Além dessas variáveis, existem as variáveis que trazem dados dos participantes no momento em que estão executando as transações, o nome dessas variáveis inicia-se com o prefixo "msg.", por exemplo:

a) "msg.sender" é o endereço do participante corrente, ou seja, aquele que está exercendo alguma ação no contrato (implementa o contrato, define o preço, compra, insere dados sobre a vida do boi);

b) "msg.value" é o valor a ser transferido em transações comerciais.

Na segunda atividade ("Inicia o Contrato"), linhas 17 a 21, o contrato é implementado e, consequentemente, disponibilizado para transações (*Deploy*); nessa atividade também é atribuído o endereço do participante corrente ao endereço da fazenda de cria, que é a proprietária do boi nesse momento; nessa etapa também são inseridos os dados cadastrais do boi.

A terceira atividade ("Define o Preço do Boi Magro"), linhas 30 a 35, é quando a fazenda de cria estabelece o preço em ether do boi para venda, e posteriormente esse preço é convertido em wei. Nesta atividade existem dois

[2] Disponível em:<https://remix.ethereum.org>.

pré-requisitos (require), o primeiro deles é que o boi, no estado magro, ainda não tenha sido vendido (venda_magro == *true*) e a segundo é que somente o proprietário, neste caso a fazenda de cria, é que esteja definindo o preço (faz_cria_A == msg.sender), ou seja, o participante corrente seja a fazenda de cria.

Uma vez que o boi, em seu estado magro, esteja disponível para venda, isto é, já esteja com o preço de venda estabelecido, a quarta atividade (“Comprar Boi Magro”), linhas 38 a 50, já pode ser exercida. O comprador (fazenda de engorda), para realizar a compra, terá de enviar a quantidade de ether igual ao valor estabelecido pelo vendedor (fazenda de cria), caso contrário a transação não poderá ser concluída. Uma vez satisfeitos os requisitos, o endereço do participante corrente (comprador) é atribuído à fazenda de engorda e o valor da compra é enviado para a fazenda de cria.

A quinta e a sexta atividades, linhas 52 a 72, são repetições da terceira e da quarta atividades, em que somente são trocados os participantes: neste caso, o vendedor é a fazenda de engorda e o comprador é o frigorífico. No final da sexta atividade o contrato também é finalizado, de forma que somente estão disponíveis funções de consulta.

A sétima atividade é “Eventos”, linhas 23 a 27, que pode ser utilizada para o cadastro de eventos que não estejam diretamente relacionados à comercialização do boi e pode ser utilizada em qualquer momento enquanto o contrato esteja ativo, ou seja, não tem atividade predecessoras ou sucessoras.

Simulação de Rastreamento Bovino Utilizando *Smart Contract*

A simulação foi realizada no ambiente de desenvolvimento Remix[2] , que é uma ferramenta desenvolvida para trabalhar com a linguagem Solidity, sendo de fácil interação com o contrato e com o acompanhamento das variações dos valores das variáveis. Antes da simulação deve-se compilar o contrato para assegurar que não existem erros que impeçam sua execução (Figura 6).

Uma vez que o contrato foi compilado, a simulação pode ser iniciada clicando-se na aba “Run”, conforme Figura 7. Nessa mesma fiigura também é possível notar que o ambiente de desenvolvimento Remix fornece cinco contas EOA com saldo de 100 ether cada para a realização de testes. Neste exemplo usaremos somente as três primeiras, porque são somente três participantes. O primeiro deles será a fazenda de cria e o endereço que utilizaremos será 0xca3...a733c, que na forma extensa é 0xca35b7d915458ef540ade6068dfe2f44e8fa733c. Neste momento como nenhuma transação foi feita ainda, o contador de transações está zerado (Figura 6).

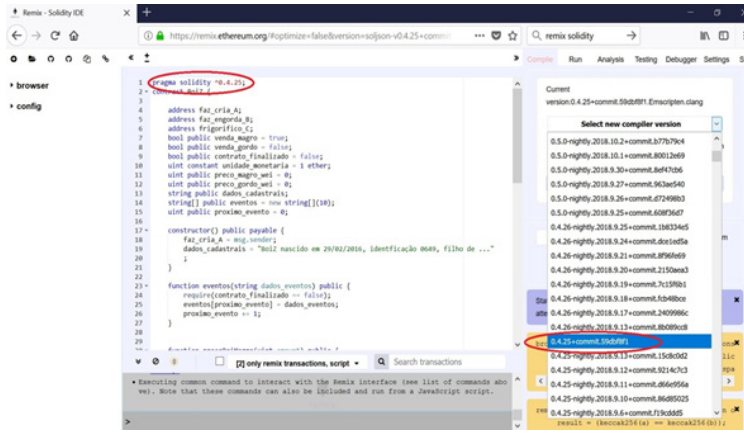

Figura 6. Ambiente de desenvolvimento Remix, selecionando compilador.

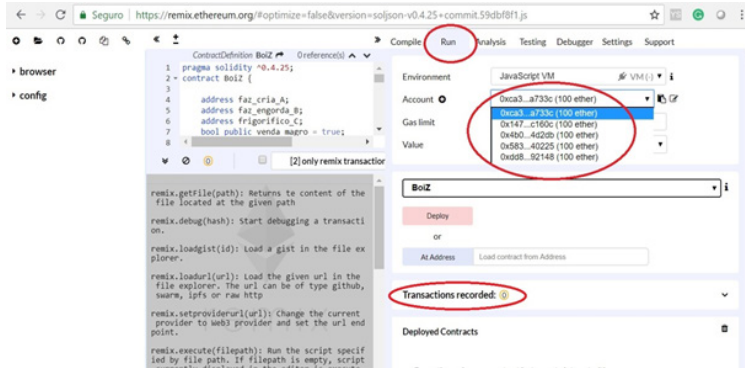

Figura 7. Ambiente de desenvolvimento Remix, selecionando a aba *Run*.

A primeira transação do contrato é a sua implementação para disponibilização para utilização, apertando-se o botão Deploy, com a conta 0xca3... a733c selecionada, que é fazenda de cria e proprietária do boi. O Deploy é a primeira transação do contrato e o sucesso da transação pode ser notado pelo sinal de checagem em verde no canto inferior esquerdo da tela; pode-se notar, também, na Figura 8 a alteração do número de transação para 1, além de o saldo da conta 0xca3...a733c ter sido reduzido. A redução de saldo ocorre em todas as transações, devido à necessidade de mineração de wei para saldar os custos da transação.

Clicando-se na transação é possível ver em detalhes todas as informações relativas à transação (Figura 9), entre elas:

a) status: se a transação foi bem-sucedida ou não;

b) o *hash* da transação;

c) o endereço do contrato;

d) a conta que originou a transação do contrato, neste caso, a conta 0xca3... a733c (fazenda de cria);

e) to: qual a operação/atividade executada;

f) a quantidade máxima de wei (gas) que a conta 0xca3...a733c está disposta a pagar para validar a transação, neste caso 3 milhões de wei; caso o custo para validação da transação ultrapasse o gas, a transação não será concluída;

g) o custo da transação (validação), que representa a redução em wei do saldo da conta 0xca3...a733c, neste caso, 1.001.536 wei;

h) o custo da execução, que é parte do custo da transação.

Existem alguns outros itens listados na Figura 9, que serão explicados em exemplos que envolvam entrada e saída de dados ou valores.

Uma vez tendo sido implementado (*Deploy*) o contrato, é possível consultar variáveis e executar funções (atividades) do contrato. Para visualização das variáveis e funções basta clicar na área da memória do endereço do contrato (Boiz at 0x692...77b3a (memory)). A Figura 10 ilustra as variáveis em azul e as funções em vermelho. Nela a variável "dados_cadastrais"; já possui valor que foi inserido no momento do *Deploy*. Toda consulta ao contrato também fica registrada e é representada por um *call* em lilás, entretanto, as consultas não incrementam o contador de transações, que neste momento, continua em 1, devido ao *Deploy*, que foi a primeira transação.

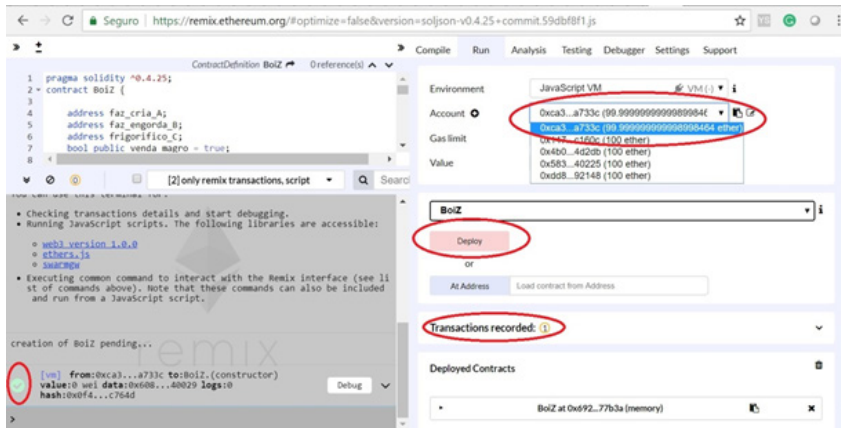

Figura 8. *Deploy* do *Smart Contract* BoiZ.

Figura 9. Detalhes da transação de *Deploy*.

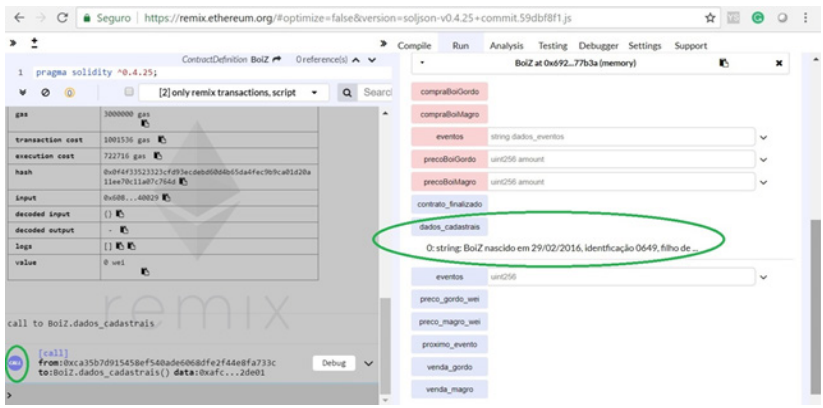

Figura 10. Consulta o conteúdo da variável dados_cadastrais.

Como o contrato já está disponível (implementado) é possível realizar as transações (funções) do contrato. Neste exemplo a fazenda de cria irá cadastrar dois eventos:

a) “Em 28/02/2017, vacinação contra febre aftosa (lote: L001, dose: D001)”;

b) “Em 28/02/2018, aplicação de antibiótico para tratar ... (lote: L002, dose: D002)”.

O cadastro dos dois eventos, assim como, a consulta desses cadastramentos, podem ser verificados nas Figuras 11 e 12, respectivamente. O índice para

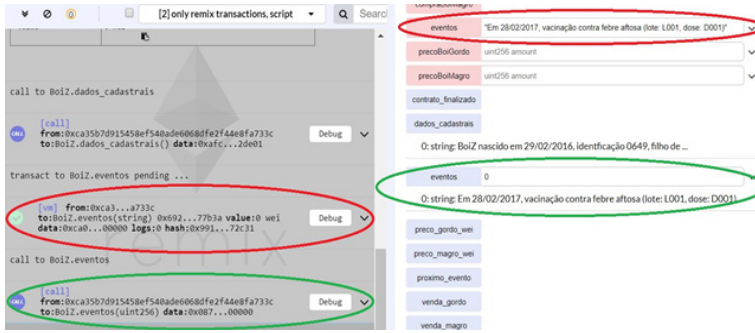

Figura 11. Consulta e cadastramento do evento 0.

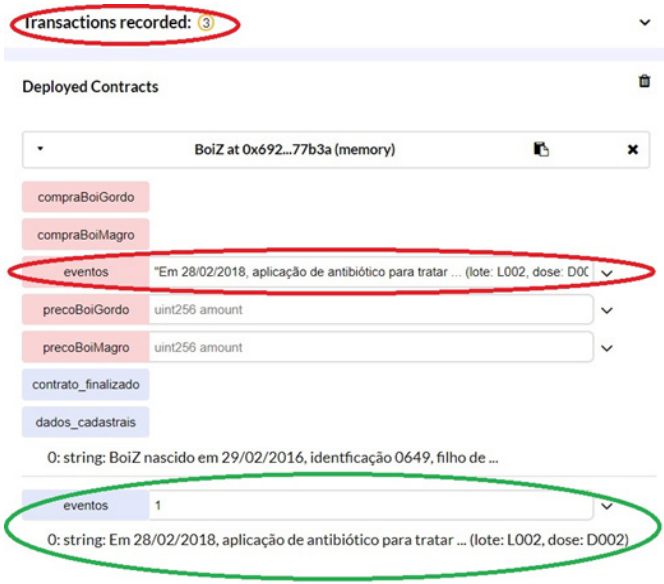

Figura 12. Consulta e cadastramento do evento 1.

input	0xca0...00000
decoded input	{ "string dados_eventos": "Em 28/02/2018, aplicação de antibiótico para tratar ... (lote: L002, dose: D002)" }
decoded output	{}
logs	[]
value	0 wei

Figura 13. Exemplo de transação com entrada de dados.

acessar os eventos inicia-se com zero e por isso, os eventos cadastrados são 0 e 1. Essas duas transações, que envolvem alteração em variáveis, incrementam o contador de transações, que passa a ter o valor 3 (Figura 12).

O cadastro de eventos é um exemplo de transação em que existem dados de entrada (Figura 13), diferentemente da transação de Deploy, na qual o campo de decoded input está vazio.

O passo seguinte, deste exemplo, é a venda do boi magro, neste caso, a fazenda de cria terá primeiro que definir um preço para o mesmo. Supondo que o valor escolhido seja de 1 ether, irá utilizar a função “precoBoiMagro” para este fim. A Figura 14 apresenta a inserção do valor de 1 ether na função “precoBoiMagro”, pela fazenda de cria, e, também, a consulta do valor corresponde em wei na variável “preco_magro_wei”.

Com o preço definido, o boi magro está disponível para venda, até este momento todas as transações foram realizadas somente pela fazenda de cria (0xca3...a733c). Neste exemplo, a segunda conta da plataforma Remix (0x147...c160c) realizará a compra. Para tanto, é necessário selecionar a conta e preencher o valor (em ether) a ser enviado ao contrato, sendo que o valor “Gas limit” é o valor máximo em wei que a conta 0x147...c160c está disposta a pagar para validar (minerar) a transação e está em 3 milhões de wei (Figura 15).

Para concluir a compra basta clicar na função “compraBoiMagro”. Na Figura 16 é possível notar que a execução teve êxito e que o contador de transações passou a ter o valor 5, uma vez que a definição do preço do boi magro foi a quarta transação.

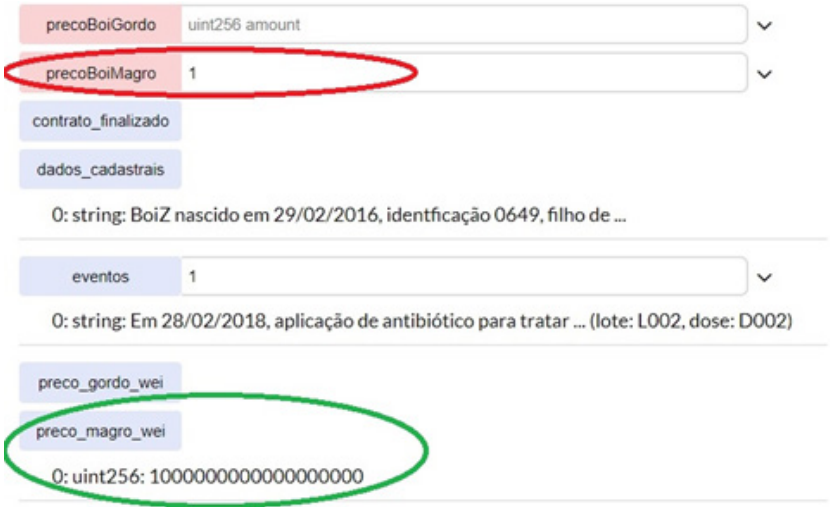

Figura 14. Definição do preço do boi magro em ether e consulta do preço em wei.

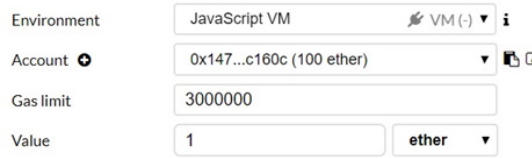

Figura 15. Seleção de conta e envio de valores para o contrato para compra do boi magro.

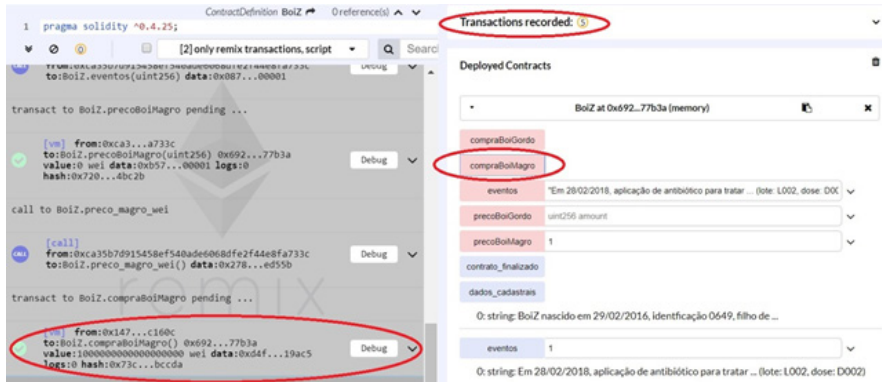

Figura 16. Execução da função compraBoiMagro e incremento do contador de transações.

A Figura 17 ilustra os dados da transação de venda do boi magro, na qual foi enviado o valor da compra em wei como dado de entrada.

Após a realização da compra é possível consultar os novos saldos das contas envolvidas (Figura 18), sendo que a conta 0xca3...a733c (fazenda de cria) teve incremento de 1 ether pela venda do boi magro e a conta 0x147...c160c, que agora passa a ser a fazenda de engorda, teve decremento de 1 ether, além de 61041 wei, em razão da necessidade de validação da transação.

A finalização deste exemplo ocorre com a decisão de venda do boi pela fazenda de engorda, definindo o preço em 3 ether na função “precoBoiGordo“ (Figura 19). Definição da conta 0x4b0...4d2db como sendo a conta do frigorífico, que irá enviar 3 ether para realizar a compra do boi gordo (Figura 20). E, por último, execução da função “compraBoiGordo”, momento em que o contrato é finalizado e não será mais possível fazer alteração nas variáveis do contrato (Figura 21).

Terminado o contrato é possível consultar o saldo das contas (Figura 22) e, caso, alguma conta tente fazer nova compra ou alterar o valor de alguma outra variável ocorrerá um erro, representado por um “x” vermelho (Figura 23).

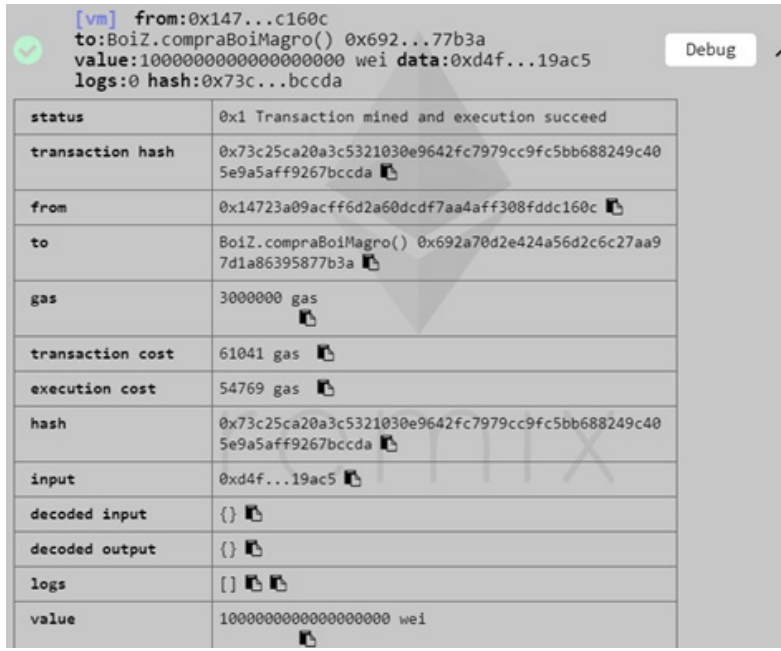

Figura 17. Exemplo de valor como dado de entrada.

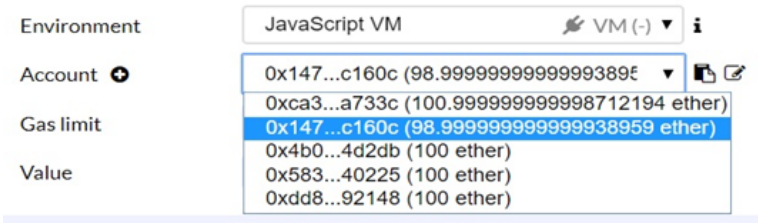

Figura 18. Novos saldos das contas envolvidas na transação de compra do boi magro.

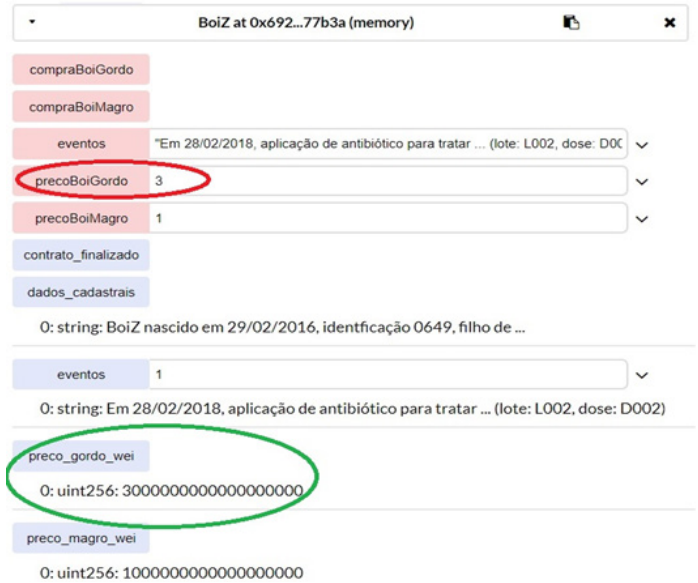

Figura 19. Definição do preço do boi gordo em ether e consulta do preço em wei.

Figura 20. Seleção de conta e envio de valores para o contrato para compra do boi gordo.

Figura 21. Execução da função “compraBoiGordo” e consulta variável contrato_finalizado.

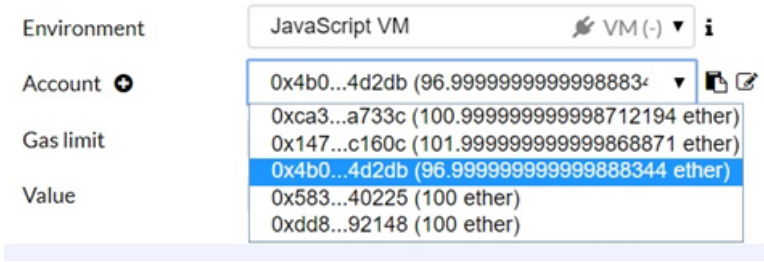

Figura 22. Novos saldos das contas após o contrato ter sido finalizado.

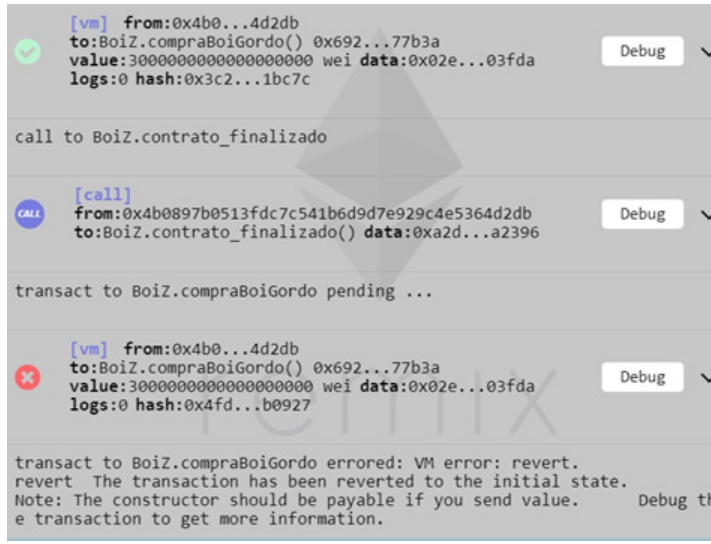

Figura 23. Mensagem de erro, devido à tentativa de compra após contrato ter sido finalizado.

Discussão

Neste comunicado técnico foi apresentado um modelo de *Smart Contract* de maneira ilustrativa e simplificada de mecanismo ideal de rastreamento na cadeira produtiva da carne. O objetivo principal é a compreensão das regras e das funções básicas de um *Smart Contract*, não apresentando todas as etapas nem o rigor de toda cadeia. Isso deverá ser realizado no futuro próximo por meio de projetos de P&D&I, com base no atual marco regulatório de C&T no país. É importante ressaltar que, em

uma situação mais realística, várias fazendas podem participar do *Smart Contract* e não apenas uma "fazenda de cria/recria/engorda", bem como não se faz necessário cadastrar um produtor e uma fazenda especificamente, uma vez que alguns confinamentos, por exemplo, podem ter animais de mais de um produtor. Outro aspecto importante é que o boi gordo é vendido pelo peso da carcaça e a negociação é feita em torno do valor da arroba. Já o preço do bezerro e do boi magro variam pelo peso vivo e em função do próprio animal. No modelo apresentado aqui, o valor do wei, menor unidade da criptomoeda ether, é flutuante e a própria rede o ajusta, contudo, numa rede fechada esse valor pode ser fixado. Por fim, vale enfatizar que o *Smart Contract* também pode ser desenhado de modo a conter um campo de validação com chave criptográfica atestando que o estabelecimento rural atende a *compliance* do Código Florestal e do Cadastro Ambiental Rural (CAR). Essa chave criptográfica poderia ser emitida por órgão governamental competente para cada estabelecimento rural já regularizado.

Conclusão

A implantação de mecanismos de rastreabilidade via *blockchain* na cadeia produtiva da carne pode proporcionar inúmeros benefícios e, por isso, merece atenção e incentivo de P&D&I. À medida que o *blockchain* funciona como um banco de dados descentralizado, em que é possível compartilhar informações com diferentes agentes envolvidos na cadeia de produção, a adoção dessa tecnologia pode conferir uma maior segurança às transações permitindo o rastreamento de toda a produção do alimento. Por meio das funções criptográficas, a tecnologia *blockchain* pode assegurar a fidelidade dos dados que circulam nas transações on-line, possibilitando realizar transferência de valores e diversos acordos comerciais. Além disso, a tecnologia *blockchain* permite a geração de massas de dados que podem no futuro próximo otimizar ainda mais as cadeias produtivas de alimento com base em análise de *Big Data* por meio da Inteligência Artificial.

Referências

ALDRIGHI, D. M. Uma avaliação das contribuições de Stiglitz à teoria dos mercados financeiros. **Revista de Economia Política**, v. 26, p. 137-57, 2006.

ALVES, E.; SOUZA, G.; SANTANA, C. Pobreza e sustentabilidade. **Revista de Política Agrícola**, v. 25, n. 4, p. 63-81, 2016.

BRASIL. Ministério da Agricultura, Pecuária e Abastecimento. **Sisbov**. Brasília, DF, 2017. Disponível em: <http://www.agricultura.gov.br/assuntos/sanidade-animal-e-vegetal/saude-animal/rastreabilidade-animal>. Acesso em: 28 out. 2018.

CARREFOUR launches Europe's first food blockchain. Disponível em: <http://www.carrefour.com/current-news/carrefour-launches-europes-first-food-blockchain>. Acesso em: 27 out. 2018.

DWORK, C.; NAOR, M. Pricing via processing or combating junk mail. 2001. In: ANNUAL INTERNATIONAL CRYPTOLOGY CONFERENCE, 12.,1992, Santa Barbara. **Proceedings...** Berlin; New York: Springer-Verlag, 1993. p. 139-147. (Lecture notes in computer

sciences, 740). Disponível em: <http://www.wisdom.weizmann.ac.il/~naor/PAPERS/pvp.pdf>. Acesso em: 27 jun. 2016.

EMBRAPA. **Visão 2014-2034**: o futuro do desenvolvimento tecnológico da agricultura brasileira. Brasília, DF, 2014. Disponível em: <http://ainfo.cnptia.embrapa.br/digital/bitstream/item/108955/1/Documento-Visao-versao-completa.pdf>. Acesso 20 out. 2018.

EMBRAPA. Departamento de Tecnologia da Informação. **Blockchain**. Brasília, DF, 2018. Nota técnica.

ETHEREUM Project. 2018. Disponível em: <https://www.ethereum.org/>. Acesso em: 20 out. 2018.

HABER, S.; STORNETTA, W. S. J. How to time-stamp a digital document. **Journal of Cryptology**, v. 3, n. 2, p. 99-111, Jan.1991. DOI: https://doi.org/10.1007/BF00196791.

MERKLE, R. C. One-way hash functions and DES. In: CONFERENCE ON THE THEORY AND APPLICATION OF CRYPTOLOGY, 1989, **Advances in Cryptology**: proceedings. New York: Springer-Verlag, 1990. p. 428-446. (Lecture notes in computer science, v. 435). Crypto '89.

MONEY. CNN Business. 2018. Disponível em: <https://money.cnn.com/2018/08/06/technology/mobile-voting-west-virginia-voatz/index.html >. Acesso em: 10 nov. 2018.

NAKAMOTO, S. **Bitcoin**: a peer-to-peer electronic cash system. 2009. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Acesso em: 28 out. 2018.

PENARD, W.; WERKHOVEN, T. van. On the secure hash algorithm family. **Cryptography in Context**, p. 1-18, 2008. Disponível em: <https://web.archive.org/web/20160330153520/http://www.staff.science.uu.nl/~werkh108/docs/study/Y5_07_08/infocry/project/Cryp08.pdf>. Acesso em: 28 out. 2018.

STIGLITZ, J. E. Markets failures, and development. **The American Economic Review**, v. 79, n. 2, p. 197-203, May 1989.

Exemplares desta edição podem ser adquiridos na:

Embrapa Informática Agropecuária
Av. Dr. André Tosello, 209 - Cidade Universitária
Campinas, SP, Brasil
CEP. 13083-886
Fone: (19) 3211-5700

www.embrapa.br
www.embrapa.br/fale-conosco/sac

1ª edição
Versão digital (2018)

Comitê Local de Publicações da Unidade Responsável

Presidente
Stanley R. de M. Oliveira

Secretária-Executiva
Carla Cristiane Osawa

Membros
Adriana Farah Gonzalez, Carla Geovana do Nascimento Macário, Flávia Bussaglia Fiorini, Jayme Barbedo, Kleber X. Sampaio de Souza, Luiz Antonio Falaguasta Barbosa, Maria Goretti Praxedes, Paula Regina K. Falcão, Ricardo Augusto Dante,Sônia Ternes

Suplentes
Michel Yamagishi e Goran Nesic

Supervisão editorial
Kleber X. Sampaio de Souza

Revisão de texto
Carla Geovana do Nascimento Macário

Normalização bibliográfica
Maria Goretti Gurgel Praxedes

Projeto gráfico da coleção
Carlos Eduardo Felice Barbeiro

Editoração eletrônica
Flávia Bussaglia Fiorini

Foto da capa
Pexels.com

CGPE 14948