Système de vote électronique

Auteurs :
Vi Long Luong
Aristide Bronchard

26 avril 2024

1 Introduction

Le but de ce projet est d'implémenter en Python un système de vote électronique faisant appel à des notions essentielles de cryptographie. En effet, nous allons simuler un vote et grâce à un algortihme de crytographie, chaque vote sera chiffré jusqu'au moment du dépouillage, où tous les votes seront déchiffrés afin d'être lisibles. Cette méthode est utilisée pour des raisons de sécurité et notamment pour garantir la confidentialité des votes ainsi que l'éligibilité des électeurs.
Comme évoqué, nous avons implémenté différents algorithmes de cryptographie afin de chiffrer et déchiffrer les votes, tout en garantissant l'éligibilité des bulletins de vote avec les algorithmes de signature. Voici les algorithmes qu'on abordera dans le cadre de ce projet : DSA, EC DSA, El Gamal et EC El Gamal.

1.1 L'algorithme DSA

Le Digital Signature Algorithm (DSA) est basé sur des concepts mathématiques associés à la théorie des nombres et des problèmes de logarithmie discrète, ce qui lui confère une résistance à de nombreux types d'attaques cryptanalytiques si implémenté et utilisé correctement. L'algorithme DSA fonctionne en trois grandes étapes : la générations des clés publiques et privées, la signature du message ainsi que la vérification de la signature.

1.2 L'algorithme EC DSA

L'Elliptic Curve Digital Signature Algorithm (ECDSA) est une variante du Digital Signature Algorithm (DSA) qui utilise la cryptographie sur les courbes elliptiques pour fournir un mécanisme de signature numérique. Comme le DSA classique, ECDSA permet de créer une signature numérique sur des données, signature qui peut ensuite être vérifiée par toute personne ayant accès à la clé publique du signataire. ECDSA offre des niveaux de sécurité comparables à DSA mais avec des clés plus courtes, ce qui le rend plus efficace en termes de performance.

1.3 L'algorithme d'El Gamal

L'algorithme El Gamal est un système de chiffrement asymétrique basé sur le problème du logarithme discret. Il est souvent utilisé pour le chiffrement de messages et comme une méthode de signature numérique, bien que son usage le plus fréquent soit pour le chiffrement. L'algorithme El Gamal se base sur les principes de la cryptographie à clé publique et s'articule autour de trois phases principales : la génération des clés, le chiffrement et le déchiffrement.

1.4 L'algorithme EC El Gamal

L'algorithme EC El Gamal (Elliptic Curve El Gamal) est une adaptation de l'algorithme de chiffrement El Gamal utilisant la cryptographie sur les courbes elliptiques. Il offre les mêmes fonctionnalités que le El Gamal classique, mais il est généralement plus sûr et plus efficace en raison de la complexité accrue du problème du logarithme discret sur les courbes elliptiques par rapport aux groupes multiplicatifs de corps finis.

2 Implémentation du vote électronique

2.1 Généralités

Notre système de vote électronique utilise des techniques de cryptographie pour garantir à la fois la confidentialité des votes et l'éligibilité des électeurs. Voici une explication détaillée du processus, étape par étape, en utilisant l'approche de chiffrement El Gamal ou EC El Gamal avec la signature numérique DSA ou ECDSA.
Notre simulation comporte cinq candidats ainsi que dix électeurs. Les électeurs sont représentés par des listes de cinq éléments, représentant chacun un candidat.
Chaque vote pour un candidat est représenté par une liste de cinq éléments, où un élément est mis à 1 pour indiquer le candidat choisi, et les autres sont mis à 0. Par exemple, un vote pour le candidat C1 est représenté par la liste (1, 0, 0, 0, 0). Le vote blanc est impossible, il faut absolument voter pour au moins un candidat. Dans notre implémentation, un élément de la liste correspond à un message qui pourra être chiffré.
Le vote se déroulera donc de la façon suivante : Une paire de clés (une clé publique et une clé privée) pour la signature est générée et attribuée à chaque électeur. L'électeur envoie son vote au système et ce dernier le chiffrera grâce à l'algorithme El Gamal ou EC El Gamal. Il le renverra ensuite à l'électeur qui signera le vote grâce à sa clé privée puis le renverra au système. Enfin, le système vérifie que la signature est valide grâce à la clé publique correspondante. Si la signature est valide, le vote est comptabilisé et il sera déchiffré lors du dépouillage.

2.2 Implémentation en Python

Pour mettre en place ce système de vote électronique, nous avons implémenté des algorithmes de chiffrement/déchiffrement ElGamal (elgamal.py) et EC ElGamal (ecelgamal.py), des algorithmes de signature DSA (dsa.py) et ECDSA (ecdsa.py) en Python. Vous trouverez un répertoire *tests* contenant des tests qui assurent la bonne implémentation de ces algorithmes.
En plus, pour représenter notre électeur, nous avons défini un objet **Voter** qui permettra à notre électeur de voter et de signer son bulletin de vote grâce aux méthodes **voting** et **sign**.

```
import random
from dsa import DSA_sign
from ecdsa import ECDSA_sign

class Voter:
    # Constructor
    def __init__(self, sign_key)

    # Return the decision of the voter
    def voting(self)

    # Sign each ballot with a specific method (DSA/ECDSA)
    def sign(self, ballot, method='DSA')
```

Nous avons également défini un objet **VoteSystem** qui représente notre système de vote et qui contient des méthodes indispensables pour le bon fonctionnement du système.

```
from dsa import *
from ecdsa import *
from elgamal import *
from ecelgamal import *

from Voter import *

NB_CANDIDATES = 5
NB_VOTERS = 10

class VoteSystem:

    ## Constructor
    def __init__(self, encrypt_method='ElGamal', sign_method='DSA'):

    ## Display the votes in plain text
    def display_votes(self)

    ## Encryption of the vote
    def encrypt_vote(self, vote)

    ## Verify the signature of the voter
    def verify_vote(self, signature, encrypted_vote, public_sign_key):

    ## Add the vote to the system
    def add_vote(self, vote)

    ## Decrypt and count votes
    def decrypt_votes(self)

    ## Display the result of the vote
    def display_result(self)
```

A l'initialisation, un **VoteSystem** va initialiser un tableau **candidates** avec 5 places utilisées pour le dépôt et le dépouillage de votes et un tableau **voters** de 10 électeurs. A chaque électeur est attribué une clé privée qui servira à la signature du bulletin. Cette clé privée va de paire avec une clé publique, qui permettra la vérification de la signature. Quand un objet **VoteSystem** est créé, une méthode de chiffrement et une méthode de signature sont également définies en avance.
Comme on a pu voir les fonctions définies, le système de vote prend en charge les tâches suivantes:

1. Quand un électeur donne son vote (contenant 5 messages), le système le chiffre avec la méthode de chiffrement choisie (chaque message est chiffré séparémment).
2. Ensuite, l'électeur signe son bulletin avec la méthode de signature choisie, en utilisant la clé privée qui lui a été attribuée.
3. Le système vérifie l'éligibilité de l'électeur avec la clé publique correspondante à la clé privée utilisée.
4. Si la vérification passe, le système comptabilise le vote.
5. A la fin du vote, il déchiffre les votes et affiche le résultat.

L'entrée du programme est le fichier **main.py** qui décrit aussi le déroulement de notre système de vote.

```python
from VoteSystem import *

def main():
    # Ask the user to choose the encrypt method and the signature method
    encrypt, sign = prompt()

    encrypt_method = 'ElGamal' if encrypt == 1 else 'EC_ElGamal'
    sign_method = 'DSA' if sign == 1 else 'ECDSA'

    vote_system = VoteSystem(encrypt_method, sign_method)

    # vote_system.display_votes()

    for voter, public_sign_key in vote_system.voters:

        # Encrypt vote
        encrypted_vote = vote_system.encrypt_vote(voter.voting())

        # Encrypt vote
        ballot = vote_system.encrypt_vote(voter.voting())

        # Aggregate vote
        b = b''
        for encrypted_message in ballot:

            c1, c2 = encrypted_message

            if vote_system.encrypt_method == 'ElGamal':
                b += int_to_bytes(c1) + int_to_bytes(c2)
            elif vote_system.encrypt_method == 'EC_ElGamal':
                c = add(c1[0], c1[1], c2[0], c2[1], p)
                b += int_to_bytes(c[0]) + int_to_bytes(c[1])

        # Voter signs its ballot
        r, s = voter.sign(b, vote_system.sign_method)

        # Verify signature
        if not(vote_system.verify_vote((r, s), b, public_sign_key)):
            print("Signature is not verified !!!")
            exit(1)

        # If all ballots signature have been verified, add the vote to the system
        vote_system.add_vote(encrypted_vote)

    # Display the result of the vote
    vote_system.display_result()


if __name__ == '__main__':
    main()
```

Vous pouvez regarder plus en détail les fichiers **Voter.py**, **VoteSystem.py** et **main.py** pour mieux comprendre comment est implémenté le code.

2.3 Déroulement du programme

Pour exécuter le programme, il faudra entrer la commande **python3 main.py**.

```
vi-long-luong@epita:~/Crypto/electronic-voting$ python3 main.py
```

Figure 1: Lancer le programme

Dans un premier temps, un invite de commande vous demandera de choisir une méthode de chiffrement et une méthode de signature.

```
Please choose a method for encryption/decryption of your ballot
[1] ElGamal     [2] EC ElGamal :   1
Please choose a method for signature verification of your ballot
[1] DSA         [2] ECDSA :   2
```

Figure 2: Choix des méthodes

Par exemple, ici, j'ai choisi El Gamal pour chiffrer les votes et ECDSA pour signer les bulletins de vote.
Ensuite, le système déroulera le processus de vote et affichera le résultat final.

```
Candidate 1 has 3 votes
Candidate 2 has 2 votes
Candidate 3 has 0 votes
Candidate 4 has 4 votes
Candidate 5 has 1 votes

Congratulations, the candidate 4 has winned the election with 4 votes !!!
```

Figure 3: Résultat du vote

Pour l'instant, le choix des électeurs est décidé aléatoirement pour simuler le processus du vote. On peut imaginer par la suite que le système donnera la main à chaque électeur pour constituer son propre vote.