TEMA: SISTEMA DE ELEIÇÃO ELECTRÓNICA COM UMA TECNOLOGIA DE SEGURANÇA ROBUSTA

Nunes Tchimúa Mucuata Rafael

Namibe, 2023

Índice

Introdução

Este trabalho, é um projeto de investigação que intitulado: “Sistema de Eleição electrónica com uma tecnologia de segurança robusta”, consubstancia-se na necessidade de implementação de iniciativas que visam a inovação tecnológica das sociedades. Desta feita, procurou-se construir um algoritmo constituído por técnicas matemáticas e criptográficas eficientes, que narram o funcionamento do sistema abordado, através de um código implementado na linguagem de programação em python. Fez-se recursos às bibliografias existentes de outros autores que abordaram assuntos relacionados ao tema, o que serviu de base para o alcance dos objectivos preconizados na nossa investigação.

O mesmo está estruturado em 3 capítulos, onde no primeiro fez-se a fundamentação teórica do tema, mediante a análise de diversas teoria e das conclusões chegadas pelos autores referenciados no trabalho, no segundo capítulo explicou-se com detalhes o procedimento metodológico aplicado durante a realização da pesquisa. Como se não bastasse, no final, foram apresentados os resultados da pesquisa, resultantes da execussão do programa, onde foi possível comprovar a eficácia do sistema, quer ser seja, a nível de funcionalidade tecnológica, bem como, em termos de segurança, o que permitiu provar a robustez do sistema contra qualquer acção maliciosa ou que visa contrapor os princípios de segurança arquitetados no sistema de eleição proposto.

Em anexo, colocou-se o código construído na linguagem de programação “python”, que foi devidamente testado, quanto à sua eficiência e funcionalidade, como garantia da concretização dos intentos do trabalho.

i. Problema de investigação

Como combinar as técnicas criptográficas recomendáveis, a fim de construir um sistema de eleição electrónica resistente contra ataques?

ii. Hipótese de investigação

A construção de um sistema de eleição electrónica com um esquema de segurança baseado na combinação das técnicas criptográficas RSA, Elgamal, Assinatura digital, Envelope digital e partilha secreta da chave, permite manter as propriedades criptográficas que tornam um sistema de eleição electrónica resistente contra ataques.

a) **Variável independente**: Construção de um sistema de eleição electrónica com um esquema de segurança baseado na combinação das técnicas criptográficas RSA, Elgamal, Assinatura digital, Envelope digital e partilha secreta da chave.
b) **Variável dependente**: propriedades criptográficas que tornam um sistema de eleição electrónica resistente contra ataques.

iii. Objectivos

✓ **Objectivo geral:**

Construir um sistema de eleição electrónica implementado em python através de código resultante da combinação de uma tecnologia de segurança robusta.

✓ **Objectivos específicos:**

a) Revisar a bibliografia relacionada a sistema de eleição electrónica com uma tecnologia de segurança robusta.
b) Analisar as conclusões de outros autores relativamente a sistemas de eleição electrónica com uma tecnologia de segurança robusta.
c) Construir o sistema de eleição electrónica proposto.
d) Explicar o processo de construção do sistema.
e) Executar o sistema, a fim de testar a sua funcionalidade e eficiência.
f) Interpretar os resultados gerados pelo sistema e tirar conclusões acerca da proposta.

iv. Objeto de estudo da investigação

Estudo da arquitectura e segurança de um sistema de eleição electrónica.

v. Campo de acção

Segurança de um sistema de eleição electrónica.

vi. Limitação da pesquisa

Pese embora a necessidade de implementação de um sistema eleitoral electrónico, a investigação em causa, limita-se apenas na província do Namibe.

vii. Justificativa da investigação

O país tem enfrentado um crescimento demográfico desafiador, e por outra, as sociedades actualmente têm feito recurso à tecnologia para tornar a vida em sociedade cada vez mais dinâmica.

Um sistema de eleição electrónica trás consigo enumeras vantagens, que seja, em circuitos de eleição micro, assim como em circuitos de eleição macro. Por outra, uma das maiores vantagens que estes sistemas trazem, está na garantia da confiabilidade dos resultados.

Não obstante a isso, a implementação de um sistema de eleição electrónica, requer a observância dos requisitos de segurança com um grau de rigor, em todas as fases do processo eleitoral.

Portanto, estas são principais razões que estiveram na base da realização deste trabalho.

1. FUNDAMENTAÇÃO TEÓRICA

Num sistema de eleição electrónica, a segurança é garantida pela Criptografia, ciência que estuda as formas de ocultar uma mensagem e torna-la inteligível de para indivíduos não autorizados. Actualmente a Criptografia adotou métodos modernos, a Criptografia de chave pública. A Criptografia está subdividida em duas grandes áreas: A simétrica e a assimétrica. Uma faz o uso de uma única chave e outra usa um par de chaves (Costa, 2010).

1.1. Criptografia Simétrica

A criptograa simétrica é uma área da Criptografia que utiliza uma chave privada no processo de cifração/decifração da mensagem. Sendo privada, a chave deve ser partilhada entre o emissor e recetor de forma antecipada, e este processo (de partilha) exige a utilização de canais de comunicação seguros. Se um intruso (atacante) intersectar a chave partilhada, a política de privacidade da comunicação entre ambos ca comprometida. Daí que é necessário a geração e utilização de uma chave criptograficamente segura. Isto é, a segurança criptográfica não depende intrinsecamente da cifra a ser utilizada. Mas sim, a segurança está mais interligada com a segurança da chave (Barroso, 2016).

O algoritmo de uma cifra simétrica tem o seguinte modelo criptográfico (Barbosa, 2017):

(a) Texto claro: essa é a mensagem ou dados originais, que servem como entrada do algoritmo de cifração.

(b) Algoritmo de cifração: realiza diversas substituições e transformações no texto claro (a cifrar).

(c) Chave secreta: é um parâmetro de entrada para o algoritmo de cifração. A chave é um valor independente do texto claro e do algoritmo, usado para cifrar o texto claro. O algoritmo transformará um parâmetro de saída diferente do texto claro.

(d) Texto cifrado: essa é a mensagem transformada (impercetível), produzida como saída do algoritmo de cifração. Ela depende do texto claro e da chave secreta. O texto cifrado é um conjunto de dados aparentemente aleatório (pseudoaleatório) e,

para determinada mensagem, duas chaves diferentes produzirão dois textos cifrados distintos.

(e) Algoritmo de decifração: esse é basicamente o algoritmo de cifração executado de modo inverso. Este algoritmo permite recuperar o texto claro (legível).

1.3. Criptografia Assimétrica

A Criptografia assimétrica (de chave pública) foi inventada por Whitfield Diffie e Martin Hellman em 1976. São utilizadas duas chaves, uma privada e outra pública. Somente a chave pública é acessível a todos os intervenientes na comunicação. Não é feita a partilha de chaves, pois cada agente detém de um par de chaves, uma para cifrar e outra para decifrar.

A Criptografia assimétrica vem resolver o problema da distribuição da chave (simétrica) secreta. Cada agente gera um par de chaves de forma independente, e a técnica de armazenamento da chave privada é concebida e conhecida somente pelo titular. É comum ver a combinação das cifras, simétrica e assimétrica, isto é, podemos usar a Criptografia de chave pública para cifrar a chave de uma cifra simétrica, mantendo confidencial a referida chave (Maziero, 2019).

Os sistemas criptográficos de chave assimétrica baseiam-se na dificuldade que existe em se calcular a operação inversa de determinadas operações matemáticas. Existem problemas matemáticos a partir dos quais são construídos os algoritmos de chave pública:

a) Factorização de inteiros

Dado um número n resultado da multiplicação de dois números primos p e q, a dificuldade consiste em encontrar p e q tendo-se somente n. Para números de tamanhos pequenos, um ataque de força bruta pode facilmente encontrar a solução, mas para valores de n de grande tamanho (da ordem de 3.000 bits ou mais), a factorização de n seria pouco eficiente. A segurança do RSA (Rivest, Shamir e Adleman) baseia-se neste problema (Masthanamma e Pleya, 2015).

b) Logaritmo discreto (DLP-Discret Logarithm Problem)

Seja a equação $y \equiv g^x \ mod \ p$, onde g é um número inteiro positivo e *p* um número primo, ambos conhecidos.

A dificuldade consiste em, dado o valor de y calcular o valor de x. Não se conhece um algoritmo eciente que resolva este problema. Isto é, existe uma intratabilidade computacional do logaritmo discreto, e a segurança dos esquemas criptográficos Elgamal e Diffie Hellman baseiam-se neste problema.

1.3.1. Assinatura digital

A assinatura digital é um processo de assinatura eletrónica baseado em um sistema criptográfico assimétrico composto de um algoritmo, mediante o qual é gerado um par de chaves, uma das quais, privada e outra pública. Essa técnica permite ao titular usar a chave privada para declarar a autoria da mensagem. O destinatário usa a chave pública do assinante para verificar se a assinatura foi criada mediante o uso da correspondente chave privada e se a mensagem foi alterada depois da assinatura (Neto, 2017).

Entende-se que uma assinatura digital pode permitir um aumento de segurança em transações consideradas inseguras. Isto é, pode evitar que um elemento não autorizado possa fazer-se passar por outro (autorizado) numa troca de informação. Com a chave pública do signatário, o destinatário pode verificar se houve modificação de alguns ou de todos os bits da mensagem. A assinatura digital visa garantir:

- Autenticação da identidade da entidade que assinou a mensagem;
- Integridade (não alteração acidental ou maliciosa) da mensagem durante a sua transmissão;
- Não repúdio, isto é, o emissor não pode reclamar que não foi ele que assinou a mensagem.

1.3.2. Segurança criptográfica

Durante o processo de geração de uma chave, deve-se gerar uma sequência pseudoaleatória de bits de elevado tamanho. A sequência de bits deve ser imprevisível, isto é, dado um segmento inicial não deve ser possível prever a sua continuação. A

repetição de chaves, a previsibilidade e as chaves de tamanhos pequenos são sensíveis de serem quebradas por um atacante ativo.

O estudo e aplicação da Criptografia tem por objetivo, garantir as seguintes propriedades de segurança (Stallings, s. d):

a) **Confidencialidade**: garante que uma informação seja manipulada somente por usuários devidamente autorizados. A confidencialidade garante que mais ninguém teve acesso ao conteúdo da mensagem. Portanto, a confidencialidade, refere-se à impossibilidade de um adversário descobrir uma quantidade não desprezível de informação acerca da mensagem transmitida.
b) **Integridade:** garante que a informação processada ou transmitida chegue ao seu destino exatamente da mesma forma em que partiu da origem. Para se garantir a integridade, propriedade relacionada à precisão das informações, atestando a sua validade de acordo com os padrões e expectativas estabelecidas previamente. O destinatário da mensagem deverá possuir ferramentas para avaliar se a mensagem foi alterada ou não durante seu processo de transmissão, ou seja, atestar se o que chegou ao destino é idêntico ao que foi enviado na origem (não foi modificado).
c) **Autenticidade e não repúdio:** a autenticidade garante ao destinatário que a mensagem recebida foi realmente enviada pelo emissor (previsto). Na autenticação deve-se provar a real identidade do emissor, isto é, deve-se ter a certeza absoluta de que uma informação provém das fontes anunciadas, Ou seja, que o emissor da mensagem não é falso e que a mesma não foi modificada ao longo do processo de envio e receção. Para tal, deve-se autenticar o remetente. Assim, o seu destinatário consegue de maneira segura identificar e verificar que foi o mesmo (emissor esperado) quem enviou a mensagem, e ninguém mais. Portanto, o recetor da mensagem pode testar se uma mensagem foi intercetada e modicada, e neste caso, conclui se vai rejeitá-la, ou não.

   O não-repúdio impossibilita que o remetente de uma mensagem negue a autoria da assinatura. Em alguns ataques é possível que um intruso realize um ataque de personificação, fazendo-se passar, por exemplo, pelo emissor ou por outro agente. Pode-se impedir este fato, garantindo autenticidade da mensagem. Portanto, é possível garantir-se o não repúdio por meio de ferramentas como

certificados e assinatura digital, onde o emissor legalmente e tecnicamente não pode negar a autoria de uma mensagem assinada com uso de seu certificado, pois se garante que ele e somente ele fez a transmissão.

d) **Anonimato:** é a garantia de que nenhum agente é capaz associar a identidade do emissor de uma mensagem à respetiva mensagem. Isto é, não deve ser possível, por exemplo, associar a identidade de um eleitor com a intenção do seu voto, isto é, não pode ser possível identificar o autor do voto exercido à favor de um determinado partido ou candidato concorrente. Portanto, o anonimato é um dos requisitos de segurança de uma eleição eletrónica, pois em qualquer ato eleitoral os eleitores devem ser mantidos anónimos.

Portanto, o anonimato é um dos requisitos de votação eletrónica e não só, pois em qualquer sistema de votação os votantes devem ser mantidos anónimos.

1.4. Criptoanálise e tipos de ataques

Em geral, o objetivo de atacar um sistema de cifração é recuperar a chave em uso, em vez de simplesmente recuperar o texto claro a partir de um único texto cifrado. Um atacante à sistemas criptográficos é classificado como, passivo ou activo. Um atacante passivo é aquele que observa os dados, acompanha o processo, mas não afecta ou altera os dados originais da entidade legítima. Ao passo um atacante activo, é aquele que possui uma capacidade computacional de, para além de intersectar e observar as transações realizadas, tem a pode alterar e reintroduzir os dados no caminho original, de modo que, o emissor e o receptor não percebam tais alterações. Portanto, um atacante passivo, sua acção é limitada, recolhem informações de terceiros sem devida autorização, observando dados confidenciais de terceiros, a fim de tirar uma vantagem dos mesmos (Quaresma, 2012).

Criptoanálise: é a ciência que estuda as técnicas que visam tentar descobrir o texto limpo (informação secreta) ou a chave de cifra que permita ler um determinado criptograma (mensagem cifrada) (Pinho, 2007).

Nessa ciência, além de pessoas mal-intencionadas, outras que desejam conhecer as vulnerabilidades de uma cifra para que possam se proteger de ataques mais graves.

Os ataques criptoanalíticos utilizam-se da natureza do algoritmo, e talvez de mais algum conhecimento das características comuns ao texto claro, ou ainda de algumas amostras de pares de texto claro-texto cifrado. Esse tipo de ataque explora as características do algoritmo para tentar deduzir um texto claro específico ou a chave utilizada, isto é, pode explorar as repetições das palavras num criptograma, assim como também, a repetição. Todavia, seguem-se os principais tipos de ataques explorados por entidades não autorizado, de modo a inverterem as políticas de segurança de uma esquema criptográfico (Quaresma, 2012):

a) Ataque do tipo normal (ciphertext- only): o inimigo só tem acesso à parte ou todo do criptograma, ou mensagem cifrada. Todo sistema criptográfico deve resistir a pelo menos esse tipo de ataque. Uma técnica usual é o estudo da frequência de carateres, por exemplo, na língua portuguesa a frequência em que ocorre a letra "a" é maior que a frequência de "y". Com a comparação das frequências dos caracteres cifrados, sendo a mensagem realizada numa determinada língua, é possível inferir alguma informação do texto cifrado.

b) Ataque por força bruta: o atacante testa todas as chaves possíveis em um trecho do texto cifrado, até obter uma tradução inteligível para o texto claro. Na média, metade de todas as chaves possíveis precisam ser experimentadas para então se obter sucesso.

Se algum dos tipos de ataque tiver sucesso na dedução da chave, o efeito é catastrófico: todas as mensagens futuras e passadas, encriptadas com essa chave, ficam comprometidas.

O cenário mais difícil surge quando a única informação disponível é apenas *o texto cifrado*. Em alguns casos, nem sequer o algoritmo de cifração é conhecido, mas em geral podemos considerar que o oponente sabe qual é o algoritmo usado para a cifrar uma mensagem. Um ataque sob essas circunstâncias é a técnica de força bruta de testar todas as chaves possíveis. Se o espaço de chaves for muito grande, isso se torna impraticável.

O ataque apenas com texto cifrado é o mais fácil de ser defendido, pois o oponente tem a quantidade mínima de informação para trabalhar. Em muitos casos, porém, o

analista tem mais informações. Ele pode ser capaz de capturar uma ou mais mensagens de texto claro, além de suas encriptações. Ou então pode saber que certos padrões de texto claro aparecerão em uma mensagem.

De acordo (Nascimento, 2011: 12), ara que se possa discutir a respeito da segurança de sistemas criptográficos é necessário que, antes, seja especificado o que está disponível a um adversário, para que este tente quebrar a segurança do criptosistema (um ataque de nível teórico) ou do sistema (um ataque de ordem prática, isto é, por meio de manipulação e acesso ao sistema). A seguir estão os principais tipos de ataques considerados na literatura. É usual em criptografia assumir que o adversário possui uma descrição completa do sistema criptográfico em uso, a menos de sua chave secreta.

Técnicas mais específicas são usadas para atacar um sistema criptográfico, e estas podem ser:

d) Ataque de texto cifrado (Ciphertext- only attack): nesse ataque o adversário possui acesso somente a uma certa quantia de texto cifrado.

e) Ataque de texto em claro conhecido (Known-plaintext attack, KPA): o adversário possui acesso ao texto em claro de uma quantidade de dados, além do acesso ao texto cifrado.O inimigo tem conhecimento de alguns criptogramas e suas correspondentes mensagens originais, formados pela chave atual.

f) Ataque de texto em claro escolhido (Chosen-plaintext attack, CPA): o adversário pode escolher quais dados e seus respectivos textos cifrados e ter acesso aos mesmos. É caracterizado pelo facto do inimigo ser capaz de submeter qualquer mensagem à criptografia e receber o criptograma correto com a chave atual utilizada.

O objetivo desse ataque é obter alguma informação que reduza a segurança do esquema criptográfico, e na pior das hipóteses, revelar o esquema de produção da chave. Esse tipo de ataque tem maior importância quando se trata de criptografia assimétrica, onde a chave é pública e um inimigo pode encriptar qualquer mensagem de seu interesse.

g) Ataque do tipo informação cifrada escolhida (chosen- ciphertext): é um tipo de ataque no qual o inimigo pode escolher um criptograma arbitrário e obter o resultado correto para sua decriptação. Alguns algoritmos famosos na sua versão mais antiga eram suscetíveis a este ataque, como o RSA.

1.5. Eleição electrónica

Uma eleição eletrónica não é simplesmente uma troca de ferramentas e materiais. Não significa passar de urna de madeira, plástica, ou de metal para uma urna eletrónica. É muito mais que as possibilidades/ funcionalidades que o novo sistema oferece, pois permite redesenhar, corrigindo o sistema eleitoral. Um sistema de eleição eletrónica é qualquer sistema de eleição eletrónica que utilize meios eletrónicos nas fases de votação, ou contagens dos votos. O processo de eleição eletrónica tem o objetivo de automatizar os diferentes processos eleitorais com intuito de se obter resultados ecientes, isto é, resultados justos e verdadeiros. Os principais tipos de eleição eletrónica são (Prince, 2004):

a) Eleição eletrónica remota: o eleitor manifesta o seu voto através da internet mediante o uso de um meio tecnológico (com acesso à internet).

b) Eleição eletrónica presencial: este tipo de eleição implica o uso de captação eletrónica do voto com transmissão e escrutínio provisório através de uma urna eletrónica colocada nos lugares (físicos) onde se realizam a eleição.

1.5.1. Estilos de um voto electrónico

Uma eleição pode ocorrer com diferentes propósitos, tendo como consequência as formas distintas de apresentar o voto. Ou seja, o número de opções ou candidatos possíveis a serem apresentados ao eleitor para que este possa realizar suas escolhas. Estas formas de apresentação do voto são denominadas de estilos de voto, e são descritas a seguir (Mursi, 2013):

a) Uma escolha de 2 opções de voto (sim/não): a resposta do eleitor é sim ou não. Cada voto emitido representa: 1 para sim e 0 para não.

b) Uma escolha de n opções de eleição: a partir de N opções o eleitor realiza uma escolha que é representada como 1, e as demais como 0.

c) k escolhas de n opções de eleição: o eleitor realiza diferentes k escolhas de um conjunto de n possibilidades. A ordem dos elementos selecionados não é importante.

d) k escolhas de n opções ordenadas: o eleitor põe em ordem diferentes k escolhas de um conjunto de n possibilidades. Neste caso, a ordem é importante.

e) Voto escrito em boletins: o eleitor formula sua própria resposta e escreve o voto. Votar é uma sequência de letras com um tamanho máximo, que representa o nome de uma pessoa, por exemplo.

1.5.2. Intervenientes numa eleição electrónica

Baseando-se nas eleições tradicionais, os atores do sistema de eleição eletrónica são(Mursi, 2013):

a) Comissão eleitoral: é responsável por todo processo eleitoral, incluindo o recenseamento eleitoral, a gestão do sistema de voto e autenticação da informação publicada;

b) Auditores: correspondem, em parte, à gura dos delegados de listas nas eleições tradicionais, e são responsáveis pelo controlo no que respeita a privacidade do eleitor e integridade da eleição.

c) Entidades de Verificação: são responsáveis por, de forma independente, verificar a validade e correção da eleição. Verificam também os resultados eleitorais. São em geral, entidades diretamente interessadas nos resultados da eleição (partidos políticos).

d) Eleitor: Corresponde a qualquer cidadão com direito de voto, e cada eleitor terá de se recensear uma única vez juntamente da comissão eleitoral, sendo necessário registar-se para cada eleição antes do processo de eleição.

e) Adversário: um adversário é uma entidade maliciosa, que tenta manipular o processo eleitoral, sua apuração ou o eleitor. O adversário pode atuar em um momento isolado ou em vários momentos de uma eleição. É possível dividir sua atuação em dois tipos:

- **Adversário externo:** fazem parte deste grupo, entidades sem intervenção direta no processo eleitoral, capazes de realizar uma ação maliciosa; isto é, não elegem,

organizam o processo, nem distribuem equipamentos eletrónicos. Exemplo deste tipo de adversário, pode ser qualquer pessoa com capacidade técnica de promover ataques.

- **Adversário interno:** refere-se a entidades com intervenção direta no processo eleitoral, capazes de realizar uma ação maliciosa. Por exemplo: pode ser um eleitor que queira expandir seu único voto para contabilizar 1.000 ou mais; um distribuidor de equipamento de hardware ou software com alguma vulnerabilidade que possa ser explorada num ataque futuro; ou uma autoridade corrupta.

1.5.3. Princípios gerais numa eleição electrónica

Os princípios a se ter em conta numa eleição eletrónica são os seguintes (Preya, 2015):

1) Princípio do isomorfismo ao processo tradicional: deve-se preservar o direito de participação; desenho da legislação eleitoral; tecnologia para a eleição deve ser acessível para todos; a eleição eletrónica deve ser vista como um meio alternativo; existência de infraestrutura pública adequada (internet gratuita e outros).

2) Princípio da elegibilidade do voto: enfatiza que os eleitores devem ser registados e autenticados para emitir o voto.

3) Princípio incoercibilidade: deve-se garantir que o voto não possa ser comprado, nem o eleitor coagido fora do sistema.

4) Princípio da liberdade de decisão: nenhuma propaganda política deve circular no local da eleição durante o acto eleitoral.

5) Princípio da opção do voto inválido: este princípio defende que deve ser dado ao eleitor o direito de optar pelo voto nulo.

6) Princípios da igualdade entre candidatos: a interface do sistema não deve prejudicar ou favorecer candidato algum. A exibição dos candidatos na tela do computador deve prover equidade. Por outra, todos candidatos devem ter acesso às mesmas ferramentas e informações para verificar e auditar o processo.

7) Princípio de igualdade entre eleitores: enfatiza que todos votos devem ter o mesmo valor e um eleitor deve votar uma única vez.

8) Princípio do sigilo: defende que uma vez lançado o voto, o processo deve ser irreversível, nem o próprio eleitor deve ser capaz de recuperar sua decisão; garantir o sigilo do voto, desde a sua emissão, transição, recepção e contagem; nenhum dos membros da comissão deve ser capaz de associar um voto a um eleitor.

9) Princípio da transparência: defende que todos os agentes envolvidos devem ser capazes de entender como o processo ocorre.

10) Princípio da verificabilidade e prestação de contas: o processo deve ser verificável, auditável sempre que necessário.

11) Princípio da confiabilidade e segurança: defende a existência da certificação dos hardwares e softwares; toda infraestrutura e funcionalidade do sistema devem ser verificáveis.

1.6. Sistemas de eleição electrónica

Lichtler (2004) no seu artigo científico intitulado: "*Um sistema seguro para votações digitais*", apresenta 7 protocolos criptográficos de segurança numa eleição electrónica:

1) Protocolo sem central
2) Voto cifrado à central de eleição
3) Voto assinado e cifrado à central de eleição
4) Voto com assinatura cega
5) Protocolos de duas centrais de voto sem assinatura
6) Protocolos de duas centrais de voto com assinatura
7) Protocolo de três canais

Um protocolo criptográfico (segurança), é a combinação de técnicas criptográficas com propósito de garantir de manter a privacidade de uma informação que circula através de um canal inseguro (internet), em todas suas fases.

- **Protocolo sem central**

Este, é um protocolo implementado sem o uso de uma central eleitoral. O faz com que cada voto passe por cada eleitor, duas vezes. O protocolo funciona da seguinte forma:

Cada eleitor manifesta o seu direito de voto: anexa um número aleatório ao seu voto, cifra e assina o seu voto, de seguida o processo de decifração e verificação da assinatura é feito entre os eleitores. Os eleitores conseguem verificar se o seu voto é autêntico, através dos números aleatórios gerados.

O protocolo apresenta alguns problemas, a responsabilidade pela correção do processo recai sobre todos os eleitores. Com agravante de que um eleitor é capaz de adulterar o voto.

- **Voto cifrado à central de eleição**

Neste tipo existe uma comissão eleitoral, faz uso da criptografia assimétrica, que faz uso de um par de chaves, uma pública e outra privada:

a) O responsável pela Comissão eleitoral, gera um par de chave;
b) O eleitor cifra o seu voto com a chave pública do responsável pela comissão eleitoral e envia-lhe o seu voto;
c) O responsável pela comissão eleitoral decifra os votos com a sua chave privada, realiza a contagem e publica os resultados

Este protocolo é inseguro, pois preserva apenas a integridade e a confidencialidade. Pois, somente o responsável pelo escrutínio está habilitado em ter acesso ao conteúdo do voto. A autenticidade do voto e o anonimato do eleitor.

- **Voto assinado e cifrado à central de eleição**

O responsável da Comissão eleitoral, detém da chave pública dos eleitores e os eleitores também possuem a chave pública do responsável da comissão eleitoral:

a) Tanto o responsável da Comissão eleitoral, quanto os eleitores, geram um par de chaves.

b) Cada eleitor assina o seu voto com a sua chave privada, cifra com a chave pública do responsável da Comissão eleitoral e envia.

c) O responsável com a sua chave privada decifra cada voto e verifica a autenticidade com a chave pública de cada eleitor.

d) Finalmente faz-se a contagem e a publicação dos resultados.

Este protocolo garante a integridade, confidencialidade e a autenticidade do voto, mas não garante a o anonimato do eleitor.

- **Voto com assinatura cega**

Assinatura cega significa que pode assinar um documento, sem ter acesso ao seu conteúdo. São intervenientes no processo, o eleitor, responsável da comissão eleitoral e um agente de confiança. O protocolo funciona da seguinte forma:

a) O eleitor exerce o direito de voto, escolhe um número secreto e ofusca (cega) o seu voto e envia-o para o agente de confiança;
b) O agente de confiança com a sua chave privada RSA assina o voto (cegado) e devolve para o eleitor;
c) O eleitor desofusca e obtém o voto assinado, de seguida envia para o responsável da Comissão eleitoral;
d) Com a chave privada do agente de segurança, verifica a autenticidade do voto.

O protocolo garante a o anonimato do eleitor e a autenticidade do voto, com a integração da cifração/ decifração, a integridade e a confidencialidade são garantidas.

- **Protocolos de duas centrais de voto sem assinatura**

As duas centrais são, Central de validação (CV) e Central de conferência (CC), respectivamente. O processo de validação do cadastro dos eleitores, escrutínio e publicação, é partilhado entre as duas centrais:

a) A CV gera e distribui para cada eleitor um número secreto;
b) Envia para a CC a lista de todos os números secretos, sem a correspondência dos eleitores;
c) Cada eleitor gera uma chave secreta, cifra o seu voto e número secreto, envia o criptograma e partilha a chave secreta (simétrica) para a CC.
d) Tendo a chave simétrica partilhada, decifra o voto e confirma a autenticidade do voto, comparando o número secreto do eleitor recebido com o que ele na lista dos números secretos recebidos anteriormente.

Portanto, este protocolo não é recomendável, por apresentar várias fragilidades: a chave simétrica (partilhada) utilizada para cifração pode ser intersectada por um agente malicioso e todo sistema de segurança estaria comprometido.

- **Protocolos de duas centrais de voto com assinatura**

Nesta outra versão de protocolo de duas centrais, a CC detém as chaves públicas dos eleitores, CV disponível a sua chave pública aos eleitores:

a) Cada eleitor envia a sua mensagem assinada para a CC, solicitando um número de validação;
b) Com a chave pública do eleitor, a CC verifica a assinatura e envia o número de validação cifrado com a chave pública do eleitor. A CC possui a associação dos números de validação com as correspondentes identidades dos eleitores, para evitar que uma pessoa vote mais de uma vez;
c) A CC envia a lista de números de validação para CV, devidamente assinada e cifrada;
d) Cada eleitor escolhe um número aleatório de identificação, com a chave pública de CV cifra juntamente o número de identificação, de validação, o voto, e envia para a CV;
e) A CV decifra o recebido, compara o número de validação recebido com o que está na lista, regista o número de identificação do eleitor, faz a conta o voto;
f) Finalmente a CV publica os resultados, alistando cada voto com o seu número de identificação gerado pelo respectivo eleitor.

Com este protocolo, o eleitor é capaz de verificar se o seu voto foi devidamente processado, com a associação do seu voto com o número de identificação apenas. A CV não é capaz de associar a identidade dos eleitores com os números de validação, pois só lhe é enviado a lista com os números de validação, desta forma, não é possível associar os votos á identidades e assim, não consegue reconhecer o eleitor que gerou um dado voto através dos resultados.

A unicidade do voto é garantida, visto que o eleitor só recebe o número de validação depois de ter a sua assinatura reconhecida pela CC. Como o eleitor permanece anónimo e cada voto é apresentado no final com o seu número de identificação, a CV não consegue

falsificar os votos, por outra, também não consegue inventar os votos, pois a CC sabe o número total de eleitores aptos para o processo e está habilitado a detectar falsificação.

A segurança do protocolo está intriscicamente com a honestidade das duas centrais (CV e CC), por isso são considerados agentes de confiança.

Protocolo de três canais

Possui 3 centrais de confiança, de votação (CEVO), validação (CEVA) e de contagem (CECO). Subdivide-se em 2 variantes.

a) **Primeira:** a CEV cifra o voto do eleitor com uma chave de cifração, de seguida gera um número aleatório de grande tamanho, que será usado como factor de ofuscação do voto cifrado, utilizando a chave pública do eleitor, assina o voto deste, e envia para CEVA o voto cegado e cifrado, juntamente com a assinatura e o ID do eleitor. ACEVA recebe, decifra e verifica e assinatura. A seguir a CEVA assina, cifra o voto ofuscado e envia para CEVO, este decifra, obtendo o voto ofuscado, assinado pela CEVA, retira a ofuscação e dessa forma obtém a assinatura do voto não ofuscado, verifica a autenticidade.

Feita a interação entre os dois, a CEVO cifra o voto, a CEVA assina, e a CEVO envia o par para a CECO, este por sua vez, decifra e verifica a assinatura. CECO assina o voto cifrado e actualiza a lista de recibos, envia o recibo, o voto assinado e cifrado de volta para CEVO, CEVO verifica a assinatura de CECO, e envia de volta o número de recibos em conjunto com a chave de decifragem. O recibo também é enviado para o eleitor. No final a CECO, decifra o voto e actualiza os resultados.

Este protocolo garante todos os requisitos de uma eleição electrónica segura. Pois, o eleitor. Também aqui a segurança do sistema está pela honestidade da CEVO, visto que, ela pode assinar um voto pelo eleitor, o que pode violar o princípio da unicidade do voto.

b) **Segunda:** Diferentemente da primeira, aqui o eleitor interage com as 3 agentes, nomeadamente: Central de Alistamento (CA), de votação (CV) e escrutínio (CE):

Cada eleitor autentica-se perante CV, a CV produz uma cédula em branco e envia para a CE, a CE assina a cédula e envia de volta à CV, a CV envia a cédula assinada para o eleitor;

O eleitor verifica a assinatura e a retira, obtendo a cédula original em branco, o eleitor assina a cédula em branco, emite o seu voto ofusca, cifra-o e envia para CE, juntamente com o factor de ofuscação. O eleitor envia para CV, o pacote assinado e cifrado contendo a cédula em branco (assinada), voto (cegado) e seu número identificador. A CV decifra o pacote e repassa para a CE. A CE verifica a assinatura do pacote e assina o voto cegado e envia-o de volta para a CV, e este repassa para o eleitor, o eleitor retira o factor de ofuscação e obtém o voto assinado, finalmente envio o voto para o escrutínio.

Portanto, apesar do protocolo satisfazer os requisitos de segurança, é manchado pelo elevado nível de complexidade.

2. METODOLOGIA

Neste capítulo, abordou-se a metodologia seguida durante a investigação, fundamentalmente a definição do tipo da investigação, determinação da população e amostra e do método de adoptado na investigação.

2.1. Tipo de investigação

O tipo da investigação é bibliográfica. Pois, procurou-se estudar o problema recorrendo às principais fontes bibliográficas de autores que abordaram o mesmo assunto, onde analisou-se as principais contribuições e debilidades dos sistemas propostos em suas obras, que permitiu construir um sistema que, de certa forma, superou as insuficiências das propostas a que tivemos acesso, pelo facto de alguns terem aplicado técnicas criptográficas com vulnerabilidades susceptíveis de invasão.

2.2. População e amostra

A população da investigação foi constituída por todos cidadãos com idade eleitoral (maiores de 18 anos de idade) da província de Namibe, cerca de 320.800 elementos (INE, 2016).

Tendo em conta que a investigação é bibliográfica tomou-se como amostra (não probabilística) 15 obras bibliográficas, cujas teorias foi alvo de análise e interpretação, a fim de construir teorias que melhore as conclusões apresentadas pelos autores.

2.3. Métodos da investigação

Com o método indutivo, visto que, partiu-se das conclusões dos autores cujas abordagens (particulares) permitiram chegar a conclusões genéricas.

Foi adoptada a observação como técnica de recolha de dados, observação assa que consistiu na análise das propostas já existentes, a análise acima referenciada consistiu na apresentação de pontos fortes e pontos a melhorar. O que permitiu a formulação de uma proposta mais consistente.

Depois da fase da observação, a construção do sistema proposto, realizaram-se as seguintes acções:

1) Selecção das técnicas criptográficas recomendáveis universalmente e com elevado nível de robustez;
2) Estudo dos algoritmos de cada técnica criptográfica para permitir uma combinação significativa com os algoritmos de outras técnicas;
3) Esboço do esquema de funcionamento do sistema eleitoral proposto;
4) Determinação de possíveis ataques contra o sistema;
5) Construção do algoritmo que implemente o sistema de segurança resistente contra ataques;
6) Construção do algoritmo geral que fundamente o funcionamento do algoritmo do sistema;
7) Execussão do sistema;
8) Recolha dos resultados;
9) Interpretação dos resultados;
10) Formulação das conclusões.

Portanto, os algoritmos foram implementados na linguagem de programação python. Todavia realçar o mesmo é flexível em outras linguagens de programação, tais como: Javascript, Php, C#, C++, entre outras.

3. RESULTADOS

3.1. Sistema de eleição proposto

O presente protocolo criptográfico, define a tecnologia que visa a construção do sistema de eleição electrónica proposto neste trabalho científico, o mesmo é resultante da combinação de técnicas criptográficas que visam garantir a confidencialidade, integridade, autenticidade e não repúdio, anonimato e verificabilidade numa eleição eletrónica.

Abordou-se neste capítulo, os intervenientes do sistema, o algoritmo do sistema proposto, onde se procurou-se detalhar cada passo do algoritmo através de exemplos práticos, e para garantir a funcionalidade computacional do mesmo, implementou-se computacionalmente, recorrendo à linguagem de programação “python”, que depois da execussão, produziu resultados esperados. Isto é, foi possível averiguar o processamento do voto, a garantia da autenticidade, verificabilidade e outras propriedades de segurança de uma informação que circula através da internet. Portanto, no final do capítulo,

estudou-se os aspectos de segurança do esquema, o que fundamenta a consistência do funcionalidade do sistema de eleição electrónica em estudo.

3.1.1. Intervenientes do esquema

São intervenientes do esquema as seguintes entidades:

a) Eleitor (El): tem a função de eleger, ofuscar, desofuscar e cifrar o voto (mensagem).

b) Membros da comissão eleitoral (MCE:) participam na partilha da chave privada e verificam a consistência das partes secretas recebidas.

c) Autoridade de confiança (AC): tem a função de gerar um par de chaves RSA e assinar o voto e provar que os votos foram corretamente formados.

d) Autoridade da comissão eleitoral (ACE): tem a função de escolher o par de chaves Elgamal e partilhar a chave privada, reconstruir a chave privada, decifrar os votos, verificar a autenticidade e verificar se os votos foram corretamente formados.

3.2. Técnicas criptográficas envolvidas

- **Elgamal:** serviu para cifrar / decifrar o voto, garantindo a confidencialidade.
- **Assinatura cega:** aplicada para garantir a autenticidade do voto, não repúdio e o anonimato do eleitor.
- **Partilha da chave secreta:** técnica aplicada para garantir um armazenamento seguro da chave secreta, evitando assim que, intrusos tenham acesso à chave privada.
- **Esquema de verificação de Feldman:** habilitou os agentes participantes da partilha verificarem as porções da chave privada recebidas, no momento da partilha.
- **Prova de conhecimento zero:** técnica utilizada para provar à entidade provadora provar ao verificador (ACE) que os votos foram correctamente formados.

3.2.1. Algoritmo da do sistema

Sejam, El, ACE, AC e MCE, segue-se o algoritmo:

I. Inicialização

1) Escolhe-se um p primo, tal que $\mathbb{Z}_p{}^* = \{1;\ 2; \dots; p-1\}$ (conjunto dos possíveis restos da divisão de inteiros positivos excepto zero, por p) um grupo com gerador $g > 1 \in \mathbb{Z}$, isto é, g é qualquer número, tal que, cada elemento de $\mathbb{Z}_p{}^*$, pode ser escrito na forma $1 \equiv g^n \ mod\ p, 2 \equiv g^n \ mod\ p, \dots, (p-1) \equiv g^\alpha \ mod\ p$ onde $\alpha \in \mathbb{N}_0$. ACE escolhe um par de chaves Elgamal: $x \in \mathbb{Z}_p{}^*$- chave privada e (x, g, g^x)- chave pública.

2) Em seguida ACE define n- total de participantes na partilha da chave privada e k- total de elementos necessários para reconstruir a chave privada, com a chave privada inicia o processo de partilha, construindo o polinómio:

a) Escolhe-se $(k-1)$ $a_i \in \mathbb{Z}_p{}^*$, coeficientes do polinómio, constrói-se e fixa-se o polinómio,

$$P_1(x) = \sum_{i=1}^{(k-1)} a_i x^i = a_{(k-1)} x^{(k-1)} + \cdots + a_1 x + a_0$$

Onde $i = 1, 2, \dots, k-1$ e a_0 é a "chave privada".

b) Calcula-se $P(t)$, onde $t = 1, 2, \dots, n$ (igual ao total de participantes na partilha), os pares ordenados $(t_j;\ P(t)_j)$ são as partes a serem partilhadas entre os n elementos, onde $j = 0, 1, \dots, n$.

3) Com os coeficientes de $P_1(x)$, a ACE calcula $c_i = g^{a_i} \ mod\ p$ $(i \in \mathbb{N})$, onde a_i são os coeficientes e g o gerador de $\mathbb{Z}_p{}^*$, envia $(t_j;\ P(t)_j)$ e c_i aos MCE.

4) Cada MCE verifica a consistência das partes recebidas, fazendo:

$$g^{P(t)_i} = c_0^{t^0} . c_1^t . \dots . c_{(k-1)}^{(k-1)}$$

II. Operacionalização

5) Depois do eleitor votar M, inicia-se o tratamento voto M: para garantir a sua autenticidade, efectua-se a assinatura cega. A AC gera um par de chave RSA, isto é, gera o par de chaves:

d- Chave privada;

$(n,\ e)$- Chave pública, onde n é o módulo RSA.

O eleitor escolhe um inteiro $u \in \mathbb{Z}_p^*$ e com a chave pública de AC ofusca o voto, de modo que posso ser assinado sem que a AC tenha acesso ao conteúdo voto:

$$M_1 = u^e \ .\ M\ (mod\ n)$$

Onde, e é a chave pública RSA da AC.

- Recebendo M_1, com a sua chave privada assina-o, sem porém saber o conteúdo do voto, obtendo o voto assinado:

$M_2 = {M_1}^d = (u^e\ .M\ )^d = u^{e\ .\ d}\ .M^d = u\ .\ M^d\ (mod\ n)$, onde $e\ .\ d = 1$. Pois, d é o inverso multiplicativo de e.

O eleitor, desofusca o voto e obtém a assinatura, multiplicando M_2 por u^{-1}, apercebendo-se assim que, o voto assinado o pertence.

$$M_3 = M^d\ (mod\ n)$$

Com o voto já assinado, o eleitor usando a chave pública de ACE, cifra o seu voto.

6) De modo a dar início ao processo de escrutínio e verificação do voto, a ACE reconstrói o polinómio $P_2(x)$, a fim de recuperar a chave privada.

A reconstrução do polinómio é feita mediante o cálculo do polinómio interpolador de Lagrange:

$$P_2(x) = \sum_{j=0}^{n} y_j . l_j(x) = y_0 . l_0(x) + y_1 . l_1(x) + \cdots + y_n . l_n(x)$$

Onde $y_j = P(t)_j$ e,

$$l_j(x) = \prod_{(i=0, i \neq j)}^{n} \frac{(x - x_i)}{(x_j - x_i)}$$

Por exemplo, sejam, $(x_i;\ y_j)$, o polinómio $P_2(x)$, para $n = 2$ seria:

$$P_2(x) = y_0 . \frac{(x - x_1)}{(x_0 - x_1)} . \frac{(x - x_2)}{(x_0 - x_2)} + y_1 . \frac{(x - x_0)}{(x_1 - x_0)} . \frac{(x - x_2)}{(x_1 - x_2)} + y_2 . \frac{(x - x_0)}{(x_2 - x_0)} . \frac{(x - x_1)}{(x_2 - x_1)}$$

Reconstruida a chave, dá-se início à decifração dos votos. Com a chave pública da AC verifica a autenticidade do voto: se a potência de base M_3 (assinatura digital) e expoente e (chave pública RSA) for igual ao voto, então conclui-se que o mesmo é autêntico, retornando True (**autêntico**) ou False (**não autêntico**).

III. Confirmação do processo

7) Para além do facto dos votos serem enviados cifrados e assinados, garantindo assim, a integridade, confidencialidade e autenticidade, para provar que não houve anomalia no processo, os votos são enviados juntamente com uma prova designada, prova de conhecimento zero, que consiste em provar ao destinatário que o processo foi realizado correctamente, sem revela-lo a informação inerente ao voto.

Sejam p e q, dois números primos, o provador adopta um segredo. O **provador** escolhe números secretos s_i calcula os quadrados modulares $v_i$de base s_i $(i = 1, 2, \dots, n)$. Escolhe os inteiros $r \in \mathbb{Z}^*_n$ e calcula $x = r^2 \ mod\ n$.

O **verificador** escolhe números binários $w_i = \{0, 1\}$, envia ao provador e ele calcula $y = r\ .\ {s_1}^{w_1} \dots {s_i}^{w_i}\ mod\ n$.

O provador efectiva a prova, enviando ao verificador o seguinte: v_i, r, y, onde:

$$v_i = {s_i}^2 \ mod \ n$$

Por fim, o verificador conclui, comparando:

$$y^2 = r^2 \ . ({s_1}^{w_1} . \ldots . {s_i}^{w_i})^2 \ mod \ n$$

$$y^2 = x \ . {s_1}^{2w_1} . \ldots . {s_i}^{2w_i} \ mod \ n$$

$$y^2 = x \ . {v_1}^{w_1} . \ldots . {v_i}^{w_i} \ mod \ n$$

Retornando True (**confirmado**) ou False (**Não confirmado**).

3.3. Implementação computacional da proposta

O algoritmo foi implementado na linguagem de programação python, com propósito de garantir a aplicação computacional da abordagem feita na proposta.

Os resultados são apresentados por fases conforme as escolhas dos intervenientes ao processo e o sistema de eleição produz os seguintes resultados possíveis:

```
Quantos participantes tera o esquema de partilha da CHAVE PRIVADA?10
Quantas partes partes secretas serao necessarias para RECONSTRUIR A CHAVE?5
Quantos concorrentes estao inscritos?7
Digite um numero para votar:5
('O(a) senhor(a) votou no candidato/ partido numero:', 5)
Um Voto válido.
('O voto assinada e:', 13474043)
('A chave privada COMPARTILHADA e:', 3352)
('O polinomio e:', 22193*x^4 + 26419*x^3 + 29135*x^2 + 640*x + 3352)
('As partes secretas a compartilhar sao:', [[1, 81739], [2, 687612], [3, 2778433], [4, 7844296], [5, 1
7907927], [6, 35524684], [7, 63782557], [8, 106302168], [9, 167236771], [10, 251272252]])
('As potencias para a verificacao de consistencia sao:', [20235, 24992, 5283, 30319, 30671])
('A verificacao da parte secreta distribuida ao PARTICIPANTE', 2, 'e:', True)
('A verificacao da parte secreta distribuida ao PARTICIPANTE', 2, 'e:', False)
(5, 'partes reconstruiram:', 22193*x^4 + 26419*x^3 + 29135*x^2 + 640*x + 3352)
(4, 'partes reconstruiram', 248349*x^3 - 747620*x^2 + 1110290*x - 529280)
('O voto cifrado e:', [16807, 9858])
('O voto decifrado e:', 5)
('A autencidade do voto e:', True)
('A autencidade do voto e:', False)
('O verificador escolheu:', [1, 0, 0])
('O verificador fica convencido, e diz:', True)
(' O verificador nao correspondido:', [2, 1, 1])
(' O provador nao o convence, e ele diz:', False)
```

Fig. 1. Possíveis resultados da execução computacional do sistema de eleição electrónica proposto

Portanto, assim como em qualquer software, nem todas operações (interna) do código são mostradas ao usuário, neste também, o usuário receberá as respostas de acordo as instruções que forem dadas.

3.4. Interpretação dos resultados

Analisemos os resultados que o sistema produz nas três fases da construção e funcionamento.

I. Inicialização

Como se vê, o sistema na fase da inicialização realiza:

1) A ACE instrui de antemão a definição do número de MCE participantes na partilha e recuperação da chave privada utilizada para a decifração dos votos; Também define o número total de candidatos concorrentes.
2) De seguida constrói o polinómio, a chave privada é o termo independente do polinómio e para garantir um armazenamento seguro da chave, executa a partilha da chave usada para decifrar os votos e disponibiliza as potências (5 números) a eles, a fim permiti-los verificar a consistência das partes recebidas, que pode resultar em True ou False. Os resultados desta fase, são:

```
    Quantos participantes tera o esquema de partilha da CHAVE PRIVADA?10
    Quantas partes partes secretas serao necessarias para RECONSTRUIR A CHAVE?5
    Quantos concorrentes estao inscritos?7
('O polinomio e:', 22193*x^4 + 26419*x^3 + 29135*x^2 + 640*x + 3352)
('As partes secretas a compartilhar sao:', [[1, 81739], [2, 687612], [3, 2778433], [4, 7844296], [5, 1
7907927], [6, 35524684], [7, 63782557], [8, 106302168], [9, 167236771], [10, 251272252]])
('As potencias para a verificacao de consistencia sao:', [20235, 24992, 5283, 30319, 30671])
('A verificacao da parte secreta distribuida ao PARTICIPANTE', 2, 'e:', True)
('A verificacao da parte secreta distribuida ao PARTICIPANTE', 2, 'e:', False)
```

Fig. 2. Resultados da fase da inicialização

A parte do código invocada na geração desses resultados é:

```
def Elgamalinicial (bits):

    p=random_prime(2**bits)
    Zp=IntegerModRing(p)
    g=Zp.multiplicative_generator()
    n=input("Quantos participantes tera o esquema de partilha da CHAVE PRIVADA?")
    while n==1:
        n=input("Impossivel partilhar a chave. Insira um numero de participantes maior que 1
    k=input("Quantas partes partes secretas serao necessarias para RECONSTRUIR A CHAVE?")
    while k<=1 or k>n:
        k=input("Insira outro valor de k, menor ou igual que n para RECONSTRUIR A CHAVE?")
    a0= ceil(randint(2, p-1))
    y=g**a0
    PuKey=(p, g, y)
    PrKey=a0
    return PuKey, PrKey, n, k, Zp, g
```

Fig.3. Geração do par de chave Elgamal

Aplicou-se a técnica criptográfica Elgamal que retorna o para de chave usado para cifrar ou decifrar os votos, cuja função Elgamal recebe o número de bits ou o tamanho da chave. Também define-se o número de elementos necessários para a partilha da chave de decifração, se haver apenas 1, o sistema recusa e informa que necessita de pelo menos 2

elementos. Do mesmo jeito, define-se o de elementos necessários para a recuperação da chave partilhada e, se for definido 1 ou um número maior que o total de participantes, o sistema está preparado para rejeitar, pois o sigilo da chave estaria comprometido. Nesta fase o sistema solicita à ACE, que defina o número total de concorrentes a serem eleitos e avançar para a fase de votação (operacionalização). Se o eleitor cometer algum erro ao inserir o do seu candidato, o sistema solicita que ele tente novamente até que insira um outro número.

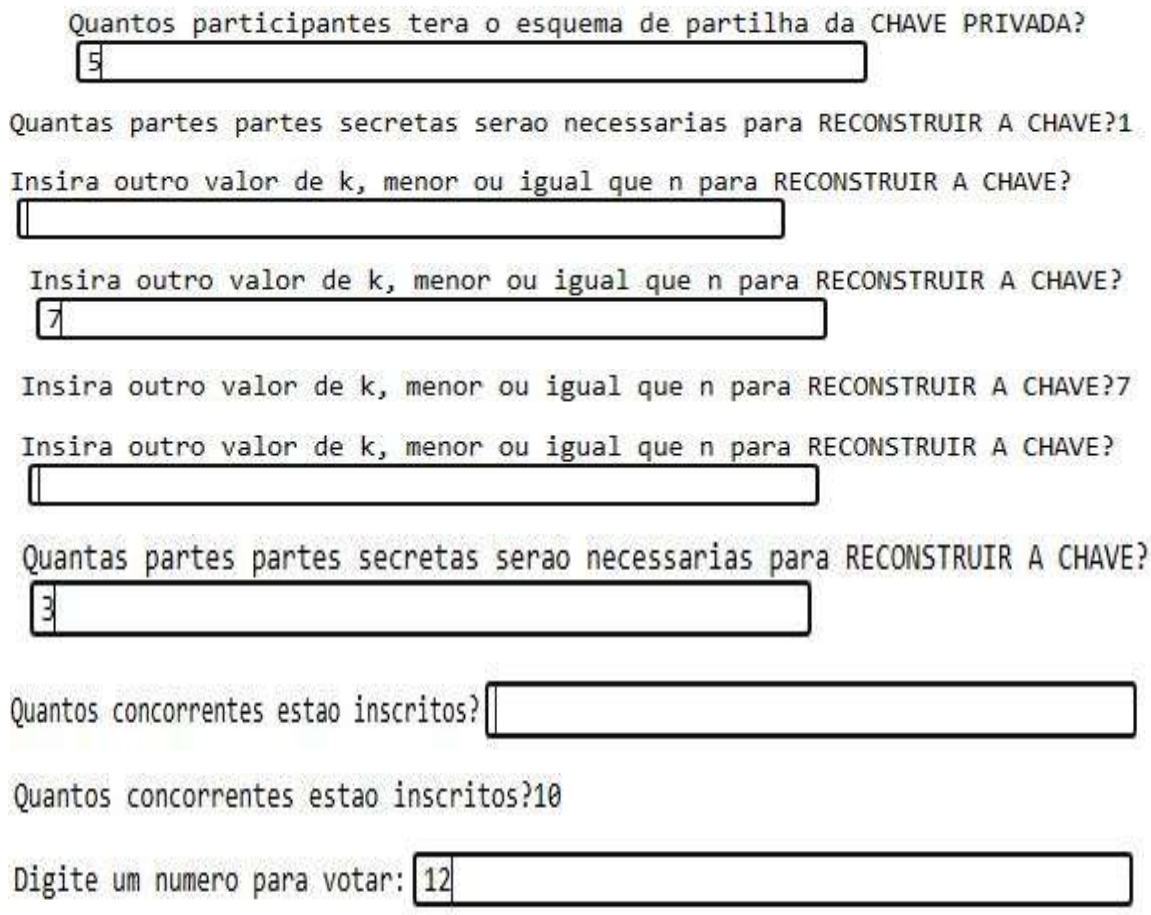

Fig.4. Consistência do sistema contra erros na fase de inicialização

Antes do processo da partilha, faz a construção do polinómio, começando por gerar os coeficientes do polinómio, invocando a função "*>>> Coefpolinomio"*, que recebe o grupo, chave privada, total de participantes na partilha e os necessários para recuperar a chave. a função *"*>>> *polinomioconstruido"* recebe os coeficientes e retorna o polinómio, tal que, o termo independente é a chave privada. A função *"*>>> *partespartilhadas",* com parâmetros de entrada, o total de elementos participantes na partilha e o polinómio, retorna as partes a partilhar.

```
def Coefpolinomio (Zp, Prkey, n, k):

    a0=Prkey
    #Zp= IntegerModRing(p)
    t=k-1
    coef = [a0]
    for r in range (1, t+1):
        a = Zp.random_element()
        coef.append(a)
    return coef

# Construção do polinómio BOB
def polinomioconstruido (coef):

    Zp=coef[0].parent()
    Pol.<x> = PolynomialRing(Zp)
    polinomio=Pol(coef)
    return polinomio

# Partes à partilhar BOB
def partespartilhadas (n, polinomio):

    pares=[]
    for x in range (1, n+1):
        pares.append([x, polinomio(x)])
    return pares
```

```
('A chave privada COMPARTILHADA e:', 7631)

('O polinomio e:', 47734*x + 7631)
('As partes secretas a compartilhar sao:', [[1, 55365], [2, 103099], [3, 150833]])
```

Fig. 5. Construção do polinómio e partilha da chave

Pode-se ver nos resultados que, a chave privada 7631, que é o termo independente do polinómio construído, que deu origem os pares ordenados (partes secretas partilhadas entre 3 pessoas).

Os participantes da partilha estão habilitados a verificar se a parte secreta recebida é consistente ou não, isto é, se permite reconstruir a chave privada ou não. Com os coeficientes do polinómio, calculam as potências e com as potências testam as partes secretas, se são verdadeiras ou não, com a função *">>> verificapartes"*, que recebe o número de ordem do participante, o gerador do grupo e a parte secreta recebida e resulta em:

```
i=0
pot=[]
def potencia (g, coef):

    pot=[]
    k=len(coef)
    for i in range (k):
        pot.append(g**coef[i])
    return pot

# Verificação da consistência das partes secretas- MEC

def verificacaopartes (i, g, pot, parte_secreta):

    prod=1
    k=len(pot)
    for j in range (k):
        prod =prod*(pot[j]**(i**j))
    if g**parte_secreta == prod:
        return True
    else:
        return False
```

```
('As partes secretas a compartilhar sao:', [[1, 55365], [2, 103099], [3, 150833]])

('As potencias para a verificacao de consistencia sao:', [18340, 14063])
('A verificacao da parte secreta distribuida ao PARTICIPANTE', 2, 'e:', True)
('A verificacao da parte secreta distribuida ao PARTICIPANTE', 2, 'e:', False)
```

Fig.6. Verificação da consistência das partes secretas partilhadas

II. Operacionalização

Nesta o sistema realiza as operações: Com a função *"*>>> *chavesblindsig"*, a AC gera o par de chaves RSA, que usará para assinatura do voto ofuscado, o eleitor com o número aleatório ">>> ko", ofusca o seu voto e solicita a AC que o assine, invocando a função *">>> ofuscacao".* Este por sua vez, realiza a assinatura cega, invocando a função *">>> assinaturacega"*:

```
def chavesblindsig (bits):

    p, q= next_prime(2**(bits/2)), next_prime(2**(bits))
    n1= p*q
    phi=(p-1)*(q-1)
    e=ZZ.random_element(phi)
    while gcd(e, phi) !=1:
        e=ZZ.random_element(phi)
    d= power_mod(e, -1, phi)
    Chaveprivada= d
    Chavepublica= (n1,e)
    return Chaveprivada, Chavepublica

# Ofuscação ALICE
def ofuscacao (m, Chavepublica):

    n1, e = Chavepublica
    Zn1=IntegerModRing(n1)
    k0=Zn1(randint(2,n1-1))
    m0=(k0**e)*m
    return (k0, m0)
```

Fig. 7. Voto ofuscado antes da assinatura cega

O eleitor desofusca o voto e apercebe-se que o voto o pertence, com a invocação de *">>> assinaturalimpa".*

```
def assinaturacega (m0, Chaveprivada, Chavepublica):

    n1, _ = Chavepublica
    d = Chaveprivada
    sig =m0**d
    return sig

# Desofuscação ALICE
def assinaturalimpa (sig, k0, Chavepublica):

    n1, e = Chavepublica
    sigm = (sig)*(1/k0)
    return sigm
```

```
Digite um numero para votar:5
('O(a) senhor(a) votou no candidato/ partido numero:', 5)
Um Voto válido.
('O voto assinada e:', 13474043)
```

Fig. 8. Assinatura do voto

Depois do voto ter sido assinado, com a chave pública da ACE, cifra-o, invocando a função *">>> cifracaovoto"* e obtém o criptograma (voto cifrado).

```
def cifracaovoto (PuKey, m):

    p, g, y =PuKey
    Zp=IntegerModRing(p)
    k1=randint(2, p-1)
    Alfa=g**k
    Beta=m*(y**k)
    Criptograma=[Alfa, Beta]
    return Criptograma, k1
```

```
Quantos concorrentes estao inscritos?10

Digite um numero para votar: 12

Digite um numero para votar:12

Voto errado. Digite outro numero maior que 0:

Voto errado. Digite outro numero maior que 0: 0

Digite um numero para votar:5
('O(a) senhor(a) votou no candidato/ partido numero:', 5)
Um Voto válido.

('O voto cifrado e:', [16807, 9858])
```

Fig. 9. Cifração do voto

Como se vê, o sistema não aceita o voto nulo, nem o voto branco, o eleitor é alertado que introduziu um número errado, solicitando uma outra tentativa.

A ACE reconstrói o polinómio e recupera a chave privada, com invocação da função *">>> reconstrucaopolinomio".* Se entre os envolvidos na partilha, apenas um elemento tentar obter a chave privada com a sua parte secreta, não é bem- sucedido:

```
def recostrucaopolinomio (coef, pares):

    Zp=coef[0].parent()
    Pol.<x> = PolynomialRing(Zp)
    k = len(coef)
    pol = Pol(0)
    for j in range(k):
        lj = Pol(Zp(1))
        for i in range(k):
            if j!=i:
                lj = lj * ((x-pares[i][0])/(pares[j][0] - pares[i][0]))
        pol = pol + lj * pares[j][1]
    return pol
```

```
(2, 'partes reconstruiram:', 47734*x + 7631)
(1, 'partes reconstruiram', 55365)
As partes secretas a compartilhar sao:', [[1, 55365], [2, 103099], [3, 150833]])
(2, 'partes reconstruiram:', 47734*x + 7631)
```

Fig. 10. Reconstrução do polinómio e recuperação da chave privada

Tendo em posse a sua chave privada, decifra os votos, o voto de cada eleitor, invocando a função "*>>> Decifracaovoto*":

```
#Decifração- RE
def Decifracaovoto (PuKey, PrKey, Criptograma):

    p, g, y = PuKey
    a0 = PrKey
    Zp=IntegerModRing(p)
    s=(Criptograma[0]**a0)
    t_p=(1/s)*Criptograma[1]
    return t_p
```

```
('O voto decifrado e:', 5)
```

Fig. 11. Decifração do voto cifrado

Não basta ter o voto decifrado, agora a ACE deve verificar também a autenticidade do voto, onde ele recebe a mensagem que lhe informa que o voto é autêntico (True) ou não (False), invocando "*>>> verificacaoassinatura*":

```
def verificacaoassinatura (m, Chavepublica, sigm):

    n1, e = Chavepublica
    if sigm**e==m:
        return True
    else:
        return False
            ('A autencidade do voto e:', True)
            ('A autencidade do voto e:', False)
```

Fig. 12. Verificação da autenticidade do voto

III. Confirmação

Para comprovar o processo foi realizado dentro parâmetros legais e que os votos foram correctamente bem tratados, aplicou-se a prova de conhecimento zero, onde dois agentes, o provador e o verificador interagem e um, prova para o verificador que não houve anomalias no processo. Com a invocação da função "*>>> parametropublico*" que recebe o número de bits, que permitiu gerar os números secretos invocando "*>>> secretsproof* " que serviu para o provador convencer ao verificador, sem ele ter acesso ao segredo. O verificador envia para o provador os números binários [a1, a2, a3], escolhidos aleatoriamente por ele. Com seus números secretos (s1, s2, s3), o provador calcula os parâmetros utilizados na prova: y, x, o verificador recebe-os sem chances ter acesso aos segredos do provador. O com os seus números secretos, calcula os parâmetros (v1, v2, v3) e transmite-os para o verificador. Com seus números binários e os parâmetros provenientes do provador, testa a prova, calculando w e k, de seguida verifica invocando "*>>> vericacaoproof* ": Se y^2 for igual a w, ou y^2 igual a k, então ele (verificador) fica convencido.

Notemos que uma outra entidade maliciosa não passaria da prova, pois, o segredo só é conhecido pelo provador. Se alguém tentasse uma prova fraudulenta, não conseguiria, porque outros números secretos, produziriam outros parâmetros e não exactamente y, x partilhado com o verificador. E, ainda que intersectasse os números binários do verificador, não teria sucesso pelo factos dos números secretos não serem partilhados.

```
#Prova
def parametropublico (bits):

    p, q= random_prime(2**(bits/2)), random_prime(2**(bits/2))
    n2 = p*q
    return n2

#Escolha dos segredos
def secretsproof (n2):

    Zn2 = IntegerModRing(n2)
    s1 = Zn2.random_element()
    s2= Zn2.random_element()
    s3 = Zn2.random_element()
    return s1, s2, s3

# Interação entre Paula (provadora) e Veigas (verificador)
def Comunicacaoproof (n2, s1, s2, s3):

    Zn2 = IntegerModRing(n2)
    r = Zn2.random_element()
    w = Zn2(1 - 2 * randint(0, 1))
    x = w*r**2
    a1 = randint(0, 1)
    a2 = randint(0, 1)
    a3 = randint(0, 1)
    y = r*(s1**a1)*(s2**a2)*(s3**a3)
    return y, x, [a1, a2, a3]
```

Fig. 13. Parâmetros partilhados na interação entre o provador e o verificador

Depois dos dois agentes terem partilhado seus números secretos, o provador executa a prova, e pode ser solicitada mediante a invocação de ">>> *Proof* " e para a verificação, invocando ">>> *Verificacaoproof* ". Se o teste positivo, o resultado é True, e o negativo (números binários ou secretos adulterados), o resultado é False.

```
# Momento da prova
def Proof (n2, s1, s2, s3):

    Zn2=IntegerModRing(n2)
    v1 = Zn2(s1)**2
    v2 = Zn2(s2)**2
    v3 = Zn2(s3)**2
    comun = Comunicacaoproof (n2, s1, s2, s3)
    return (v1, v2, v3, comun[0:2]), comun[2]
# Verificação
def Verificacaoproof (proof, lista_a):

    v1, v2, v3, (y, x) = proof
    [a1, a2, a3] = lista_a
    w = -x*(v1**a1)*(v2**a2)*(v3**a3)
    k = x*(v1**a1)*(v2**a2)*(v3**a3)
    if (y**2 == w or y**2 == k):
        return True
    else:
        return False
```

```
('O verificador escolheu:', [1, 0, 0])
('O verificador fica convencido, e diz:', True)
(' O verificador nao correspondido:', [2, 1, 1])
(' O provador nao o convence, e ele diz:', False)
```

Fig. 14. Resultados finais da fase de confirmação

Conclusão

Portanto, foi possível a construção de um sistema de eleição eelctrónica com uma tecnologia de segurança robusta, cuja implementação foi feita em python. O sistema abordado ou proposto é eficiente computacionalmente e seguro em todas suas fases, facto comprovados pelos resultados oriundos da execussão do código. Os resultados foram devidamente fundamentados, mediante as explicações detalhadas dadas ao funcionamento do sistema e o significado de cada resultado.

A robustez do sistema, resume-se no facto de ser sigiloso, verificável, auditável, o voto é tratado nos padrões de segurança recomendado internacionalmente e o eleitor é mantido anónimo durante todo processo.

Diferentemente dos sistemas de outros autores, a segurança desta proposta não depende da honestidade das entidades envolvidas na funcionalidade do esquema.

Os principais aspectos inovadores tidos em conta na minha proposta em comparação com as abordagens de outros, são: o armazenamento da chave privada usada na decifração dos votos, assinatura cega e a prova de conhecimento zero.

- **Armazenamento da chave privada**

Nesta proposta, procurou-se manter um elevado nível de segurança da chave de cifração, pois, ela foi partilhada entre agentes de confiança e, sempre que necessário, pode ser reconstruida (recuperada) pelo responsável do escrutínio e utilizada para fins legais. Este é um procedimento de armazenamento secreto robusto, visto que a chave secreta deve ser inacessível por agentes não autorizados. Todavia, é um procedimento resultante da aplicação do polinómio interpolador da Lagrange no esquema de “Shamir”

- **Prova de conhecimento zero**

Esta é uma técnica aplicada na fase da confirmação, que permite ao agente responsável pelo escrutínio estar munido de atributos para verificar se o processo foi devidamente bem tratado. E sempre que haver alguma anomalia, ele conseguirá convencer o emissor e rejeitar os votos.

Recomendações

À comunidade académica, entidades governamentais e interessados na proposta, recomenda-se o seguinte:

1) A tecnologia de segurança envolvida no sistema, é no âmbito de software e não de hardware. Cabendo aos futuros investigadores, estudarem os aspectos de segurança a nível dos equipamentos tecnológicos a serem utilizados na eleição. Pois, uma acção maliciosa também pode ser arquitetada nos meios tecnológicos pelo fabricante ou durante o transporte, por um outro agente. Isto é, uma introdução (intencional ou não) de vulnerabilidade nos equipamentos que pode ser aproveitada pelos atacantes.
2) Este projecto, é uma etapa inicial de uma ideia macro, de desenvolver um software do género, que seja capaz de facilitar a realização de processos de pequena e grande escala, isto é, que engloba pouco número de eleitor, ou em eleições que abarca um país, por exemplo.
3) A implementação do software, carecerá a avaliação do impacto, que verá medida mediante a aplicação de instrumentos de recolha de dados depois da disponibilização e utilização do projecto.

Referências bibliográficas

1) Barbosa, R. T. N. (2017). Algoritmos criptográficos e o seu desempenho no arduino. ISEP. instituto Superior de Engenharia do Porto: Portugal.

2) Barroso, L. B. (2016). Criptografia: Aspectos teóricos e Proposta de Autenticação Utilizando dados cadastrais do Usuário. Universidade Federal do Rio de janeiro: Brasil.

3) Costa, C. (2010). Introdução à Criptografia Volume 1- Módulo 1. FAPERJ. Rio de Janeiro: Brasil.

4) Instituto Nacional de Estatística (2016). Projecção da População a província do Namibe: 2014- 2050. Luanda: Angola.

5) Lichtler, L. R. (2004). Um sistema seguro para votações digitais. CIP-Catalogação na publicação. Porto Alegre: Brasil.

6) Maziero, C. (2019). Segurança criptográfica: Criptografia assimétrica.

7) Neto, F. N. (2017). Esquemas de assinaturas digitais: o uso de Criptografia assimétrica como um método técnico para autenticidade e não repúdio em emissão de laudos médicos remotos para PACS. Universidade Federal do Rio Grande do Norte: Brasil.

8) Stallings, W. (s. d). Criptografia e segurança em Rede.

9) Pinho, A. (2007). Criptoanálise. Matemática, F. C. T. U. C. Coimbra. Portugal.

10) Preya, G. (2015). E-cient Data Security in Cloud Computing Using the RSA Encryption Process Algorithm. Vol. 4. International Journal of Innovative Research in Science, Engineering and Technology (An ISO 3297: 2007 Certified Organization): UG Scholar, Department of Information Technology, Saveetha School of Engineering. Saveetha University, Chennai, Tamil Nadu, India.

11) Quaresma, T. (2012). Criptoanálise clássica. v24.

Anexos: Implementação da proposta em python

```
def Elgamalinicial (bits):

    p=random_prime(2**bits)
    Zp=IntegerModRing(p)
    g=Zp.multiplicative_generator()
    n=input("Quantos participantes tera o esquema de partilha da CHAVE PRIVADA?")
    while n==1:
        n=input("Impossivel partilhar a chave. Insira um numero de participantes maior que 1
    k=input("Quantas partes partes secretas serao necessarias para RECONSTRUIR A CHAVE?")
    while k<=1 or k>n:
        k=input("Insira outro valor de k, menor ou igual que n para RECONSTRUIR A CHAVE?")
    a0= ceil(randint(2, p-1))
    y=g**a0
    PuKey=(p, g, y)
    PrKey=a0
    return PuKey, PrKey, n, k, Zp, g
```

```
def Coefpolinomio (Zp, Prkey, n, k):

    a0=Prkey
    #Zp= IntegerModRing(p)
    t=k-1
    coef = [a0]
    for r in range (1, t+1):
        a = Zp.random_element()
        coef.append(a)
    return coef

# Construção do polinómio BOB
def polinomioconstruido (coef):

    Zp=coef[0].parent()
    Pol.<x> = PolynomialRing(Zp)
    polinomio=Pol(coef)
    return polinomio

# Partes à partilhar BOB
def partespartilhadas (n, polinomio):

    pares=[]
    for x in range (1, n+1):
        pares.append([x, polinomio(x)])
    return pares
```

```
i=0
pot=[]
def potencia (g, coef):

    pot=[]
    k=len(coef)
    for i in range (k):
        pot.append(g**coef[i])
    return pot

# Verificação da consistência das partes secretas- MEC

def verificacaopartes (i, g, pot, parte_secreta):

    prod=1
    k=len(pot)
    for j in range (k):
        prod =prod*(pot[j]**(i**j))
    if g**parte_secreta == prod:
        return True
    else:
        return False
```

```
def assinaturacega (m0, Chaveprivada, Chavepublica):

    n1, _ = Chavepublica
    d = Chaveprivada
    sig =m0**d
    return sig

# Desofuscação ALICE
def assinaturalimpa (sig, k0, Chavepublica):

    n1, e = Chavepublica
    sigm = (sig)*(1/k0)
    return sigm
```

```
def cifracaovoto (PuKey, m):

    p, g, y =PuKey
    Zp=IntegerModRing(p)
    k1=randint(2, p-1)
    Alfa=g**k
    Beta=m*(y**k)
    Criptograma=[Alfa, Beta]
    return Criptograma, k1
```

```
def recostrucaopolinomio (coef, pares):

    Zp=coef[0].parent()
    Pol.<x> = PolynomialRing(Zp)
    k = len(coef)
    pol = Pol(0)
    for j in range(k):
        lj = Pol(Zp(1))
        for i in range(k):
            if j!=i:
                lj = lj * ((x-pares[i][0])/(pares[j][0] - pares[i][0]))
        pol = pol + lj * pares[j][1]
    return pol
```

```
#Decifração- RE
def Decifracaovoto (PuKey, PrKey, Criptograma):

    p, g, y = PuKey
    a0 = PrKey
    Zp=IntegerModRing(p)
    s=(Criptograma[0]**a0)
    t_p=(1/s)*Criptograma[1]
    return t_p
```

```
#Decifração- RE
def Decifracaovoto (PuKey, PrKey, Criptograma):

    p, g, y = PuKey
    a0 = PrKey
    Zp=IntegerModRing(p)
    s=(Criptograma[0]**a0)
    t_p=(1/s)*Criptograma[1]
    return t_p
```

```
def verificacaoassinatura (m, Chavepublica, sigm):

    n1, e = Chavepublica
    if sigm**e==m:
        return True
    else:
        return False
```

```
#Prova
def parametropublico (bits):

    p, q= random_prime(2**(bits/2)), random_prime(2**(bits/2))
    n2 = p*q
    return n2

#Escolha dos segredos
def secretsproof (n2):

    Zn2 = IntegerModRing(n2)
    s1 = Zn2.random_element()
    s2= Zn2.random_element()
    s3 = Zn2.random_element()
    return s1, s2, s3

# Interação entre Paula (provadora) e Veigas (verificador)
def Comunicacaoproof (n2, s1, s2, s3):

    Zn2 = IntegerModRing(n2)
    r = Zn2.random_element()
    w = Zn2(1 - 2 * randint(0, 1))
    x = w*r**2
    a1 = randint(0, 1)
    a2 = randint(0, 1)
    a3 = randint(0, 1)
    y = r*(s1**a1)*(s2**a2)*(s3**a3)
    return y, x, [a1, a2, a3]
```

```
# Momento da prova
def Proof (n2, s1, s2, s3):

    Zn2=IntegerModRing(n2)
    v1 = Zn2(s1)**2
    v2 = Zn2(s2)**2
    v3 = Zn2(s3)**2
    comun = Comunicacaoproof (n2, s1, s2, s3)
    return (v1, v2, v3, comun[0:2]), comun[2]
# Verificação
def Verificacaoproof (proof, lista_a):

    v1, v2, v3, (y, x) = proof
    [a1, a2, a3] = lista_a
    w = -x*(v1**a1)*(v2**a2)*(v3**a3)
    k = x*(v1**a1)*(v2**a2)*(v3**a3)
    if (y**2 == w or y**2 == k):
        return True
    else:
        return False
```