Uma Proposta de Integração da Biblioteca Criptográfica do Sistema Helios ao Sistema Civis

Paula Rosene dos Santos[1], **Roberto Samarone Araujo**[1]

[1]Laboratório de Segurança e Criptografia Aplicada (LabSC)
Universidade Federal do Pará (UFPA)
Belém – PA – Brasil

paula.santos@icen.ufpa.br, rsa@ufpa.br

Abstract. *Internet voting systems are an alternative to traditional electronic elections. In order to guarantee security, solutions such as Helios and Civis employ advanced mechanisms defined by means of their cryptographic libraries. These libraries share some criptographic primitives, but when compared, the Helios library has an inherent advantage. Thanks to its widespread use in elections, it has been constantly tested and validated. To benefit Civis with the advantages of the Helios library, this work introduces a proposal to integrate the Helios library into the Civis voting system. In particular, the ElGamal cryptosystem.*

Resumo. *Sistemas de votação via Internet têm mostrado-se uma alternativa às eleições eletrônicas tradicionais. De forma a garantir segurança, soluções como o Helios e o Civis utilizam-se de avançados mecanismos definidos por meio de suas bibliotecas criptográficas. Tais bibliotecas compartilham algumas primitivas criptográficas, mas quando comparadas, a biblioteca empregada no Helios possui uma vantagem inerente. Graças ao emprego do Helios na realização de eleições, sua biblioteca tem sido constantemente testada e validada. De forma a aprimorar o Civis com os benefícios da biblioteca do Helios, este trabalho introduz uma proposta de integração da biblioteca do Helios ao Civis. Em particular, o criptosistema ElGamal.*

1. Introdução

Nos últimos anos a sociedade vem sendo transformada por meio da digitalização de diversos serviços. Tal digitalização acentuou-se principalmente devido à pandemia de COVID-19. Eleições tradicionais foram substituídas por aquelas realizadas via Internet, por exemplo. Nesse contexto, sistemas de votação via Internet tornaram-se protagonistas na condução de processos eleitorais como a escolha de reitores em universidades, por exemplo, as eleições realizadas na [UFSCar 2023].

A adoção desses sistemas durante a pandemia não somente evitou o deslocamento de votantes aos locais de votação, mas trouxe mais conveniência aos eleitores. É imprescindível, no entanto, que sistemas desse tipo sejam seguros a fim de apresentar resultados que correspondam fielmente à vontade dos eleitores. Sistemas de votação considerados seguros como o Helios [Adida 2008] e o Civis [Araujo et al. 2018] são opções para a realização de eleições via Internet. Estes sistemas são disponibilizados via ambiente *Web* e utilizam-se de modernos recursos criptográficos para garantir segurança em votações.

Embora tais sistemas possuam o mesmo propósito (i.e. eles tornam possível realização de votações via Internet), existem diferenças quanto a seus objetivos de segurança. O Helios, por exemplo, possui um mecanismo de verificação fim-a-fim (E2E) [Cominetti et al. 2023] que permite aos votantes verificarem se seus votos foram corretamente criptografados. Diferentemente, o Civis possui recursos que provém aos votantes meios de reagirem a ataques coercivos (como aqueles em que um atacante acompanha o votante durante a emissão de seu voto). Além dessas diferenças, esses sistemas possuem arquiteturas e bibliotecas criptográficas distintas.

As bibliotecas criptográficas empregadas nesses sistemas, embora diferentes, possuem algumas similaridades quanto aos seus mecanismos criptográficos. Tanto o Helios como o Civis utilizam-se do criptosistema de chaves pública ElGamal [Gamal 1985] (baseado em aritmética modular) e de provas de conhecimento, por exemplo. Todavia, a biblioteca empregada no Helios dispõem de maior credibilidade. Como a literatura apresenta, existem inúmeros estudos sobre o sistema Helios. O seu criptosistema ElGamal, especialmente, foi analisado por [Laz et al. 2020]. Ademais, o fato desse sistema ser utilizado amplamente em eleições resulta em uma biblioteca criptográfica melhor testada e validada. Tais vantagens levaram aos autores do trabalho aqui apresentado a seguinte questão: é possível integrar a biblioteca criptográfica em uso no sistema Helios ao sistema Civis ? Essa integração traria benefícios ao Civis que poderia dispor das vantagens inerentes à biblioteca do Helios.

Este trabalho visa responder parcialmente à pergunta acima. Ele tem como principal contribuição uma proposta de integração do criptosistema ElGamal presente na biblioteca do sistema Helios ao sistema Civis. O trabalho está organizado da seguinte forma. A seção seguinte apresenta os trabalhos relacionados. Em seguida, a Seção 2 descreve brevemente os sistemas de votação Helios e Civis e suas arquiteturas. A arquitetura apresentada tem como foco as classes relativas ao criptosistema ElGamal. A Seção 3 introduz a proposta de integração das bibliotecas. Por fim, as conclusões e trabalhos futuros são apresentados na Seção 4.

1.1. Trabalhos Relacionados

O trabalho aqui apresentado propõe a integração do criptosistema ElGamal existente na biblioteca criptográfica do Helios ao sistema Civis. A literatura apresenta diversos trabalhos que abordam ambos os sistemas de votação. No entanto, ela não apresenta trabalhos similares ao aqui proposto.

O sistema Civis, particularmente, possui outros trabalhos que visam sua melhoria. O trabalho de [Neto and Araújo 2017] propõe a integração do criptosistema ElGamal limiar ao Civis. A integração proposta aqui, no entanto, utiliza-se da versão padrão do criptosistema ElGamal. O trabalho de [Sobrinho and Araújo 2019] aborda a implementação do protocolo proposto por [de Sá et al. 2020] para geração de credenciais anônimas no Civis. Como esse e os trabalhos anteriores requerem o criptosistema ElGamal, eles têm relação com à proposta apresentada aqui.

Assim como o Civis, o sistema Helios possui vários trabalhos relacionados. Todavia, considerou-se aqui as propostas que introduzem melhorias ao sistema original. Por exemplo, o trabalho de [Cortier et al. 2013] propõe uma variação do sistema Helios para geração de chaves distribuídas. Enquanto, o trabalho de [Bulens et al. 2011] introduz uma

versão do sistema Helios baseada em redes de misturadores. Considera-se aqui a versão do Helios empregada por [Pereira 2016]. Tal versão incorporou diversas melhorias que asseguram confiabilidade dos votos por meio de criptografia distribuída.

2. Os Sistemas de Votação Helios e Civis

A seguir, parte das funcionalidades e das arquiteturas desses sistemas são brevemente apresentadas. As descrições das arquiteturas, particularmente, têm como foco o criptosistema ElGamal definido nas bibliotecas criptográficas.

2.1. O Sistema de Votação Helios

O sistema Helios foi desenvolvido por meio do *framework Django* e das linguagens de programação *Python* e *JavaScript*. Ele utiliza o JSON (*JavaScript Object Notation*) para troca de dados entre os usuários (clientes) e os servidores do sistema. Para garantir segurança em uma eleição, o Helios emprega um conjunto de recursos criptográficos como o criptosistema ElGamal [Gamal 1985] e provas de conhecimento não interativas (NIZKP).

Uma votação no Helios é configurada pelo administrador da eleição. Ele estabelece os parâmetros eleitorais e define se será a única autoridade eleitoral ou se existirão outras autoridades. No primeiro caso, o administrador controla todo o processo eleitoral, incluindo a apuração dos votos. No segundo caso, o administrador define outras autoridades para realizar a apuração dos votos.

A definição do administrador da eleição permite a realização das etapas de configuração, votação e apuração. Na etapa de configuração, o administrador define os parâmetros gerais da eleição, como disputa e opções de voto, bem como realiza o cadastramento de votantes aptos a votar e das autoridades. Nessa etapa, também ocorre a geração dos pares de chave.

A etapa seguinte é a de votação. Nesta etapa, cada votante escolhe sua opção de voto. O voto é então criptografado pelo seu computador (via criptosistema ElGamal) e o votante opta por submeter esse voto (criptografado) ao sistema (servidor) ou auditá-lo. Em caso de auditoria, o votante pode verificar se a opção criptografada é a mesma escolhida. Esse voto é descartado e o votante pode votar novamente.

Por fim, na etapa de apuração, a autoridade realiza a soma homomórfica dos votos (por meio da propriedade de homomorfismo do criptosistema ElGamal) e emprega sua chave secreta para descriptografar o resultado dessa soma. Caso seja empregado um conjunto de autoridades, elas cooperam a fim de descriptografar esse resultado.

Arquitetura do Sistema Helios

A arquitetura do sistema Helios segue o modelo cliente-servidor. Dessa forma, a sua biblioteca criptográfica é dividida em duas partes. A primeira parte é voltada para operações criptográficas realizadas no servidor e foi desenvolvida por meio da linguagem de programação Python. A segunda parte possui operações criptográficas a serem realizadas no cliente e foi desenvolvida na linguagem de programação JavaScript.

As classes relativas às operações criptográficas realizadas no servidor estão organizadas em arquivos no diretório *crypto*. Esse diretório contém os arquivos electionalgs.py, numtheory.py e utils.py, algs.py e elgamal.py. O arquivo numtheory.py possui

funções relativas a operações matemáticas envolvendo teoria dos números. O arquivo electionalgs.py possui classes relativas às operações relacionadas à eleição como *Election*, *Voter*, *CastVote*. O arquivo utils.py possui apenas duas funções: o gerador de números pseudoaleatórios e uma função *hash* criptográfico. A Figura 1 mostra o diagrama de classes simplificado existente no arquivo elgamal.py, bem como o relacionamento entre elas. O arquivo algs.py contém o algoritmo ElGamal [Gamal 1985] e provas de conhe-

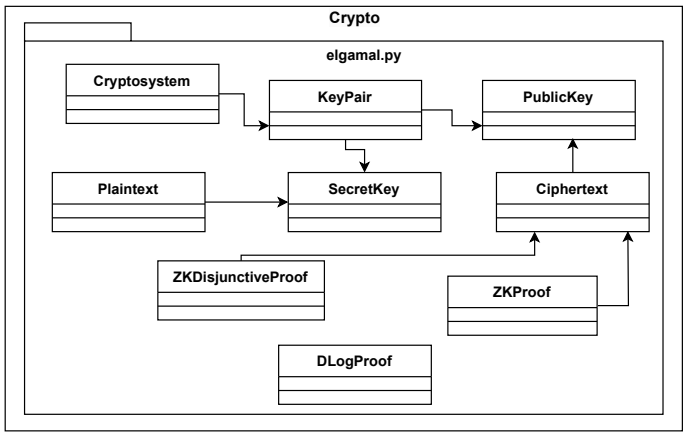

Figura 1. Diagrama de classes do arquivo ElGamal.py (Servidor).

cimento não interativas. Ele contém as seguintes classes: ElGamal, EGKeypair, EPublicKey, EGSecretkey, EGPlaintext, EGCiphertext, EGZKProof, EGZKDisjunctiveProof e DLogProof. O arquivo elgamal.py é uma cópia do arquivo algs.py e objetiva facilitar a modularidade do sistema. Ele possui as classes Cryptosystem, KeyPair, PublicKey, SecretKey, Plaintext, Ciphertext, ZKProof, ZKDisjunctiveProof e DLogProof.

As classes destinadas às operações criptográficas realizadas no cliente localizam-se em arquivos no diretório *jscrypto*. Esse diretório contém os arquivos: bigint.dummy.js, bigint.java, bigint.js, class.js, helios.js, jsbn.js, jsbn2.js, random.js, sha1.js, sha2.js, e elgamal.js. Parte desses arquivos pertencem a bibliotecas externas como a jsbn [Wu 2013].

Os arquivos bigint.dummy.js, bigint.java e bigint.js possuem classes de operações matemáticas com números inteiros. O arquivo class.js contém as definições base para classes no JavaScript. Os arquivos jsbn.js e jsbn2.js fazem parte da biblioteca de números inteiros jsbn. Os arquivos random.js, sha1.js e sha2.js possuem as funções de geradores de números pseudoaleatório e de hash criptográfico. No arquivo helios.js estão as classes Election, EncryptedAnswer, EncryptedVote, Tally e Trustee que estão relacionadas ao processo das eleições. A Figura 2 apresenta o diagrama de classes simplificado do arquivo elgamal.js.

A estrutura funcional de uma eleição no sistema Helios pode ser subdividida da seguinte forma: configuração, votação e apuração. Estas etapas são descritas a seguir pela perspectiva dos relacionamentos das classes do seu criptosistema.

Fase de Configuração. De forma a empregar o criptosistema ElGamal a partir de sua biblioteca, o Helios faz uso da classe *ElGamal.Params* para carregar os parâmetros pré-definidos no arquivo views.py do sistema. Para a geração dos pares de chaves das autoridades executa-se o método *generate* da mesma *Elgamal.Params*, após isso, as chaves

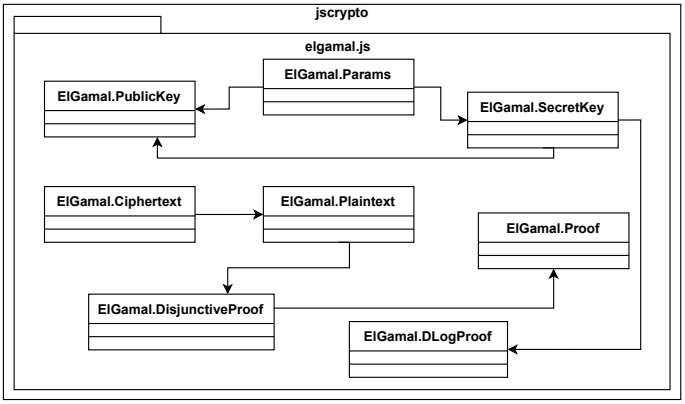

Figura 2. Diagrama de classes do arquivo elgamal.js (Cliente).

são codificadas por meio das classes *ElGamal.PublicKey* e *ElGamal.SecretKey*.

Provas de conhecimentos não interativas (NIZKP) relativas às chaves criadas por essas autoridades é executada pelo método *proveKnowledge* da classe *ElGamal.SecretKey*, como resultado retorna um objeto da classe *ElGamal.DLogProof*. Após isso, a autoridade faz o *download* de sua chave privada disponibilizada no formato JSON. Em seguida, a chave pública pode ser carregada ao servidor Helios, enquanto a chave secreta salva pela autoridade (no cliente) deve ser mantida em segredo.

Fase de Votação. Após o congelamento da eleição pela autoridade, os votantes podem emitir seus votos criptografados. Para isso, o sistema Helios utiliza a classe *Elgamal.Plaintext* para realizar o tratamento e encapsulamento dos votos. Por meio do método *encrypt* dessa mesma classe (Elgamal.Plaintext) os dados são criptografados, a criptografia resultante retorna um objeto texto cifrado da classe *ElGamal.Ciphertext*. Logo após a emissão do voto do eleitor, a NIZKP de validade da cédula emitida pelo votante é executada. Essa validação é realizada por meio da classe *ElGamal.Proof* e *ElGamal.DisjunctiveProof* que contêm todas as funções e os métodos utilizados para essas verificações.

Fase de Apuração. Na etapa de apuração a autoridade de apuração realiza o *upload* da sua chave secreta no sistema e gera a descriptografia parcial. Caso haja mais de uma autoridade, cada uma delas deve realizar esse processo. Em seguida, NIZKP para prova que o apurador contribui honestamente na descriptografia dos votos é realizada por meio do método *decryptAndProve* da classe *ElGamal.SecretKey*. A descriptografia ocorre por meio do método *decrypt* da classe *ElGamal.SecretKey*. Após o fim do processo de descriptografia, o servidor Helios realiza a integração de todos os resultados parciais e disponibiliza a apuração completa da eleição.

2.2. O Sistema de Votação Civis

O Civis [Araujo et al. 2018] é um sistema de votação via *web* que possibilita aos votantes resistência a ataques coercivos. Assim como o Helios, ele foi desenvolvido na linguagem de programação *Python*, por meio do *Framework* Django, e *JavaScript*. Além disso, o sistema utiliza o JSON (JavaScript Object Notation) para troca de mensagem e para disponibilizar informações públicas sobre a eleição. O sistema é uma implementação do protocolo de votação resistente à coerção de ABRTY [Araújo et al. 2010]. Assim, ele

utiliza mecanismos criptográficos como o criptosistema ElGamal [Gamal 1985], provas de conhecimento não interativas (NIZKP) e redes de misturadores.

Uma eleição no Civis envolve três tipos de autoridades eleitorais. A autoridade da eleição (AE) que estabelece os parâmetros da eleição. A autoridade de registro (registrador) que emite credenciais legítimas de votação. Além disso, existe o papel do apurador responsável pela identificação das credenciais legítimas e apuração dos resultados da eleição. Embora o sistema requeira diversas autoridades, é possível definir uma única autoridade que possui todos esses papéis.

Além das autoridades, o sistema requer um administrador da eleição. Ele tem o papel de controlar o processo eleitoral estabelecido dentro das quatro etapas requeridas pelo sistema: configuração, registro, votação e apuração. Na fase de configuração, são cadastradas as autoridades e os votantes aptos a receberem suas credenciais legítimas. Além disso, são gerados o par de chave ElGamal necessário ao sistema e estabelecidos outros parâmetros com disputas e opções.

Na fase de registro, são geradas as credenciais legítimas de votação (uma senha para fins de ilustração) aos eleitores aptos. Para isso, o Civis utiliza-se do protocolo seguro para o registro de votantes SAST [Sá et al. 2020]. Após os votantes receberem suas credenciais, inicia-se a fase de votação. Nesta fase, os eleitores realizam suas escolhas e utilizam suas credenciais legítimas (recebidas na fase de registro) para votar. Caso um votante esteja sob um ataque coercivo (e.g. um atacante observa o votante enquanto ele vota), o votante deve utilizar uma credencial diferente da legítima para votar. O voto emitido com essa credencial será descartado na apuração.

O sistema permite que o eleitor emita seu voto mais de uma vez, apurando apenas o último voto emitido com a credencial recebida no registro. A distinção entre credenciais legítimas e ilegítimas ocorre na fase de apuração. O processo é realizado utilizando uma rede de misturadores para anonimizar votos e credenciais. Os votos correspondentes às credenciais legítimas são apurados e o resultado final é divulgado.

Arquitetura do Sistema Civis

O sistema Civis foi concebido de forma modular, possuindo três módulos principais: administração, votação e quadro público. Esses módulos podem ser desativados ou ativados em paralelos quando necessário [Araujo et al. 2018]. O módulo de votação, por exemplo, pode ser ativado somente durante o período da votação. Cada módulo tem funcionalidades definidas. O módulo administrativo controla as fases de uma eleição, exceto a votação. O módulo de votação possibilita a emissão de votos enquanto que o quadro público disponibiliza os dados públicos. O sistema segue o modelo cliente-servidor e vários arquivos facilitam a interação com a biblioteca criptográfica do sistema.

Os arquivos que compreendem as classes de processamento no *servidor* estão organizadas no diretório cripto. Esse diretório contém os arquivos ElGamalSg e VerificaProvas. O arquivo ElGamalSg contém classes relativas ao criptosistema ElGamal, ou seja, o ElGamalSgKeyPairGen, o ElGamalSgPK, o ElGamalSgSK e o ElGamalSgFunc. O arquivo VerificaProvas possui funções responsáveis pela verificação das provas de conhecimento não interativas (NIZKP). A Figura 3 apresenta o diagrama de classes do criptosistema ElGamal (simplificado) empregado no servidor do Civis.

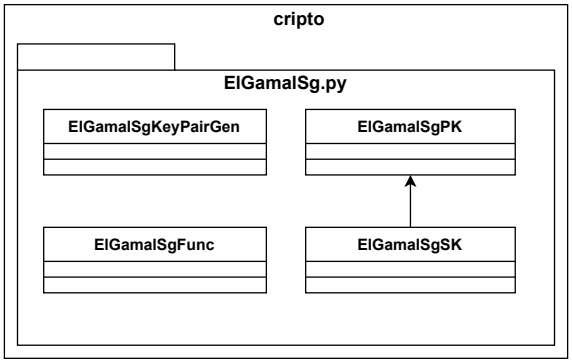

Figura 3. Diagrama de classes do arquivo ElgamalSg.py (Servidor).

As classes pertencentes às operações criptográficas realizadas no *cliente* estão localizadas no diretório protocolo. Tal diretório contém os subdiretórios lib e elgamal. O subdiretório lib contém as bibliotecas externas necessárias ao sistema. No subdiretório elgamal encontram-se os arquivos chaves.js, credencias.js, disputas_configuracao.js, Provas-Civis.js, votos.js.

Os arquivos chaves.js, disputas_configuracao.js e credenciais.js contém funções que estabelecem os pares de chaves das autoridades, disputas e opções, e credenciais. O arquivo credenciais, particularmente, contém as funções de geração de credenciais. As provas de conhecimento e cédulas de votação são realizadas por meio das funções contidas nos arquivos Provas-Civis.js e votos.js. A Figura 4 apresenta o diagrama de classes (simplificado) que relaciona o criptosistema ElGamal (definido na biblioteca criptográfica do Civis) às outras classes do sistema.

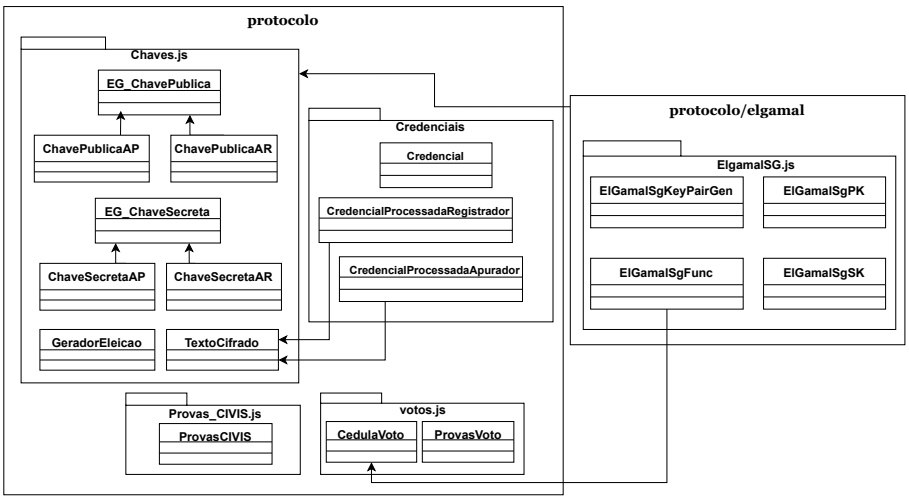

Figura 4. Diagrama de classes dos arquivos do diretório protocolo (Cliente).

O diretório elgamal contém os arquivos aleatorio.js e ElGamalSG.js. O arquivo aleatorio.js possui funções para a geração de números pseudoaleatório. O arquivo ElGamalSG.js possui as classes ElGamalSgKeyPairGen, ElGamalSgPK e ElGamalSgSK. Tais classes são responsáveis pela geração das chaves assimétricas. A classe ElGamalSgFunc contêm métodos para realização de operações de encriptação, reencriptação e decriptação.

A seguir são apresentadas as fases de uma eleição no contexto da interação entre elas e as classes relativas ao criptosistema ElGamal.

Fase de Configuração. Nesta etapa, as autoridades registrador e apurador geram seus pares de chaves. Para isso, um objeto da classe ElGamalSgKeyPairGen (arquivo ElgamalSG.js) é instanciado e por meio do método keyPairGen são gerados parâmetros como números primos e geradores. A geração do par de chave das autoridades ocorre por meio do método *gerar_par_chaves* pertencente a mesma classe (ElGamalSgKeyPairGen). Tais chaves são armazenadas por meio das classes ElGamalSgPK e ElGamalSgSK. Por conseguinte, as classes EG_ChavePublica e EG_ChaveSecreta contêm os métodos e atributos acessados pelas classes ChavePublicaAP e ChaveSecretaAP. Estas codificam o par de chaves da autoridade de apuração. Enquanto, as classes ChavePublicaAR e ChaveSecretaAR codificam o par de chave do registrador.

Fase de Registro. A autoridade de registro emite as credenciais legítimas para os eleitores. Para isso, ela utiliza a sua chave secreta criada anteriormente. A autoridade gera as credenciais por meio das classes Credencial e CredencialProcessadaRegistrador do arquivo credenciais.js.

Fase de Votação. O eleitor seleciona as suas opções de voto e informa a sua credencial de votação. Nesse processo, o voto e a credencial são criptografados por meio do criptosistema ElGamal e também são geradas NIZKPs. A criptografia dos votos é realizada por meio do método EGEncriptar da classe ElGamalSgFunc (arquivo ElGamalSG.js). As NIZKPs são geradas por meio da classe ProvasCivis (arquivo ProvasCivis.js).

Fase de Apuração. O registrador e o apurador utilizam suas chaves secretas para identificar os votos duplicados submetidos com a mesma credencial. Para a remoção de tais votos, o registrador e o apurador utilizam-se do método processar_credencial das classes CredencialProcessadaRegistrador e CredencialProcessadaApurador do arquivo credenciais.js. Para descriptografar os votos, o Civis utiliza o método EGDecriptar da classe ElGamalSgFunc (arquivo ElGamalSG.js).

3. Proposta de Integração entre o Sistema Civis e a Biblioteca Criptográfica do Sistema Helios

As bibliotecas criptográficas dos sistemas Helios e Civis contêm várias primitivas criptográficas. Algumas dessas primitivas são comuns em ambos os sistemas, como o criptosistema ElGamal. Outras, como a rede de misturadores, são empregadas apenas pelo Civis. A seguir é apresentada a integração do criptosistema ElGamal do sistema Helios ao sistema Civis. A maior parte da integração ocorreu no cliente, uma vez que ambos os sistemas realizam a maioria das operações criptográficas no cliente ao invés do servidor.

3.1. Integrando o Processamento Realizado no Servidor

A cada eleição no sistema Civis, a fase de configuração requer a geração de novos parâmetros p, q, g para o grupo numérico utilizado no criptosistema ElGamal. O sistema Helios, diferentemente, utiliza parâmetros (p, q, g) fixos para eleições distintas.

Além de gerar valores p, q, g únicos para cada eleição, o Civis requer três geradores do grupo numérico g_1, g_3, o. Esses valores são gerados no servidor Civis por meio

da classe ElGamalSgKeyPairGen (arquivo ElGamalSg.py). De modo a integrar o criptosistema ElGamal do sistema Helios ao Civis, é necessário utilizar os valores fixos p, q, g utilizados no Helios nas variáveis de ambiente do sistema Civis. Agora, a partir dos novos parâmetros p, q, g definidos no Civis, o sistema utiliza-se de sua classe ElGamalSgKeyPairGen para gerar os valores g1, g3, o. A geração de novos parâmetros a cada eleição requer a realização de testes a fim de verificar a segurança destes. Ao utilizar parâmetros fixos, evita-se tais testes agilizando assim a fase de configuração.

3.2. Integrando o Processamento Realizado no Cliente

Como descrito na Seção 2.2, no Civis as classes que interagem com o criptosistema ElGamal estão dispostas nos arquivos chaves.js, credenciais.js, provas.js e votos.js. Assim, de forma a utilizar o criptosistema ElGamal do Helios, é necessário alterar tais arquivos. O arquivo ElGamalSG.js, assim, tornou-se uma interface para a abstração do material criptográfico ElGamal do sistema Helios. A Figura 5 ilustra o diagrama de classes resultante da integração do criptosistema ElGamal do Helios ao sistema Civis. A sequência numérica indica as etapas de integração.

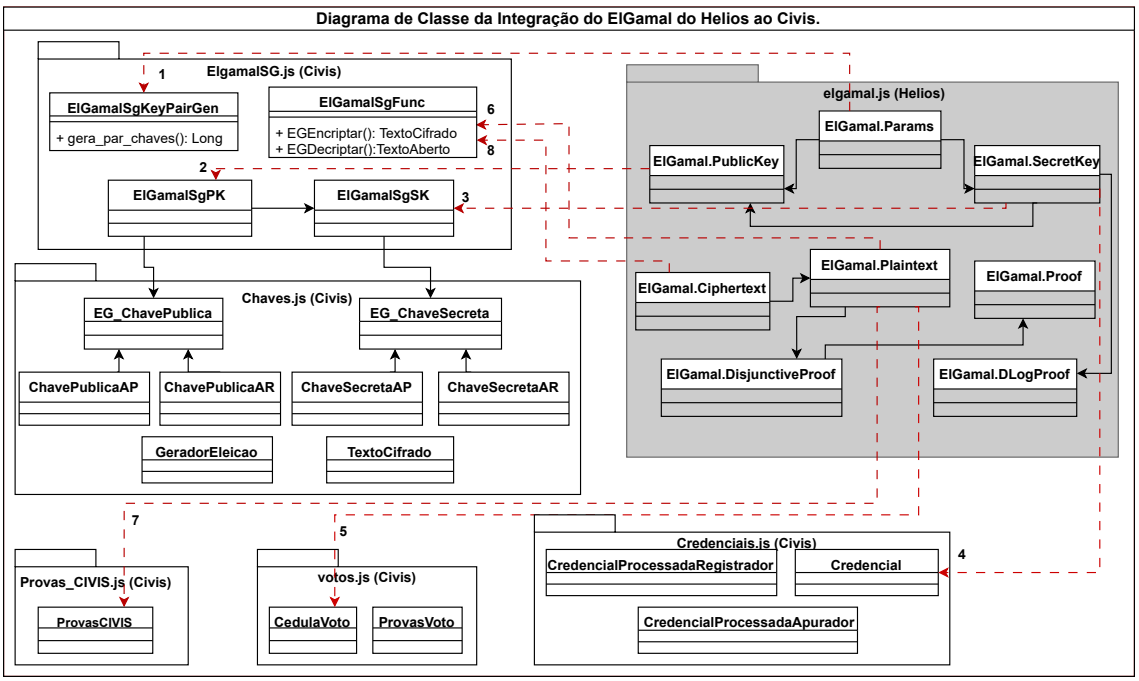

Figura 5. Diagrama de classes da integração do ElGamal do Helios ao Civis.

No Civis a fase de configuração de uma eleição requer a (geração) dos parâmetros p, q, g (e.g. o gerador g e os primos p, q). Esses parâmetros são relativos ao grupo numérico necessário ao criptosistema ElGamal. Assim como na integração anterior, o Civis utiliza os mesmos parâmetros fixos estabelecidos no Helios. Para integrar o criptosistema ElGamal do Helios ao Civis, é necessário realizar a leitura desses parâmetros p, q, g em todos os arquivos do Civis que fazem uso do criptosistema ElGamal. A definição dos parâmetros p, q, g possibilita a geração dos pares de chaves ElGamal em ambos os sistemas. No Civis, a geração do par de chaves ElGamal é realizada por meio do método gerar_par_chaves da classe ElGamalSgKeyPairGen (arquivo ElgamalSG.js).

De forma integrar o criptosistema ElGamal do Helios, propõem-se instanciar no método (gerar_par_chaves) um objeto da classe ElGamal.Params da biblioteca do Helios (arquivo elgamal.js) e executar o método generate dessa classe (etapa 1).

Logo, os parâmetros gerados são repassados para as classes ElGamalSgPK e ElGamalSgSK. Essas classes, ao integrar o ElGamal do Helios, devem receber as instâncias das classes ElGamal.PublicKey e ElGamal.SecretKey, etapas 2 e 3 respectivamente. Após a execução do método generate da biblioteca do Helios no método gerar_par_chaves do Civis, os parâmetros de chaves gerados são codificados segundo a autoridade a qual pertence às chaves. A geração das credenciais requer a utilização de alguns parâmetros de chave ElGamal gerados anteriormente (e.g. os valores p, q, g). O arquivo credenciais.js do Civis contém a classe responsável pela geração e tratamento das credenciais dos votantes. Esta classe (Credencial) requer os parâmetros p, q, g e a chave privada da autoridade de registro. Esses parâmetros são passados por meio de uma instância da classe *ElGamal.SecretKey* do Helios (etapa 4).

Durante a fase de votação, os métodos relativos ao criptosistema ElGamal são utilizados pelo Civis para criptografia dos votos, por exemplo. A fim de realizar essa operação do Civis via biblioteca do Helios, é necessário realizar a instanciação de um objeto da classe Elgamal.Plaintext na classe cedulaVoto do (arquivo votos.js), etapa 5. Além disso, é necessário instanciar um objeto da classe Elgamal.Plaintext do Helios no método EGEncriptar da classe ElGamalSgFunc do Civis, etapa 6. Por fim, executa-se o método ElGamal.encrypt para criptografar o dado. O resultado da criptografia é então encapsulado em um objeto da classe TextoCifrado contido no arquivo chaves.js do Civis.

Após a encriptação dos votos, o sistema Civis precisa gerar um conjunto de provas de conhecimento não interativas (NIZKP). Elas evidenciam, por exemplo, que os dados foram corretamente encriptados. Várias dessas provas são comuns a ambos os sistemas, mas existem algumas provas particulares a cada sistema. O Civis faz uso da classe ProvasCivis onde todas as provas necessárias ao sistema são definidas. Tais classes estão presentes no arquivo Provas_Civis.js. A integração do criptosistema ElGamal do Helios ao Civis requer a substituição da prova de validade do texto aberto do Civis pelas correspondentes do Helios. Isso requer a instanciação no método gerarPValidVotoEnc (classe ProvasCivis) de um objeto da classe ElGamal.Plaintext do sistema Helios (etapa 7).

Por fim, é necessário descriptografar os votos e credenciais durante a apuração no Civis. Para isso, utiliza-se o método EGDecriptar pertencente a classe ElGamalSgFunc (arquivo ElgamalSG.js). De forma a utilizar-se do método de descriptografia do Helios, instancia-se um objeto da classe ElGamal.Ciphertext no método EGdecriptar do sistema Civis (etapa 8).

4. Conclusões e Trabalhos Futuros

Este trabalho apresentou uma proposta de integração da biblioteca criptográfica utilizada pelo sistema Helios ao sistema Civis. Tal integração responde parcialmente à questão proposta anteriormente pois apenas o criptosistema ElGamal foi integrado.

Além do criptosistema ElGamal, a biblioteca criptográfica do Helios dispõem de outras primitivas (e.g. como NIZKPs) que podem ser integradas ao Civis. Integrações desse tipo são deixadas como trabalho futuro. Avaliações empíricas como estudos de caso em ambientes reais de votação também serão realizadas adiante.

Referências

Adida, B. (2008). Helios: Web-based open-audit voting. In van Oorschot, P. C., editor, *Proceedings of the 17th USENIX Security Symposium, July 28-August 1, 2008, San Jose, CA, USA*, pages 335–348. USENIX Association.

Araújo, R., Ben Rajeb, N., Robbana, R., Traoré, J., and Youssfi, S. (2010). Towards practical and secure coercion-resistant electronic elections. In *Cryptology and Network Security: 9th International Conference, CANS 2010, Kuala Lumpur, Malaysia, December 12-14, 2010. Proceedings 9*, pages 278–297. Springer.

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

Bulens, P., Giry, D., and Pereira, O. (2011). Running Mixnet-Based elections with helios. In *2011 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 11)*, San Francisco, CA. USENIX Association.

Cominetti, E. L., Junior, M. A. S., Matias, P., and Araújo, R. (2023). Sistemas de votação fim-a-fim: Teoria e prática. In *Minicursos do XXIII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg 2023)*, pages 51–102, Porto Alegre, RS, Brasil. SBC.

Cortier, V., Galindo, D., Glondu, S., and Izabachène, M. (2013). Distributed elgamal à la pedersen: Application to helios. In Sadeghi, A. and Foresti, S., editors, *Proceedings of the 12th annual ACM Workshop on Privacy in the Electronic Society, WPES 2013, Berlin, Germany, November 4, 2013*, pages 131–142. ACM.

de Sá, M., Araujo, R., Sobrinho, A., Neto, A. S., Maximino, G., and Traoré, J. (2020). How colored passwords can improve the usability of coercion-resistant internet voting systems. In *Anais do XIX Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais*, pages 436–441, Porto Alegre, RS, Brasil. SBC.

Gamal, T. E. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory*, 31(4):10–18.

Laz, M. E., Grégoire, B., and Rezk, T. (2020). Security analysis of elgamal implementations. In Samarati, P., di Vimercati, S. D. C., Obaidat, M. S., and Ben-Othman, J., editors, *Proceedings of the 17th International Joint Conference on e-Business and Telecommunications, ICETE*, pages 310–321. ScitePress.

Neto, A. S. and Araújo, R. (2017). A integração do criptossistema el gamal limiar ao sistema de votação via internet civis. In *Anais do XVII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pages 697–706, Porto Alegre, RS, Brasil. SBC.

Pereira, O. (2016). Internet voting with helios. In *Real-World Electronic Voting*, pages 293–324. Auerbach Publications.

Sobrinho, A. and Araújo, R. (2019). Aprimorando a segurança do sistema de votação civis através da geração distribuída de credenciais. In *Anais Estendidos do XIX Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pages 101–104, Porto Alegre, RS, Brasil. SBC.

Sá, M., Araújo, R., Sobrinho, A., and Traoré, J. (2020). Registro prático aplicado a um sistema de votação resistente à coerção. In *Anais do XX Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pages 408–421, Porto Alegre, RS, Brasil. SBC.

UFSCar (2023). Eleições - ufscar. `https://eleicoes.ufscar.br/`. Acessado em: Junho/2024.

Wu, T. (2013). Rsa and ecc in javascript. `http://www-cs-students.stanford.edu/~tjw/jsbn/`. Acessado em: Junho/2024.