Credenciais de Votação baseadas em BIP para Protocolos de Votação Resistentes à Coerção

Matheus O. L. de Sá, Roberto Araújo

[1]Faculdade de Computação – Universidade Federal do Pará (UFPA)
Belém – PA – Brasil

matheus.sa@icen.ufpa.br, rsa@ufpa.br

Abstract. *Coercion-resistant voting protocols rely on credentials to fight coercive attacks. For this, each voter receives a credential that she uses for voting and must generate a new random one when she is under coercion. However, credentials are usually large numbers that are difficult to hold. This makes hard the use of these protocols in practical voting scenarios. Aiming at overcoming this shortcoming, this work investigate the use of BIP-39 as a base to more practical credentials.*

Resumo. *Protocolos de votação resistentes à coerção dependem de credenciais para deterem ataques coercivos. Para isso, cada votante recebe uma credencial para votar e deve gerar credenciais falsas ao ser coagido. As credenciais, no entanto, são números grandes de difícil memorização. Isso dificulta a utilização desses protocolos em cenários práticos de votação. De forma a facilitar o emprego dessas credenciais, esse trabalho investiga a utilização do BIP-39 como base para credenciais mais práticas.*

1. Introdução

Coerção é um problema inerente a qualquer sistema de votação via Internet. Votantes podem utilizar seus dispositivos para votarem de qualquer lugar. No entanto, a inexistência de um ambiente controlado de votação potencializa a ação de opressores. Embora existam inúmeros ataques coercivos que podem ser realizados nesse ambiente, a resistência a tais ataques pode ser sumarizada pela noção introduzida por [Juels et al. 2010] (JCJ). Essa noção considera um protocolo como resistente à coerção se ele for capaz de deter ataques em que votantes são forçados a não votarem, ataques de simulação em que votantes podem revelar seus segredos e ataques em que votantes são forçados a escolherem opções aleatórias de voto.

Na literatura existem diversos protocolos que satisfazem a noção introduzida por JCJ, como a proposta de [Araújo et al. 2010]. Esses protocolos funcionam adequadamente em teoria e muitos deles tiveram sua segurança provada. No entanto, se considerados em cenários práticos, questões de usabilidade podem dificultar a resistência à coerção. Um dos principais problemas é relativo ao uso da credencial de votação.

Nesses protocolos, votantes recebem credenciais e as utilizam para votar. Ao serem coagidos, eles devem gerar credenciais falsas. Em geral, tais credenciais são números grandes, o que dificulta o seu armazenamento e a sua geração pelos votantes. Como apontado por [Neto et al. 2018], questões de usabilidade podem impactar na segurança do sistema de votação, dada a utilização incorreta das credenciais de votação.

A fim de facilitar o uso dessas credenciais em protocolos e sistemas de votação resistentes à coerção, esse trabalho avalia a possibilidade de uso do BIP-39 em credenciais de votação e propõe um algoritmo para esse fim.

Este trabalho está organizado da seguinte forma: a Seção 2 descreve as credenciais de votação empregadas em protocolos resistentes à coerção bem como os problemas inerentes a sua utilização prática; a Seção 3 descreve o BIP-39 e introduz um algoritmo baseado nesse mecanismo para geração de credenciais. As limitações desse algoritmo também são discutidas nessa seção. Finalmente, a Seção 4 conclui esse trabalho.

1.1. Trabalhos Relacionados

Soluções para problemas relativos ao uso de credenciais de votação já foram abordadas em outros trabalhos. [Neumann and Volkamer 2012] propõe o uso de credenciais de votação armazenadas em cartões inteligentes (*smart cards*), em conjunto de uma senha. Para que a credencial seja utilizada para votação, o votante deve utilizar seu cartão e inserir uma senha válida. Qualquer senha diferente resultará no envio de um voto com uma credencial falsa, que não será contabilizada durante a apuração. A proposta, no entanto, é passível de ataques de abstenção, já que um opressor poderia forçar o votante a entregar a ele o seu *smart card*, impedindo-o de votar.

[Neto et al. 2018] propõe que as credenciais sejam armazenadas em um servidor remoto. O acesso à credencial legítima é realizado por meio de senha. Caso o votante esteja sofrendo um ataque coercivo, ele deve informar uma senha falsa. Esta solução, no entanto, depende da confiança no sistema remoto. Caso contrário, o sistema pode facilmente revelar informações sobre as credenciais aos atacantes.

Por fim, [Clark and Hengartner 2008] propõe senhas de pânico para abstração das credenciais de votação, e [Clark and Hengartner 2011] utiliza essa ideia em um protocolo resistente à coerção. As senhas de pânico são determinadas pelo usuário. É definido um conjunto de senhas válidas, com tamanho especificado na eleição. Apenas uma dessas senhas será definida como a senha principal, associada a uma credencial válida. Votos emitidos com qualquer uma das demais senhas do conjunto serão descartados na apuração. Qualquer senha fora do conjunto resultará em erro, impedindo que equívocos de digitação invalidem votos. Entretanto, a proposta é passível de ataques de iteração caso o tamanho do conjunto seja pequeno, pois o conjunto de senhas válidas geradas não é expansível.

2. Credenciais em Protocolos Resistentes à Coerção

Protocolos de votação resistentes à coerção utilizam uma ideia diferente para identificar votos enviados por votantes. Nesses protocolos, os votantes não se autenticam antes de submeterem seus votos, como em protocolos convencionais, e.g. por meio de usuário e senha. Isso revelaria a identidade do votante e assim facilitaria ataques coercivos, como o ataque de abstenção forçada (o adversário força o votante a não votar). Ao invés, protocolos resistentes à coerção utilizam a ideia de credencial de votação. Ela identifica os votos que farão ou não parte do resultado final da votação.

Uma credencial é gerada por um conjunto de autoridades em cooperação, e é um segredo compartilhado onde cada autoridade conhece apenas uma parte dele. A credencial é entregue ao votante em segredo, livre da interferência de adversários, e apenas o

votante deve conhecer o inteiro conteúdo de sua credencial. Essa credencial é denominada legítima e cada votante recebe uma única e exclusiva credencial desse tipo. Esse processo ocorre em uma fase anterior a de votação. Votantes utilizam suas credenciais legítimas para votar e elas identificam (sem revelar qualquer informação sobre os votantes correspondentes) os votos que serão apurados.

Votantes podem submeter mais de um voto utilizando a mesma credencial, mas somente o último voto é considerado na apuração. Como não há autenticação prévia antes da submissão de votos, a identificação dos que fazem parte do resultado final é realizada pela utilização das credenciais legítimas. De forma a possibilitar resistência à coerção, votantes devem ser capazes de gerar credenciais falsas e devem utilizá-las ao sofrerem ataques coercivos. Para qualquer votante as credenciais falsas devem ser indistinguíveis das legítimas e votos submetidos com elas são removidos durante a apuração.

Uma credencial é composta por um conjunto de *bits* e sua composição pode diferir em protocolos resistentes à coerção. Em alguns protocolos, como o proposto por [Juels et al. 2010], a credencial é formada por um número aleatório grande (e.g. 160 bits). Outros, como o protocolo de [Araújo et al. 2010], utilizam credenciais baseadas em estruturas matemáticas, que também resultam em números aleatórios.

Independentemente da composição, credenciais são fundamentais para protocolos atingirem resistência à coerção, pelo menos em teoria. Todavia, quando esses protocolos são considerados em cenários práticos, as credenciais trazem problemas relativos à usabilidade, como apontado por [Neto et al. 2018]. Consequentemente, a utilização incorreta das credenciais levam votantes a não resistirem a ataques coercivos. Isso é potencializado pelos seguintes problemas.

O primeiro problema é relativo ao armazenamento e a posterior recuperação da credencial. Ao receberem suas credenciais, votantes devem mantê-las em segurança. Devido à credencial ser composta por um número aleatório grande, o seu armazenamento (e.g. por meio escrito ou digital) torna-se uma solução natural em um cenário prático. No entanto, um atacante poderia facilmente forçá-lo a revelar o valor armazenado. Dessa forma, idealmente, cada votante deveria memorizar os números relativos a sua credencial. Tal solução seria impraticável em um cenário real, tendo em vista que certamente a maioria dos votantes teria problemas para memorizá-los.

De forma a reagirem a ataques coercivos, votantes precisam gerar credenciais falsas e que pareçam aleatórias para um atacante. A geração e a utilização dessas credenciais, no entanto, resulta em outro problema prático. Em princípio, votantes poderiam gerar credenciais falsas com o auxílio de dispositivos (e.g. um smartphone) ou não. No primeiro caso, um votante teria que utilizar esse equipamento antes de ser coagido. Do contrário, um atacante poderia facilmente verificar que a credencial gerada é falsa.

Além disso, caso o atacante (por algum meio) tenha acesso ao dispositivo e consiga verificar se uma credencial foi gerada com ele (e.g. através dos registros existentes no dispositivo), o votante receberia uma punição. Consequentemente, isso poderia levar votantes a não utilizarem dispositivos e a revelarem suas credenciais legítimas. Assim, idealmente a geração de credenciais falsas deveria ocorrer mentalmente. No entanto, gerar números aleatórios grandes dessa forma não é uma tarefa simples.

3. Uma Proposta para o Emprego de Credenciais

Protocolos resistentes à coerção possibilitam votantes reagirem a ataques coercivos. Para isso, eles consideram que os votantes armazenam suas credenciais corretamente e estão sempre aptos a reagirem a esses ataques. A memorização da credencial legítima é certamente o meio mais eficaz para evitar que a credencial seja obtida por opressores. No entanto, o grande valor numérico inviabiliza sua memorização.

De forma a reduzir tal problema, o valor numérico poderia ser convertido em uma sequência de caracteres que possa ser mais facilmente memorizável. Isso resultaria em um mapeamento do valor numérico para a uma sequência de caracteres (ou palavras). A seguir é apresentado uma proposta para realizar tal mapeamento. Ela facilita a utilização de credenciais por votantes em protocolos resistentes à coerção. A ideia tem como base a proposta de melhoria de *Bitcoin*, BIP-39.

3.1. O BIP-39

O *Bitcoin Improvement Proposal* (Proposta de Melhoria de *Bitcoin*, em tradução livre; ou BIP) é um tipo de procedimento que visa aprimorar a utilização de Bitcoin [Bitcoin 2019]. Existem diversos BIPs diferentes, cada um atuando de forma bastante específica, e um dos já estabelecidos é o BIP-39. Ele descreve um protocolo para a geração de uma semente a partir de um conjunto de palavras mnemônicas [Palatinus et al. 2013].

Uma sequência mnemônica é um conjunto de palavras que podem ser lembradas por um usuário e que, neste caso, é utilizada para geração da semente. A semente é posteriormente empregada para geração de uma carteira determinística [Buterin 2013]. Essa carteira é necessária para armazenar as chaves privadas relacionadas às transações realizadas pelo seu proprietário. As carteiras digitais são geradas por outros BIPs (BIP-32 e BIP-44), mas estes não são necessários neste trabalho.

O BIP-39 consiste em duas etapas: a geração da sequência mnemônica do usuário (MS) e a conversão desta sequência para uma semente binária. Na primeira etapa é necessário gerar uma entropia (ENT), que obrigatoriamente deve ser um número pseudoaleatório múltiplo de 32 *bits*. O tamanho de ENT deve estar entre 128 e 256 *bits*. Quanto maior esse valor, mais palavras comporão a sequência mnemônica criada.

Após definido o valor ENT, é gerado um *checksum* ($CSum$). Para gerar esse valor, calcula-se o *hash* criptográfico de ENT utilizando o algoritmo SHA-256. O valor $CSum$ corresponde aos $ENT/32$ primeiros *bits* do resultado desse *hash*. Após gerado o valor $CSum$, ele é concatenado à entropia, resultando em $ENT||CSum$. O valor $ENT||CSum$ é organizado em grupos de 11 *bits*, resultando na quantidade de palavras da sequência mnemônica $MS = (ENT||CSum)/11$.

Cada elemento desse grupo será equivalente a uma palavra em um dicionário. O dicionário é composto por um total de 2^{11} palavras, associadas a um valor inteiro de 11 *bits*, ou seja, no intervalo de 0 à 2047. Por exemplo, se $ENT = 128$ *bits* e $CSum = 4$ *bits*, logo $ENT||CSum = 132$ *bits* e a sequência mnemônica terá o total de $MS = 12$ palavras selecionadas a partir do dicionário.

Após a definição do MS, é gerada a semente necessária para a criação de sua carteira digital. Para isso, todas as palavras da MS do participante devem ser concatenadas em uma frase mnemônica (MP), por ordem de geração. Como resultado obtém-se uma

única palavra com uma longa sequência de caracteres. Também pode ser gerado um *salt*, mas esse valor não é obrigatório.

Para finalizar, ambos os valores de MP e *salt* são utilizados como entrada para um algoritmo de derivação de chaves (nesse caso, é utilizado o *PBKDF*2 [Kaliski 2000]). O algoritmo retornará uma chave derivada dos valores inseridos, que será utilizada como semente para uma carteira digital determinística para Bitcoins.

3.2. Credenciais de Votação baseadas no BIP-39

Aparentemente, o BIP-39 poderia ser empregado diretamente no processo de geração de credenciais legítimas. As autoridades poderiam gerar um número aleatório e utilizá-lo como entropia ENT. A semente gerada ao final do processo seria a credencial do votante. Dessa forma, os votantes precisariam apenas memorizar sua sequência, como na proposta original. Durante a eleição, o votante utilizaria a sequência ao votar e esta seria convertida em sua credencial numérica em seu dispositivo computacional, antes do envio de seu voto.

No entanto, para garantir a segurança na geração dessas credenciais, as autoridades precisariam proceder de forma distribuída ao utilizar o BIP-39. Durante esse processo elas não poderiam obter nenhum *bit* da entropia bem como do conjunto de palavras mnemônicas entregue ao votante. Para isso, seria necessária uma função criptográfica de *hash* em que os valores de entrada e saída fossem desconhecidos pelas autoridades. Infelizmente não existe na literatura tal função e isso inviabiliza o uso do BIP-39 para gerar credenciais dessa forma.

A utilização da semente como credenciais legítimas é inviável. Todavia, parte do BIP-39 ainda pode ser útil para geração dessas credenciais. O BIP-39 pode ser dividido em duas partes. Na primeira parte, ele recebe uma entropia inicial e a utiliza no processo de definição das palavras mnemônicas. Na segunda parte, essas palavras são utilizadas para gerar a semente final via *PBKDF*2. A partir dessa divisão, o seguinte algoritmo para geração de credenciais é apresentado.

Seja R um conjunto de autoridades responsáveis pela geração (em cooperação) de uma credencial legítima e numérica σ para cada votante. O conjunto R gera a credencial σ de acordo com o protocolo de votação e a entrega ao votante em sigilo. Em um ambiente real, essa iteração entre R e o votante ocorreria em um computador cliente acessando um conjunto de servidores. Supõe-se que o votante tem um par de chaves assimétricas, necessário apenas para esse processo, e ocorre após o eleitor se autenticar (e.g. apresentando sua carteira de identidade) para R. Tendo esse par de chaves sido estabelecido entre ambos, R retorna a credencial σ criptografada com a chave pública do votante.

Ainda nesse ambiente sigiloso, o votante informa sua chave privada ao cliente. Após isso, o computador descriptografa σ e executa a primeira parte do BIP-39 utilizando o valor de σ como entropia. Como resultado, o computador retorna uma sequência de palavras mnemônicas ao votante, que devem ser memorizadas.

O Algoritmo 1 descreve o processo de geração de credenciais baseada no BIP-39. Note que o protocolo determina um mapeamento bijetor entre a credencial numérica σ e a sequência de palavras mnemônicas. A partir da credencial numérica σ pode-se obter as palavras mnemônicas que representam a credencial. Reversalmente, a partir das palavras

mnemônicas, pode-se obter o valor σ. O BIP-39 calcula o valor *hash* da entropia para determinar os *bits* que serão concatenados com a entropia mais adiante. O resultado disso é dividido em grupos de 11 *bits*, que indicarão as palavras.

De posse das palavras, pode-se retornar ao valor da entropia realizando processo inverso. Ou seja, convertendo as palavras em valores numéricos de acordo com o dicionário; concatenando os grupos numéricos de 11 *bits* em um único valor numérico grande; e destacando os primeiros ENT *bits* iniciais do número gerado. Esses primeiros valores serão relativos à credencial de votação, e seu *hash* deve iniciar com os mesmos valores que os $ENT/32$ *bits* finais do número grande gerado anteriormente. Essa conversão é importante, pois, ao enviar um voto devem ser informadas as palavras mnemônicas do votante, que serão convertidas no valor numérico referente à credencial. Por outro lado, ao receber sua credencial σ, o votante precisa convertê-la em palavras mnemônicas.

Algoritmo 1 Geração de Credencial baseada no BIP-39

Entrada: Uma credencial legítima σ em formato númerico, cujo valor é múltiplo de 32 *bits*. Uma lista de palavras mnemônicas
Saída: Um conjunto de palavras mnemônicas que representam uma credencial legítima
Algoritmo:

1. $ENT \leftarrow \sigma$
2. Calcular o *Hash* da entropia: $H \leftarrow Hash(ENT)$
3. Calcular o tamanho do *Checksum* em *bits*: $m \leftarrow ENT/32$
4. Selecionar os m primeiros *bits* de H
5. Concatenar os m *bits* selecionados ao final da entropia, gerando a sequência $ENTC \leftarrow ENT \,||\, mBits$
6. Dividir o valor $ENTC$ em grupos de 11 bits
7. Converter cada 11 *bits* para um número inteiro
8. Utilizar cada número inteiro para selecionar uma palavra na lista de palavras mnemônicas correspondente, criando uma sequência de palavras MS
9. Retorna MS

A Figura 1 ilustra o algoritmo proposto. Os participantes envolvidos no algoritmo são um conjunto de autoridades de registro R e um votante. As autoridades geram uma credencial σ, que será usada como entropia ENT (passo 1). A credencial pode possuir uma estrutura matemática como a definida no protocolo de [Araújo et al. 2010]). Ela passa então por uma função criptográfica de *Hash* (passo 2), onde é destacado como *checksum* $CSum$ apenas os seus primeiros bits (passo 4), com o tamanho definido no passo 3). A credencial σ é concatenada ao $CSum$ (passo 5).

O valor resultante da concatenação é dividido em grupos de 11 bits, na etapa de clusterização (passo 6). Cada valor passa por um dicionário com 2^{11} palavras, cada uma associada a um valor, onde será convertido em uma palavra (passo 7). O conjunto de palavras geradas a partir dos passos anteriores é a sequência de palavras mnemônicas (passo 8), que é recebida pelo usuário (passo 9).

3.3. Resultados e Discussões

A memorização de palavras ao invés de números grandes certamente facilita o armazenamento e a posterior recuperação de credenciais. A fim de realizar isso, é necessário um

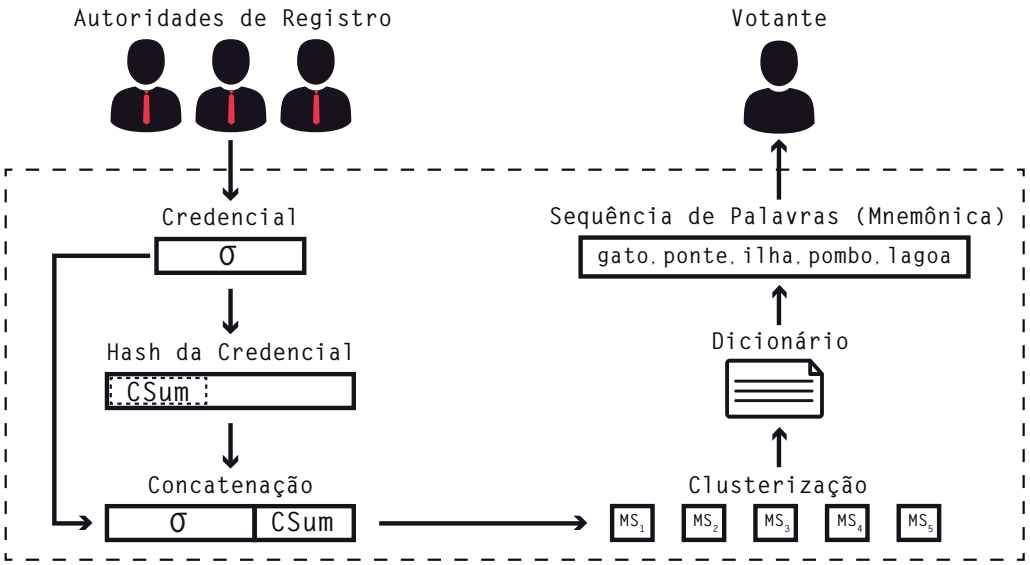

Figura 1. Algoritmo para geração de credencial baseada no BIP-39. O processo inicia com a geração das credenciais pelas autoridades e finaliza com a sequência de palavas representando a credencial de votação.

mapeamento entre a credencial numérica gerada e a sequência de palavras a serem memorizadas. O protocolo apresentado possibilita tal mapeamento. A seguir são discutidos alguns aspectos que podem limitar o uso de tal solução e que são pontos de pesquisa em andamento.

Assim como o BIP-39, o protocolo apresentado requer uma entropia de 128 *bits* e um *checksum* de 4 *bits* para gerar 12 palavas mnemônicas. Com uma entropia de 160 *bits* e 5 *bits* de *checksum* são geradas 15 palavras. Essas palavras provêm de uma lista fixa e predefinida de 2048 palavras do BIP-39. Elas são selecionadas aleatoriamente a partir da entropia e do *checksum*. O número alto de palavras é um dos fatores que garantem a segurança no BIP-39. Esse número, todavia, pode ser alto se for considerada a quantidade média de palavras memorizadas por votantes. Assim, testes são necessários para determinar a usabilidade do protocolo. Por outro lado, é possível adaptar o protocolo para gerar um número menor de palavras (e.g. definindo-se uma entropia menor), mas é necessário verificar se tal mudança ainda garantiria a segurança. Além disso, seria necessário definir uma nova lista de palavras.

A ideia apresentada utiliza como entropia a credencial gerada por um grupo de autoridades. Como descrito na Seção 2, a credencial é um número aleatório grande. Em protocolos que utilizam credenciais matematicamente estruturadas, a credencial é um número aleatório dentro de um grupo numérico seguro (e.g. o grupo dos inteiros onde o problema de decisão de Diffie-Hellman é difícil). A entropia definida no BIP-39 possui um tamanho fixo. Todavia, como uma credencial é um número grande, é necessário garantir que o tamanho em *bits* do número gerado tenha o mesmo tamanho requerido pela entropia. Uma alternativa seria determinar um tamanho mínimo e máximo para a entropia, mas isso requer uma análise cuidadosa para evitar comprometimento da segurança.

Em protocolos com resistência à coerção, uma credencial falsa é um número aleatório qualquer e diferente da credencial legítima. Seguindo a ideia apresentada, uma

credencial falsa seria qualquer sequência de palavras dentro da lista de palavras disponíveis. No BIP-39 essa lista tem 2048 palavras. Como essas palavras mapeiam um número, ao selecionar palavras aleatórias a partir da lista, o votante está gerando um valor aleatório equivalente a uma credencial falsa. Todavia, para evitar coerção, o votante não poderia escolher nenhuma palavra diferente da lista. Caso contrário, um atacante poderia facilmente verificar que a palavra não consta na lista. Votantes precisariam lembrarem-se de algumas palavras da lista para utilizarem como credencial falsa. Uma alternativa, seria misturar partes das palavras verdadeiras com as falsas.

A geração de credenciais falsas em protocolos com credenciais matematicamente estruturadas teria um problema adicional. Ao selecionar palavras utilizando a lista de palavras disponíveis, a sequência resultante pode não corresponder a um número pertencente ao grupo numérico. Um atacante poderia facilmente verificar tal pertinência e punir sua vítima.

Antes de gerar a semente a ser utilizada na carteira, o BIP-39 emprega o *PBKDF*2. Esse mecanismo objetiva dificultar ataques de força bruta. A ideia apresentada, no entanto, tem como base a primeira parte do BIP-39 e assim não utiliza o *PBKDF*2. De forma a estabelecer a relação entre uma credencial numérica e uma sequência de palavras mnemônicas, é necessário que o mapeamento seja bijetor. Como resultado, pode-se obter o valor numérico a partir da sequência e vice-versa.

Se o *PBKDF*2 fosse utilizado como no BIP-39, não seria possível retornar as palavras a partir da saída do *PBKDF*2. Por outro lado, a ideia apresentada facilita ataques de força bruta. No entanto, para realizar um ataque desse tipo, ainda seria necessário enviar credenciais criptografadas ao sistema, após gerar credenciais. Em princípio, esse ataque poderia ser minimizado requerendo a resolução de um problema (tal como ocorre na prova de trabalho em *blockchain* onde um minerador precisa resolver um problema antes de adicionar um bloco na cadeia) antes de aceitar um voto e sua correspondente credencial.

4. Considerações Finais

Credenciais de votação são um dos principais instrumentos disponíveis para deter ataques coercivos. Embora credenciais possibilitam resistência à coerção, em cenários práticos a ação dos votantes é fundamental para deter esses ataques. Tal ação depende da facilidade de uso da credencial. Como apresentado, as credenciais são representadas por um número aleatório, o que dificulta sua memorização.

Neste contexto, este trabalho propôs um algoritmo, baseado no uso do BIP-39, para facilitar o uso de credenciais de votação. A ideia mapeia a credencial para uma sequência de palavras e assim facilita o seu emprego. No entanto, apesar de inicialmente facilitar o uso de credenciais enquanto garante sua segurança, o trabalho ainda carece de investigações e estas são objetivos de trabalhos futuros. É necessário avaliar mais detalhadamente a segurança da solução bem como realizar testes de usabilidade após sua implementação.

Agradecimentos

Agradecemos à CAPES pelo apoio financeiro, o qual foi imprescindível para a concretização desse trabalho.

Referências

Araújo, R., Rajeb, N. B., Robbana, R., Traoré, J., and Yousfi, S. (2010). Towards practical and secure coercion-resistant electronic elections. In Heng, S., Wright, R. N., and Goi, B., editors, *Cryptology and Network Security - 9th International Conference, CANS 2010, Kuala Lumpur, Malaysia, December 12-14, 2010. Proceedings*, volume 6467 of *Lecture Notes in Computer Science*, pages 278–297. Springer.

Bitcoin (2019). Readme.mediawiki. `https://github.com/bitcoin/bips`. GitHub repository.

Buterin, V. (2013). Deterministic wallets, their advantages and their understated flaws. *Bitcoin Magazine*.

Clark, J. and Hengartner, U. (2008). Panic passwords: Authenticating under duress. *HotSec*, 8:8.

Clark, J. and Hengartner, U. (2011). Selections: Internet voting with over-the-shoulder coercion-resistance. In *International Conference on Financial Cryptography and Data Security*, pages 47–61. Springer.

Juels, A., Catalano, D., and Jakobsson, M. (2010). Coercion-resistant electronic elections. In *Towards Trustworthy Elections*, pages 37–63. Springer.

Kaliski, B. (2000). Rfc 2898; pkcs# 5: Password-based cryptography specification version 2.0.

Neto, A. S., Leite, M., Araújo, R., Mota, M. P., Neto, N. C. S., and Traoré, J. (2018). Usability considerations for coercion-resistant election systems. In *Proceedings of the 17th Brazilian Symposium on Human Factors in Computing Systems*, page 40. ACM.

Neumann, S. and Volkamer, M. (2012). Civitas and the real world: problems and solutions from a practical point of view. In *2012 Seventh International Conference on Availability, Reliability and Security*, pages 180–185. IEEE.

Palatinus, M., Rusnak, P., Voisine, A., and Bowe, S. (2013). Mnemonic code for generating deterministic keys. `https://github.com/bitcoin/bips/blob/master/bip-0039.mediawiki`. GitHub repository.