A Implementação de um Protocolo Criptográfico para Geração Distribuída de Credenciais no Sistema CIVIS

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. *The Internet election system CIVIS is based on a cryptographic voting protocol that enables resistance to coercive attacks. For this, it uses the idea of credentials. A credential consists of a set of bits that must be generated by a set of electoral authorities and delivered to each voter in secrecy. Otherwise, the protocol does not guarantee resistance to coercive attacks. In the current version of the CIVIS system, however, each credential is generated by single electoral authority. As a consequence, the system relies on this authority to be trusted to ensure secure credential generation. In this context, this work presents an implementation of a protocol for the distributed generation of credentials in order to make credential generation more secure in the CIVIS system.*

Resumo. *O sistema para eleições via Internet CIVIS é baseado em um protocolo criptográfico que possibilita resistência à ataques coercivos, e para tal utiliza a ideia de credenciais. Uma credencial é formada por um conjunto de bits que deve ser gerado por um conjunto de autoridades eleitorais e entregue a cada votante em sigilo. Do contrário, o protocolo não garante resistência à ataques coercivos. Na versão atual do sistema CIVIS, entretanto, cada credencial é gerada por única autoridade eleitoral. Como consequência, o sistema depende que essa autoridade seja confiável para garantir a geração segura da credencial. Nesse contexto, este trabalho apresenta uma implementação de um protocolo para a geração distribuída de credencias a fim de tornar a geração de credenciais mais segura no sistema.*

1. Introdução

A noção de resistência à coerção introduzida por Juels, Catalano e Jakobsson [Juels et al. 2010] (denominado aqui como JCJ) considera um cenário de votação realístico em que um atacante pode, por exemplo, forçar um votante a revelar sua chave secreta de votação. A partir dessa noção, JCJ apresentou o primeiro protocolo criptográfico de votação para mitigar tais ataques. Tal protocolo é hoje utilizado como *framework* no projeto de protocolos resistentes à coerção.

De forma a obter resistência à coerção, o protocolo de JCJ utiliza a ideia de credencial anônima. Cada votante recebe secretamente (e.g. em um local de registro livre de adversários) uma única e exclusiva credencial (e.g. um número aleatório) para votar. Essa credencial autoriza um voto válido na apuração. Idealmente, a credencial recebida pelo votante deve ser gerada em cooperação por um conjunto de autoridades e garantindo que ela seja conhecida apenas pelo votante.

Baseado nas ideias de JCJ, Araujo et al. [Araújo et al. 2010] (denominado aqui como ABRTY) introduziram um protocolo de votação com uma fase de apuração mais eficiente (linear *versus* quadrática). Para alcançar tal eficiência, a solução utiliza uma credencial que possui uma estrutura matemática ao invés de um número aleatório. Assim como o protocolo de JCJ, essa credencial também deve ser criada de forma distribuída por um conjunto de autoridades.

Recentemente, uma implementação do protocolo de ABRTY foi apresentada por [Araújo et al. 2018]. Tal implementação possibilita a realização de votações onde existam meios de se resistir a ataques coercivos. A versão apresentada, no entanto, utiliza apenas uma autoridade para gerar as credenciais de votação. Dessa forma, uma autoridade maliciosa poderia comprometer a propriedade de resistência à coerção do sistema. Como a autoridade conhece a credencial de votação, ela poderia personificar o votante, ou mesmo informá-la a um atacante. É necessário garantir que o sistema resista contra o problema.

A geração de credenciais do CIVIS pode ser aperfeiçoada utilizando técnicas de criptografia limiar, permitindo que a geração seja realizada de forma distribuída em um conjunto de autoridades de registro. Este conjunto deve colaborar para a criação de credenciais, minimizando a vulnerabilidade sobre autoridades corruptas. Este trabalho apresenta adaptações ao protocolo de geração distribuída de assinaturas digitais proposto por [Wang et al. 2005] (denominado aqui como WZF) para possibilitar sua utilização para geração distribuída de credenciais. Também é apresentada sua implementação.

Este trabalho está estruturado da seguinte maneira: a Seção 2 apresenta alguns trabalhos presentes na literatura e que possuem abordagens semelhantes a este trabalho; na Seção 3 são apresentadas as bases que fundamentaram o tema abordado; a Seção 4 apresenta a modificação do protocolo e a implementação realizada, detalhando sua estrutura e arquitetura; e por fim, a Seção 5 apresenta as conclusões e observações finais sobre o trabalho, assim como apresenta os possíveis trabalhos futuros.

2. Trabalhos Relacionados

Aprimoramentos ao sistema de votações via *Internet* CIVIS também foram o foco de outros trabalhos. O trabalho de [Silva Neto and Araújo 2017] propõe a implementação e integração do criptossistema El Gamal limiar utilizado pelos CIVIS. Entretanto, o criptossistema implementado por esse trabalho não pôde ser utilizada diretamente para gerar credenciais de forma distribuída. A partir dos resultados de sua implementação, o trabalho auxiliou também como base teórica para a implementação do protocolo de geração distribuída de credenciais, proposto neste trabalho.

O trabalho de [Silva Neto et al. 2018] também é baseado no funcionamento do CIVIS, onde é analisada a usabilidade do sistema. São propostas adaptações no procedimento de envio de voto por cada participante. Essas adaptações lidam diretamente com o uso das credenciais, mas não tratam de alterar sua estrutura matemática ou a sua geração, não causando divergência com essa implementação.

O protocolo de WZF também foi usado como mecanismo em outro trabalho, também aplicado a eleições via *Internet*, como fundamentalção de sua implementação. O trabalho de de [Souheib et al. 2012] utiliza o protocolo como forma de minimizar a complexidade do sistema implementado, que é $O(n^2)$. É proposto um esquema distribuído de geração de marcas d'água para as credenciais como forma de verificação.

3. Fundamentação Teórica

Esta seção apresenta as principais tecnologias empregadas nesse trabalho.

3.1. O Sistema de Votação via *Internet* CIVIS

O CIVIS [Araújo et al. 2018] é um sistema de votação via *Internet* desenvolvido em *Python* [Van Rossum and Drake 2003], utilizando o *framework* de desenvimento *web Django* [Holovaty and Kaplan-Moss 2009]. O sistema utiliza o protocolo criptográfico proposto por [Araújo et al. 2010], que estabelece uma votação via *Internet* segura e resistente à coerção, utilizando credenciais geradas com base em primitivas criptográficas.

O sistema é organizado em eleições, onde cada uma possui um conjunto de fases, e são organizadas por autoridades com responsabilidades específicas. Três autoridades atuam numa eleição; a autoridade de eleição, responsável por configurar a eleição num geral; a autoridade de registro, responsável por gerar credenciais para os participantes; e a autoridade de apuração, responsável pela apuração dos votos em uma eleição.

As credenciais são um mecanismo utilizado pelo sistema para possibilitar a resistência à coerção de votantes. Elas são geradas unicamente pela autoridade de registro, e possuem uma estrutura matemática composta pelos valores $(A = (g_1 g_3^x)^{\frac{1}{y+r}}, r, x)$, onde $g_1, g_3 \in \mathbb{Z}_p^*$ são dois geradores aleatórios, $r, x \in \mathbb{Z}_q^*$ são dois números aleatórios, e y é a chave privada da autoridade de registro [Araújo et al. 2018].

Toda credencial é única, e associada a apenas um votante, impossibilitando que mais de um votante utiliza uma mesma credencial. Cada uma será utilizada para enviar votos em uma eleição. O sistema também permite que um voto seja enviado utilizando uma credencial diferente da associada ao votante, mas este voto não será considerado durante a apuração. O sistema não indica que o voto enviado é inválido, impedindo que qualquer usuário além do votante possa diferenciar um envio de voto que utilize a sua credencial original, ou uma forjada.

Cada eleição no sistema possui apenas uma autoridade de registro, portanto a geração de credenciais é centralizada por eleição. Supõe-se que a autoridade de registro seja confiável, mas caso essa autoridade seja maliciosa, ela pode arruinar a segurança do sistema, possibilitando um cenário de coerção. A proposta deste trabalho é a implementação de um protocolo de geração de credenciais realizada de forma distribuída, com mais de uma autoridade de registro.

3.2. O Protocolo de Assinatura Digital de Wang, Zheng e Feng

O protocolo de Wang, Zheng e Feng é uma versão limiar do esquema de assinaturas digitais curtas de [Boneh and Boyen 2004]. A versão de Boneh e Boyen fundamenta-se em um grupo bilinear seguro (G_1, G_2). Neste protocolo, para gerar o par de chaves pública de assinatura, calcula-se um gerador $g_2 \in G_2$ que é também utilizado para se determinar g_1. Então calcula-se $u = g_2^x \in G_2$ e $v = g_2^y \in G_2$, onde $x, y \in_R \mathbb{Z}_p^*$, e $z = e(g_1, g_2) \in G_T$. A chave pública é formada por (g_1, g_2, u, v, z) e a chave privada correpondente é x, y. Para assinar uma mensagem $m \in \mathbb{Z}_{\text{ı}}^*$, seleciona-se $r \in_R \mathbb{Z}_p^*$ e calcula-se $\sigma = g_1^{1/(x+m+yr)} \in G_1$. A assinatura é formada por (σ, r). A assinatura é verificada calculando-se: $e(\sigma, u \cdot g_2^m \cdot v^r) = z$.

A versão do protocolo de WZF segue a mesma ideia de Boneh e Boyen. No entanto, o par de chaves de assinatura e a assinatuta digital são gerados de forma distribuída por um conjunto de partes. O esquema fundamenta-se em técnicas de compartilhamento de segredos já conhecidas, como o protocolo de compartilhamento de segredos de [Pedersen 1991] (PedVSS) e o esquema de geração distribuída de chaves de [Gennaro et al. 1999]. Tais protocolos são utilizados para dividir um segredo S em n partes $S_1, \ldots, S_n$ de forma que seja necessário um mínimo de t de partes S_i para se recuperar S, como proposto por Shamir [Shamir 1979]. Aqui utiliza-se o termo fração de segredo como referência às partes S_i do segredo a ser compartilhado.

O esquema é dividido em três etapas: a geração da chaves de comprometimento; a geração do par de chaves de assinatura; a geração da assinatura digital limiar. Além dessas três etapas, existe a etapa de verificação da assinatura que é similar a do esquema original. Na primeira etapa ocorre a geração das chaves de comprometimento de forma distribuída utilizando o protocolo de [Pedersen 1991]. Compromentimentos são utilizados em várias partes do esquema que as partes possam verificar segredos recebidos. Após essa etapa ocorre a geração distribuida das chaves de assinatura. Aqui é utilizado o protocolo de [Gennaro et al. 1999], denominado por WZF como ExpVSS, e obtém-se como saída a chave pública de assinatura (g_1, g_2, h_2, u, v, z) e a chave privada x, y correspondente.

Após ambas as etapas, as partes estão aptas a gerar assinaturas de forma distribuída. Para isso, elas primeiramente executam o RndVSS para gerar o valor aleatório r para a assinatura. O RndVSS corresponde à execução do algoritmo PedVSS por cada parte. Em seguinda, as partes executam o algoritmo ExpVSS para gerar o valor aleatório a. Seja m uma mensagem, cada parte então calcula $b_i = x_i + m + ry_i (mod\ p)$ e $c_i = a_i \cdot b_i (mod\ p)$ utilizando as partes a_i, x_i, y_i de seus segredos e, utiliza o PedVSS para compartilhar c_i com os demais participantes. O valor secreto $c = a \cdot b (mod\ p)$ é obtido através da combinação linear dos valores c_i. Na sequência, cada parte calcula localmente $\sigma = (F_0^{(a)})^{c^{-1}} = (g_2^a)^{c^{-1}} = g_1^{\frac{1}{(x+m+ry)}}$, formando a assinatura (σ, r).

4. A Implementação do Protocolo Criptográfico para Geração Distribuída de Credenciais

4.1. A Adaptação do Protocolo de WZF

A credencial utilizada no protocolo implementado pelo CIVIS compartilha uma estrutura matemática similar à assinatura digital gerada no esquema de Boneh e Boyen. Por exemplo, elas são baseados no mesmo problema $q - SDH$ (q-Diffie-Hellman forte) e os membros A e r da credencial são semelhantes aos membros σ e r da assinatura. Tais similaridades possibilitam que o esquema de WZF possa ser utilizado para gerar credenciais. No entanto, tal utilização requer adaptações.

O esquema de WZF considera um grupo billinear enquanto que o CIVIS considera o grupo dos inteiros. Além disso, embora possuam similaridades, uma assinatura digital gerada no protocolo WZF tem a estrutura $(\sigma = g_1^{\frac{1}{(x+m+ry)}}, r)$. Diferentemente, uma credencial no CIVIS tem a estrutura $(A = (g_1 g_3^x)^{\frac{1}{y+r}}, r, x)$. Assim, para gerar as credenciais de acordo com o protocolo de WZF, foram realizadas três adaptações.

A primeira modifica ao grupo utilizado, considerando-se o grupo cíclico onde o problema de decisão de Diffie-Hellman é difícil, ao invés de um grupo bilinear. Esse

é utilizado no CIVIS e é estabelecido selecionando dois primos aleatórios p, q tal que $p = 2q+1$. A segunda altera o expoente da credencial, mantendo seu valor como $(y+r)^{-1}$, requerido pela credencial, ao invés de $(x + m + ry)^{-1}$, como usado por WZF. Assim, na adaptação, não é utilizado a chave secreta x e o texto aberto m. Com esta modificação, a variável b_i do protocolo WZF também deverá ser alterada, sendo $b_i = y_i + r_i \mod q$.

A terceira adaptação modifica os geradores de grupo da credential. O protocolo de WZF utiliza apenas um gerador na assinatura, mas o modelo de credencial do CIVIS utiliza dois geradores, $g_1, g_3 \in \mathbb{Z}_p^*$. Além disso, o gerador g_3 é elevado a um número aleatório $x \in \mathbb{Z}_q^*$. Para que $F_0^{(a)} = (g_1 g_3^x)^a \mod p$, de acordo com o protocolo de WZF, é necessário que $g_A = (g_1 g_3^x) \mod p$, portanto é necessário que ambos os geradores sejam calculados antes de gerar o valor aleatório a. A Figura 1 apresenta como o protocolo deverá funcionar para gerar credenciais, após adaptado.

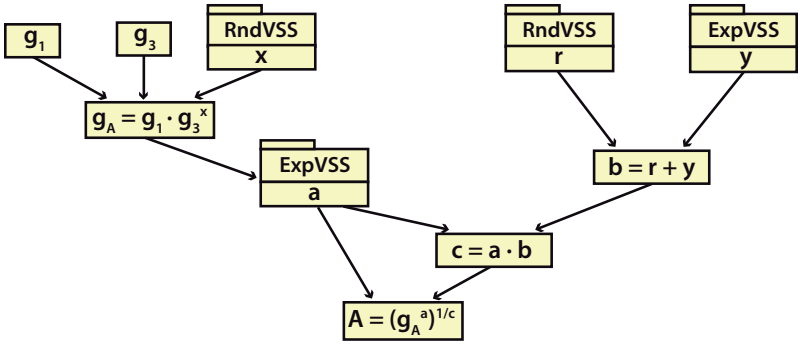

Figura 1. Visão geral das variáveis utilizadas na adaptação do protocolo de WZF

4.2. A Arquitetura da Implementação

O protocolo de geração distribuída de credenciais foi implementado na forma de módulos. Cada um de seus módulos representa um dos protocolos limiares utilizados, sendo organizados como: PedVSS; RndVSS; ExpVSS; e WZF. A utilização de módulos também auxiliou no reaproveitamento de mecanismos criptográficos já existentes no sistema CIVIS, mas também necessários para os módulos implementados. Todos os módulos descritos são diretamente relacionados aos seus anteriores.

Um módulo já existente no CIVIS, mas reaproveitado nesta implementação, é o relativo ao protocolo de Lagrange, responsável por realizar interpolações polinomiais. Ele é necessário para a implementação do módulo PedVSS, uma vez que este compartilhamento se baseia em interpolações de frações para recuperar o segredo [Pedersen 1991].

A Figura 2 apresenta as interações entre os protocolos implementados e reaproveitados, apresentando também o que cada protocolo gera como resultado. Considerando os protocolos implementados, o WZF utiliza os protocolos seguintes para a geração de termos necessários para gerar uma credencial; o ExpVSS gera um polinômio público a partir do resultado do RndVSS; o RndVSS compartilha um segredo desconhecido por todos a partir da soma das frações de segredo, geradas no PedVSS; e o PedVSS utiliza o protocolo de Lagrange para compartilhar um segredo conhecido por um participante.

A Figura 3 apresenta as classes utilizadas no módulo de *Lagrange*. A classe *PolinomioLagrange* apresenta um polinômio calculado. Este é composto por um *array* de

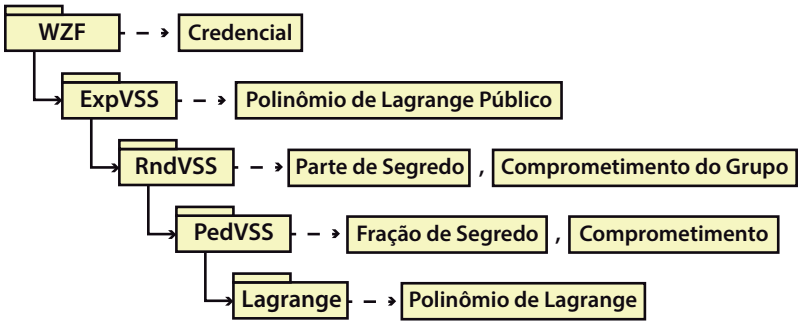

Figura 2. Visão geral do sistema representada em um diagrama de classes

números inteiros longos, onde cada um destes é um dos termos do polinômio, variando de 0 até t. Esta classe possui dois métodos: o gerador *gerar_polinomio()*, que seleciona um polinômio com fator aleatório; e *gerar_coordenada()*, que calcula e retorna um objeto da classe *Coordenada*. A classe *Coordenada* abrange as coordenadas de um polinômio de *Lagrange*, onde x é um identificador e y uma informação secreta.

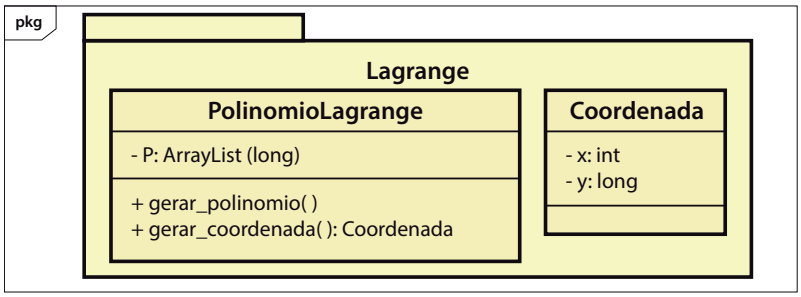

Figura 3. Diagrama do módulo Lagrange [Silva Neto and Araújo 2017]

A Figura 4 apresenta o diagrama de classes do módulo PedVSS. A classe *Segredo* representa o segredo a ser compartilhado, com seu valor representado pelo atributo S, e possui como método $compartilhar\_segredo()$, que resulta em um objeto do tipo *FracaoSegredo*. A classe *FracaoSegredo* corresponde às frações geradas a partir de um compartilhamento, composta pelos atributos id e f, representando o valor da fração o identificador do usuário desta fração.

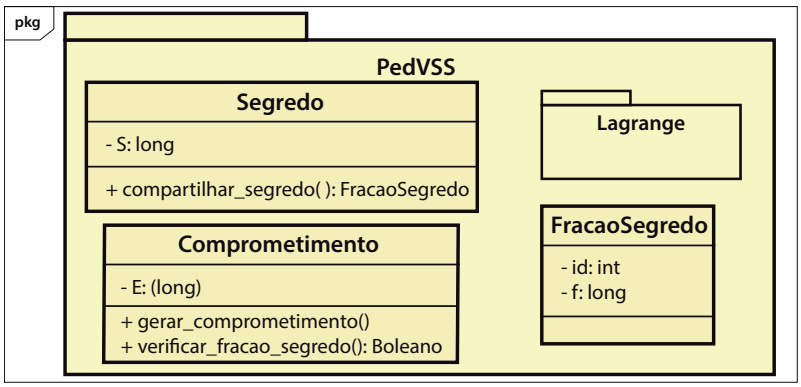

Figura 4. Diagrama de classes do módulo PedVSS

A classe *Comprometimento* representa valores utilizados para verificação das frações de segredo, com atributo E correspondente a uma lista destes valores. Possui como métodos o gerador $gerar\_comprometimento()$, que realiza os comprometimentos necessário para o compartilhamento de segredo realizado pelo método; e $verificar\_fracao\_segredo()$, que aplica a verificação de uma fração de segredo.

A Figura 5 apresenta o diagrama de classes do módulo RndVSS. Este módulo possui duas novas classes, *ComprometimentoGrupo* e *ParteSegredo*, ambas herdadas das classes *Comprometimento* e *FraçãoSegredo* do módulo anterior, respectivamente. Não existe uma classe referente ao segredo compartilhado nesse módulo, já que o protocolo garante que seu valor seja desconhecido para todos os participantes.

A classe *ParteSegredo* representa o somatório de frações de segredo recebidas por um determinado usuário. Possui como método $somatorio\_segredo()$, que é seu construtor. A classe *ComprometimentoGrupo* representa o produtório de comprometimento dos compartilhamentos necessários para gerar cada fração de segredo recebida por um determinado usuário. Possui como método contrutor $produtorio\_comprometimento()$, onde os comprometimentos de [Pedersen 1991] são reunidos.

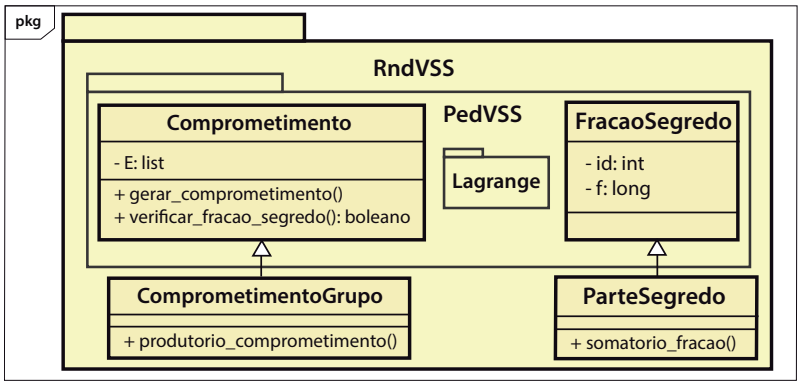

Figura 5. Diagrama de classes do módulo RndVSS

A Figura 6 apresenta o diagrama de classe do módulo ExpVSS. Sua nova classe criada, *PolinomioLagrangePublico* se relaciona diretamente como uma herança da classe *PolinomioPublico*, detalhada anteriormente no módulo de *Lagrange*. Esta nova classe representa o polinômio $F_k = g^{f_k}$, utilizado para verificar o objeto *ParteSegredo* e permitindo a recuperação de uma chave pública para o segredo gerado no módulo anterior. Como métodos possui $gerar\_polinomio\_publico()$, seu construtor; e $verificar\_polinomio()$, responsável pela verificação do polinômio de Lagrange Público.

A Figura 7 apresenta o diagrama de classes do módulo WZF. Nele estão contidos todos os módulos anteriores. A classe *Credencial* possui os parâmetros A, representando o valor de $A = (g_A)^{\frac{1}{y+r}}$; e x, r, ambos segredos aleatórios utilizados para calcular A. Como métodos possui seu construtor $gerar\_credencial()$; o método $gerar\_ci()$, que gera a variável $c_i = a_i \cdot b_i \mod q$; e $gerar\_A()$, que gera o valor do atributo A. A classe *GeradorCredencial* possui como atributo g, equivalente a $g_A = g_1 \cdot g_3^x$, e como método seu construtor $gerar\_gerador\_credencial$.

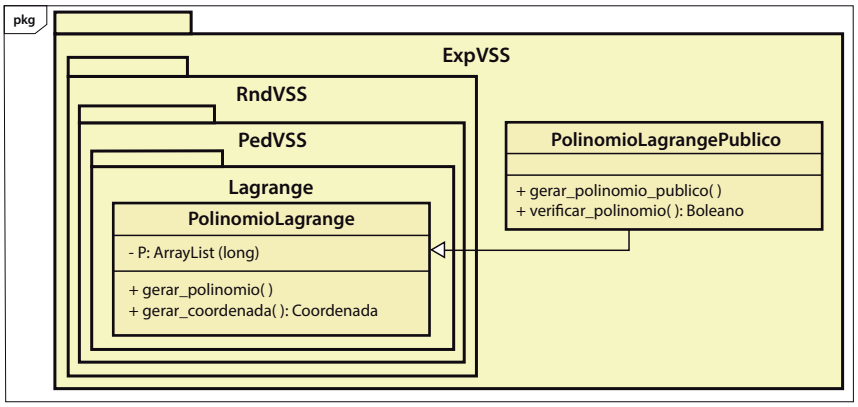

Figura 6. Diagrama de classes do módulo ExpVSS

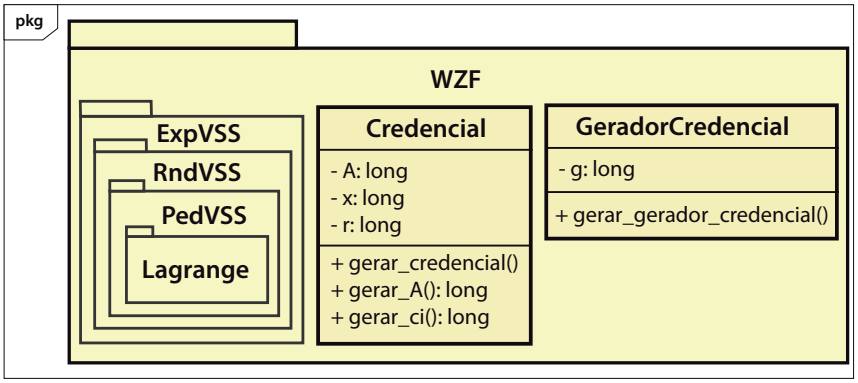

Figura 7. Diagrama de classes do módulo WZF

4.3. Testes Realizados

Para a verificação dos módulos implementados foram realizados testes. Os testes ocorreram de forma sequencial, com as informações de cada participante determinadas dentro da sequência de testes. Etapas onde mais de um participante deveria participar em conjunto foram realizadas utilizando um laço de repetição *Para*, repetindo a etapa até todos os procedimentos serem realizados por cada participante. Todos os testes foram realizados usando três participantes, com qualquer informação pessoal gerada de forma aleatória.

O único módulo não testado foi o de Lagrange, pois este já se encontra implementado no sistema de votações CIVIS, e sua implementação não fez parte deste trabalho. O teste do módulo WZF foi realizado sem um procedimento padrão de verificação da credencial gerada, de forma que não é possível que participantes verifiquem as credenciais recebidas. Entretanto, foi possível verificar seu valor durante o teste deste módulo realizando uma comparação da credencial gerada pelo módulo com o cálculo modelo matemático utilizado no CIVIS.

O primeiro módulo testado foi o PedVSS. Foi necessário definir um segredo para apenas um dos membros, e executar o método *compartilhar_segredo()*, gerando objetos do tipo *FracaoSegredo*, compartilhados entre os participantes. Cada membro utiliza o método *verificar_fracao_segredo()* do objeto do tipo *Comprometimento* para verificar os objetos recebidos, resultando em um valor boleano. O resultado do teste é satisfatório apenas caso todas as verificações resultem no valor boleano verdadeiro.

O segundo módulo testado foi o RndVSS. O teste consiste de execuções repetidas do PedVSS, para cada participante, e após todos os compartilhamentos serem realizados corretamente, é necessário gerar os objetos dso tipos *ParteSegredo* e *ComprometimentoGrupo*. O resultado do teste é satisfatório apenas caso todas as verificações, de cada compartilhamento antes da geração dos objetos dos tipos *ParteSegredo* e *ComprometimentoGrupo* resultem no valor boleano verdadeiro.

O terceiro módulo testado foi o ExpVSS. O teste consiste da execução do módulo anterior, mas gerando um objeto do tipo *PolinomioLagrangePublico*. Para gerar tal objeto, é necessário receber o valor de cada *PolinomioLagrange* utilizado nos compartilhamentos do módulo PedVSS. Sua verificação ocorre pelo método *verificar_polinomio()*. O resultado do teste é satisfatório apenas caso o módulo RndVSS e a verificação dos termos do objeto de tipo *PolinomioLagrangePublico* resultem no valor boleano verdadeiro.

O último módulo testado foi o WZF. Seu teste consiste da comparação entre os termos $((g_A)^a)^{c^{-1}} = (g_1 g_3^x)^{\frac{1}{y+r}}$. Para tal, deveria ser gerado um objeto do tipo *Credencial*, e seus valores inicializados utilizando seus métodos. Este objeto seria comparado à uma credencial gerada pela recuperação de segredos distribuídos y, x e r. O resultado do teste é satisfatório apenas caso ambas as credenciais geradas sejam idênticas.

Como cada módulo é dependente do anterior, é necessário que todos os módulos anteriores obtenham resultados satisfatórios para que o módulo WZF finalize de forma correta. Desta forma, mesmo analizando individualmente cada um, todos os testes atingiram resultados corretos, demonstrando que a implementação gera credenciais dentro do formato necessário, de forma completamente distribuída.

4.4. Resultados e Colaborações

A principal colaboração deixada é o protocolo de geração de crendenciais de forma distribuída. As credenciais geradas por esse protocolo são verificáveis em tempo linear, pois seguem a mesma estrutura das utilizadas pelo protocolo de ABRTY, e são geradas de forma distribuída baseadas em uma adaptação do protocolo de WZF. Toda a arquitetura descrita também soma às colaborações, podendo ser utilizada para documentação ou como guia para desenvolvimento de adaptações no próprio protocolo.

Com toda a arquitetura do protocolo organizada e cada módulo testado e funcionando corretamente, todo o protocolo pode finalmente ser integrado ao sistema de votação CIVIS. Para tal, entretanto, se faz necessário adaptar páginas do sistema para receber tal solução, o que não será tratado neste trabalho. A disponibilização desta implementação em um repositório será realizada quando tal integração for concluída, pois o sistema de votações ainda está em desenvolvimento.

5. Considerações Finais

No sistema de votação CIVIS originalmente as credenciais de votação são emitidas por uma única autoridade. No entanto, o protocolo de Araújo et al. [Araújo et al. 2010], no qual o CIVIS se baseia, requer a geração dessas credenciais de forma distribuída. A fim de possibilitar a geração distribuída de credenciais, este trabalho adaptou o protocolo de assinatura digital distribuída de Wang, Zheng e Feng (WZF), possibilitando que esse possa ser utilizado para geração de credenciais verificáveis em complexidade linear.

Além disso, o trabalho apresentou uma implementação do protocolo adaptado. A implementação proposta satizfaz os requisitos do protocolo de ABRTY. No entanto, ainda são necessárias provas designadas de verificação para que o votante verifique sua credencial ao recebê-la. Além disso, ainda são necessários a integração da implementação ao CIVIS e a realização de testes posteriores. Esses são deixados como trabalhos futuros.

Referências

Araújo, R., Rajeb, N., Robbana, R., Traoré, J., and Yousfi, S. (2010). Towards practical and secure coercion-resistant electronic elections. In *Cryptology and Network Security - 9th International Conference, CANS 2010*, volume 6467, pages 278–297. Springer.

Araújo, R., Neto, A., and Traoré, J. (2018). Civis - a coercion-resistant election system. In *Anais do XVIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 29–42, Porto Alegre, RS, Brasil. SBC.

Boneh, D. and Boyen, X. (2004). Short signatures without random oracles. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 56–73. Springer.

Gennaro, R., Jarecki, S., Krawczyk, H., and Rabin, T. (1999). Secure distributed key generation for discrete-log based cryptosystems. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 295–310. Springer.

Holovaty, A. and Kaplan-Moss, J. (2009). *The definitive guide to Django: Web development done right*. Apress.

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

Pedersen, T. P. (1991). A threshold cryptosystem without a trusted party. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 522–526. Springer.

Shamir, A. (1979). How to share a secret. *Communications of the ACM*, 22(11):612–613.

Silva Neto, A. A. and Araújo, R. (2017). A integração do criptossistema el gamal limiar ao sistema de votação via internet civis. In *WTICG - SBSeg 2017*, pages 697–706, Brasília, DF, Brasil. SBC.

Silva Neto, A. A., Sá, M. O. L. d., Araújo, R. S. d. S., Mota, M. P., Sampaio Neto, N. C., 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.

Souheib, Y., Stephane, D., and Riadh, R. (2012). Watermarking in e-voting for large scale election. In *Multimedia Computing and Systems (ICMCS), 2012 International Conference on*, pages 130–133. IEEE.

Van Rossum, G. and Drake, F. L. (2003). *Python language reference manual*. Network Theory United Kingdom.

Wang, H., Zhang, Y., and Feng, D. (2005). Short threshold signature schemes without random oracles. In *International Conference on Cryptology in India*, pages 297–310. Springer.