Université des Sciences et de la Technologie Houari Boumediene

Rapport de projet (Vote sécurisé)

Réalisé par :
DERRADJI Zakaria
AISSAT Achour
SI LAKHAL Aymen
HABOU Ayoube
MAIZIA Mohamed

Sécurité des systèmes informatique
Master 1
2019-2020

Table des matières

Introduction

Dans ce projet nous devons réaliser un vote électronique tout en gardant la confidentialité, l'authentification et l'intégrité du vote, ce vote doit assurer les objectifs suivants :

- Authentification par carte à puce (RSA) .
- Un étudiant peut voter qu'une seule fois.
- On doit garder la confidentialité et l'intégrité de vote des étudiants.
- Ce système crée des sessions de vote avec un délai limité.

Pour cela on fait appelle à des outils mathématiques qui nous permettre d'atteindre les objectifs cités précédemment.

Outils

Carte à puce

Définition

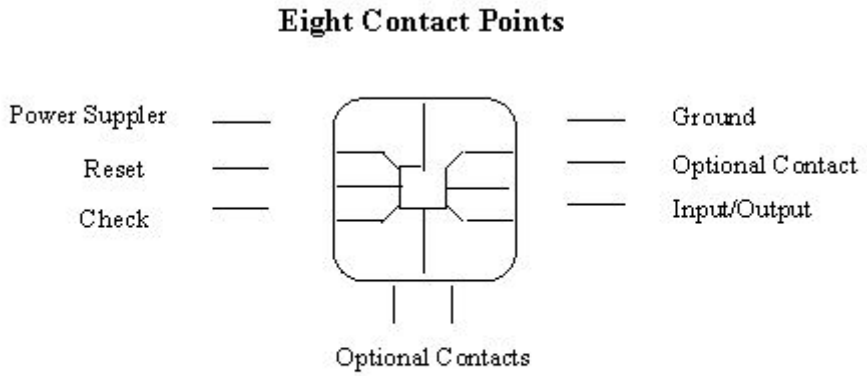
Java Card fait référence à une technologie logicielle qui permet aux applications basées sur Java (applets) d'être exécutées en toute sécurité sur des cartes à puces et des périphériques similaires à faible encombrement mémoire. Java Card est la plus petite des plates-formes Java ciblée pour les appareils intégrés. Java Card donne à l'utilisateur la possibilité de programmer les appareils et de les rendre spécifiques à l'application. Il est largement utilisé dans SIM cartes (utilisé dans GSM téléphones mobiles) et ATM cartes.

Aperçu

Lorsqu'une applet Java Card est créée, une instance est créée et enregistrée dans la table de registre JCRE (environnement d'exécution Java Card). En outre, il ouvre une connexion à l'hôte local sur le port 9025. L'application hôte configuré la connexion au port 9025 pour communiquer avec l'applet. Le JCRE sélectionne une applet sur la carte en fonction des commandes de sélection entrantes. Chaque applet est identifiée et sélectionnée par son identifiant d'application (AID). Les commandes telles que la commande de sélection sont formatées et transmises sous la forme d'unités de données de protocole d'application (APDU). Les applets répondent à chaque commande APDU avec un mot d'état (SW) qui indique le résultat de l'opération. Une applet peut éventuellement répondre à une commande APDU avec d'autres données

Carte Smart

Une carte à puce de la taille d'une carte de crédit stocke et traite les informations via les circuits électroniques intégrés au silicium dans le substrat plastique de son corps. Une carte à puce intelligente contient un microprocesseur et offre des capacités de lecture, d'écriture et de calcul, comme un petit micro-ordinateur. Une carte à puce ne contient pas son alimentation, son écran ou son clavier. Il interagit avec un dispositif d'acceptation de carte (CAD) à l'aide d'une interface de communication, fournie par un ensemble de huit points de contact électriques ou optiques, comme illustré sur la figure suivant



Huit contacts de la carte à puce

Unités de données du protocole d'application

Lorsque deux ordinateurs communiquent entre eux, ils échangent des paquets de données, qui sont construits selon un ensemble de protocoles. De même, les cartes à puce parlent au monde extérieur en utilisant leurs propres paquets de données - appelés APDU (Application Protocol Data Units). Les cartes à puce sont des communicateurs réactifs, c'est-à-dire qu'elles n'initient jamais de communication, elles ne répondent qu'aux APDU à partir du CAD (Card acceptation device). Le modèle de communication est basé sur la réponse à la commande, c'est-à-dire que la carte reçoit une APDU de commande, effectue le traitement demandé par la commande et renvoie une APDU de réponse.
Les tableaux suivants illustrent respectivement les formats APDU de commande et de réponse. La structure APDU illustrée ci-dessous est celle décrite dans ISO-7816.

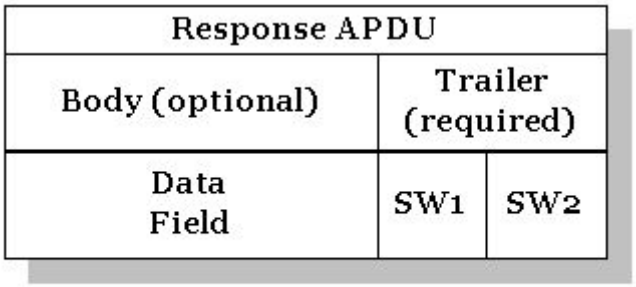
Command APDU						
Header (required)				Body (optional)		
CLA	INS	P1	P2	Lc	Data Field	Le

Commande APDU

L'en-tête code la commande sélectionnée. Il se compose de quatre champs: classe (CLA), instruction (INS) et paramètres 1 et 2 (P1 et P2). Chaque champ contient 1 octet:
CLA: octet de classe. Dans de nombreuses cartes à puce, cet octet est utilisé pour identifier une application.
INS: octet d'instruction. Cet octet indique le code d'instruction.
P1-P2: octets de paramètres. Ceux-ci fournissent une qualification supplémentaire à la commande APDU.

Lc désigne le nombre d'octets dans le champ de données de la commande APDU.

Le indique le nombre maximal d'octets attendu dans le champ de données de l'APDU de réponse suivante.



Réponse APDU

Les octets d'état SW1 et SW2 indiquent l'état de traitement de la commande APDU dans une carte.

Méthodes de l'applet javacard

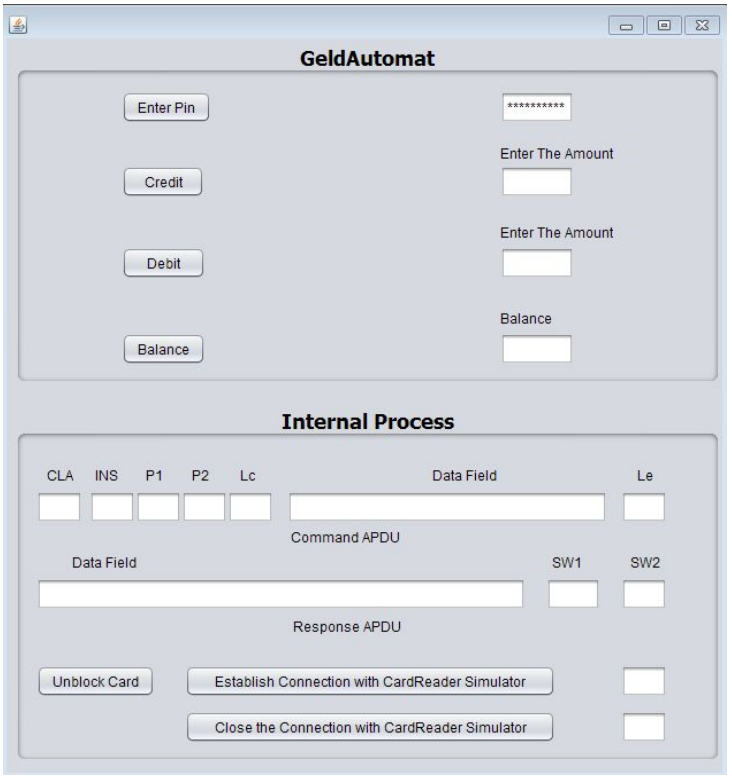
1. install (byte [] bArray, short bOffset, byte bLength)
Le JCRE appelle cette méthode statique pour créer une instance de la sous-classe Applet.
2. sélectionnez ()
Appelé par le JCRE pour informer l'applet qu'il a été sélectionné.
3. désélectionnez ()
Appelé par le JCRE pour informer l'applet actuellement sélectionné qu'une autre (ou la même) applet sera sélectionnée.
4. process (APDU apdu)
Appelé par le JCRE pour traiter une commande APDU entrante.
5. register ()
Cette méthode est utilisée par l'applet pour enregistrer cette instance d'applet auprès du JCRE et affecter l'AID par défaut dans le fichier CAD à l'instance d'applet.
6.register (byte [] bArray, short bOffset, byte bLength)
Cette méthode est utilisée par l'applet pour enregistrer cette instance d'applet avec le JCRE et pour affecter l'AID spécifié dans le tableau bArray à l'instance d'applet.
7. crédit (APDU apdu)
Cette méthode est utilisée pour ajouter un montant au solde. Le montant est spécifié dans le champ de données de l'APDU.
8. débit (APDU apdu)
Dans cette méthode, l'objet APDU contient un champ de données qui spécifie le montant à débiter du solde.
9. getBalance (APDU apdu)
La méthode getBalance renvoie l'équilibre du portefeuille dans le champ de données de l'APDU de réponse. Étant donné que le champ de données dans l'APDU de réponse est facultatif, l'applet doit informer explicitement le JCRE des données supplémentaires.

Application hôte

Les mêmes chaînes d'outils sont utilisées pour créer et compiler l'application hôte de la carte java. Une interface graphique swing est conçue pour interagir avec l'application. L'interface graphique se compose de boutons et de champs de texte pour définir les commandes et les données dans l'APDU.

Méthodes de l'application hôte de la carte Java

1. EstablConnectionToSimulator ()
Cette méthode établit une connexion T = 1 à un simulateur écoutant le port 9025 sur localhost.
2. pwrUp ()
Dans cette méthode, le coomand powerUp est envoyé pour mettre la carte sous tension.
3. setTheAPDUCommands (byte [] cmnds)
Cette méthode définit l'en-tête de l'APDU à envoyer avec les commandes (CLA, INS, P1 et P2) dans APDU.
4. setTheDataLength (octet ln)
Cette méthode définit le (nombre d'octets présents dans le champ de données) Champ Lc de l'APDU
5. setTheDataIn (octet [] données)
Dans cette méthode, les champs de données de l'APDU sont définis avec des données.
6. setExpctdByteLength (octet ln)
Cette méthode définit le nombre d'octets attendus dans le champ apdu.Le.
7. exchangeTheAPDUWithSimulator ()
Cette méthode échange l'APDU avec une carte ou un simulateur.
8. byte [] decodeDataOut ()
Dans cette méthode, les données de l'applet en réponse APDU sont décodées et la méthode renvoie les nombres hexadécimaux à afficher sur l'interface graphique.
9. byte [] decodeStatusBytes ()
Cette méthode décode les mots d'état SW1 et SW2 de l'APDU de réponse et retourne les octets d'état à afficher sur l'interface graphique.
10. String atrToHex (byte atCode)
Cette méthode convertit les commandes atr en nombres hexadécimaux et renvoie les nombres sous forme de chaînes.
11. pwrDown ()
Cette méthode est utilisée pour mettre la carte hors tension.
12. closeConnection ()
Cette méthode ferme la connexion du socket.
L'interface graphique
L'interface utilisateur graphique est divisée en deux sections. Sur une section, le processus du guichet automatique peut être simulé. Dans une autre section, le processus interne a été présenté à l'utilisateur.



Authentification

L'authentification pour un système informatique est un processus permettant au système de s'assurer de la légitimité de la demande d'accès faite par une entité (être humain ou un autre système...) afin d'autoriser l'accès de cette entité à des ressources du système (systèmes, réseaux, applications…), conformément au paramétrage du contrôle d'accès. L'authentification permet donc, pour le système, de valider la légitimité de l'accès de l'entité, ensuite le système attribue à cette entité les données d'identité pour cette session (ces attributs sont détenus par le système ou peuvent être fournis par l'entité lors du processus d'authentification). C'est à partir des éléments issus de ces deux processus que l'accès aux ressources du système pourra être paramétré (contrôle d'accès).

Dans notre cas on a des étudiants qui souhaite faire un vote électronique qui assure l'authentification,chaque étudiant possède une clé privée et une clé publique et un matricule unique qui authentifie l'étudiant.

Pour authentifié l'étudiant on va utiliser le principe de défi (challenge), dans notre démarche pour authentifier un étudiant on suit les étapes suivantes:

- L'étudiant crypte son matricule avec la clé publique du serveur **c=E(matricule,** e_s **)** et envoie **c** au serveur.

- Le serveur décrypte le chiffré **c** avec sa clé privé d_s et extraire le matricule de l'étudiant **matricule=D(c,** d_s **)**, ensuite le serveur génère une phrase secrète et la

crypte avec la clé publique de l'étudiant e_e pour vérifier la clé publique de l'étudiant **c'=E(phrase secrète,** e_e **)** et l'envoie à l'étudiant.

- L'étudiant décrypte **c'** avec sa clé privé d_e **, phrase secrète =D(c',** d_e **),** et il va crypter cette phrase secrète avec la clé publique du serveur e_s **,** et envoyer au serveur **c=E(phrase secrète,** e_s **)**

- Le serveur décrypte **c** avec sa propre clé privé d_s **, phrase secrète' =D(c,** d_s **),** le serveur vérifie l'égalité de sa phrase secrète généré avant avec celle envoyé par l'étudiant si ils sont égaux donc l'étudiant est bien authentifié donc il envoie la clé de session crypté avec la clé publique de l'étudiant (e_e) à l'étudiant **c'=E(clé de session,** e_e **)**

- L'étudiant récupère la clé de session on décryptant **c'** avec sa propre clé privé d_e **clé de session = D(c',** d_e **).**

La figure ci dessous montre les étapes d'authentification

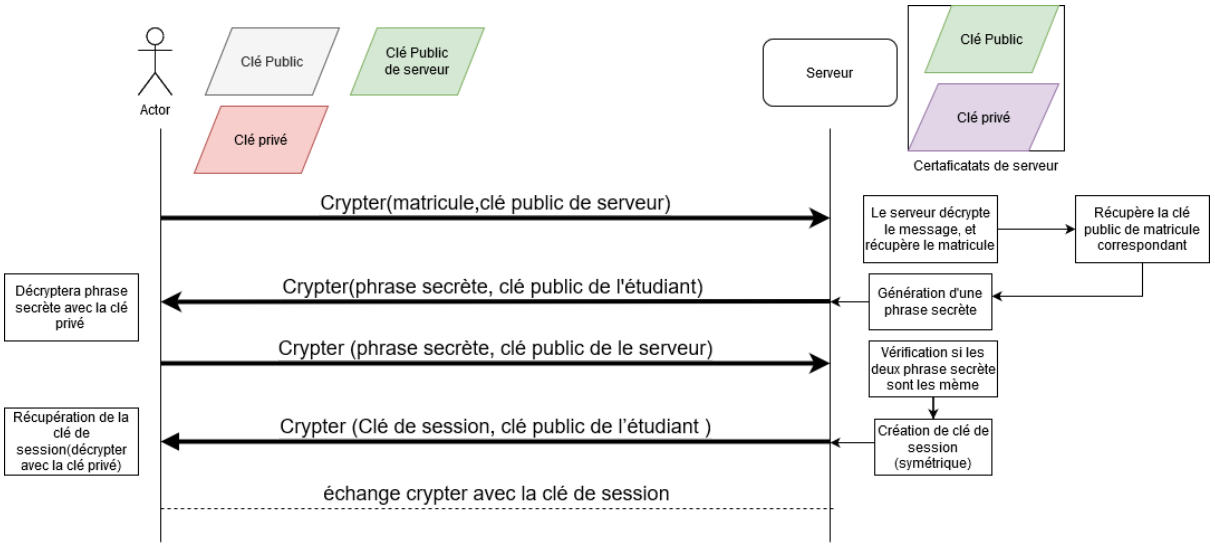


Figure : Les étapes d'authentification

Signature aveugle

En cryptographie, une signature aveugle, telle que définie par David Chaum, est une signature effectuée sur un document qui a été masqué avant d'être signé, afin que le signataire ne puisse prendre connaissance de son contenu. De telles signatures sont donc employées lorsque le signataire et l'auteur du document ne sont pas la même personne. La signature (aveugle) résultante peut être publiquement vérifiée avec le document original (démasqué) comme toute signature numérique. Certains protocoles, dont RSA, permettent

même d'annuler directement l'effet du masque sur la signature, avant la vérification.(source : wikipedia)

On va appliquer la signature aveugle sur le cryptosystème RSA. on montre les étapes mathématiquement, on suppose que Bob va signer un message transmit par Alice sans le connaitre .

Supposons que Bob ait conçu un système cryptographique RSA de clé publique (e,N) et de clé privée d'Alice souhaite voir Bob signer un message m codé sous la forme d'un entier compris entre 0 et N−1. Alice commence par choisir un entier k premier avec N qui joue le rôle de facteur de masquage. Alice transmet ensuite à Bob, l'entier

$$m' = mk^{e} \ mod(N)$$

Ne connaissant pas k, Bob ne peut déterminer m et ne connaît donc pas le message transmis par Alice. Bob peut néanmoins apposer sa signature au message en transmettant à Alice la valeur :

$$m'' = (m')^{d} \ mod(N)$$

A partir de cette signature masqué m'', Alice n'a alors plus qu'à évaluer

$$s = m''\, k^{-1} \ mod(N)$$

pour disposer d'un entier s vérifiant $s^{d} = m \ mod(N)$

Ainsi, avec le couple (m, s), Alice dispose d'un message signé par Bob alors que celui-ci ne connaît pas la nature du message qu'il a indirectement signé.

la figure suivante montre les étapes et les échanges entre Alice et Bob.

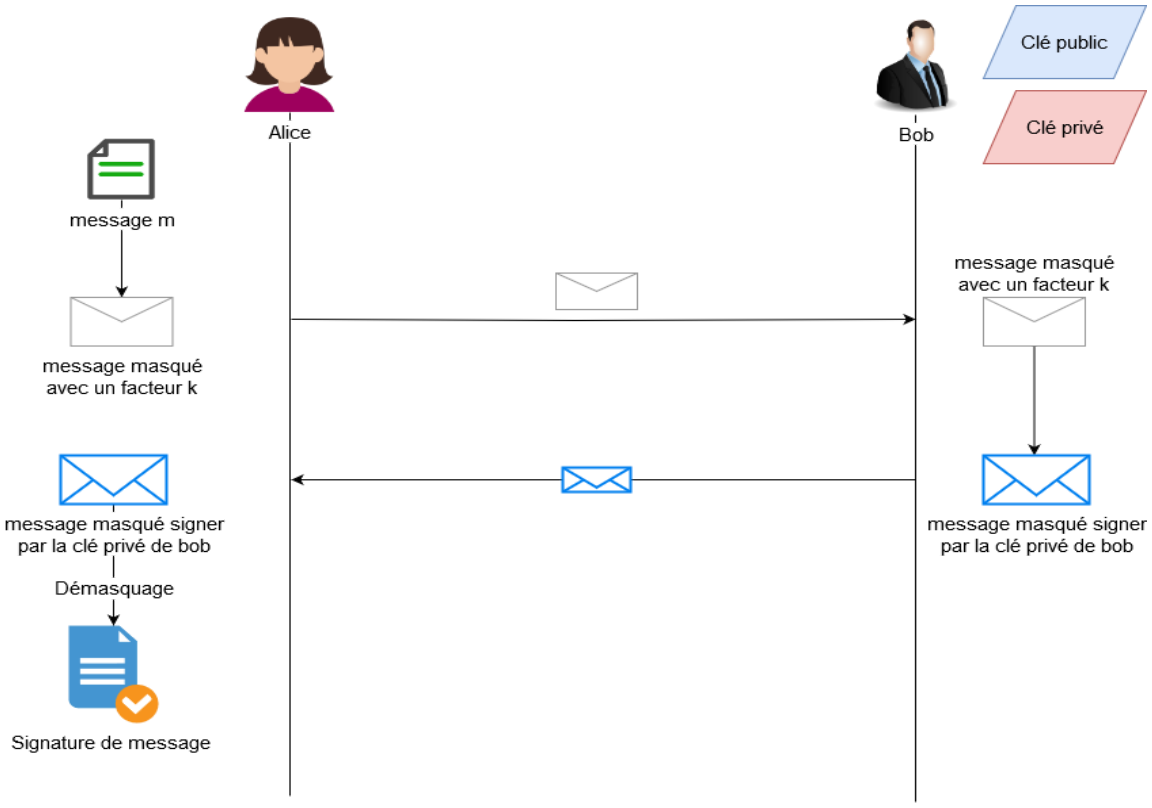


Figure : Les étapes et les échanges signature aveugle.

Les composants

Administrateur

Il fait les tâches suivantes :

1) Connexion des professeurs et des étudiants.
2) Création des comptes étudiants et des professeurs.
3) Création des sessions.
4) Signature aveugle.
5) Création de carte à puces.
6) Création des clés publique et privé pour les étudiants.
7) Gestion des utilisateurs.
8) Affichage des résultats.

Agent

Il fait les tâches suivantes :

1) Génération des Valeur N1 et N2.
2) Envoi des Valeurs N1 et N2 aux étudiants.
3) Il garde les valeurs N1 et le hash de N2.
4) Vérification de N1.
5) Mise à jour de valeur N1 et le hash N2.
6) Préparation des sessions.

Anonymiseur

Il fait les tâches suivantes :

1) Garde les votes crypter jusqu'au la fin de la session de vote.
2) Transmit les votes crypter au décompteur.

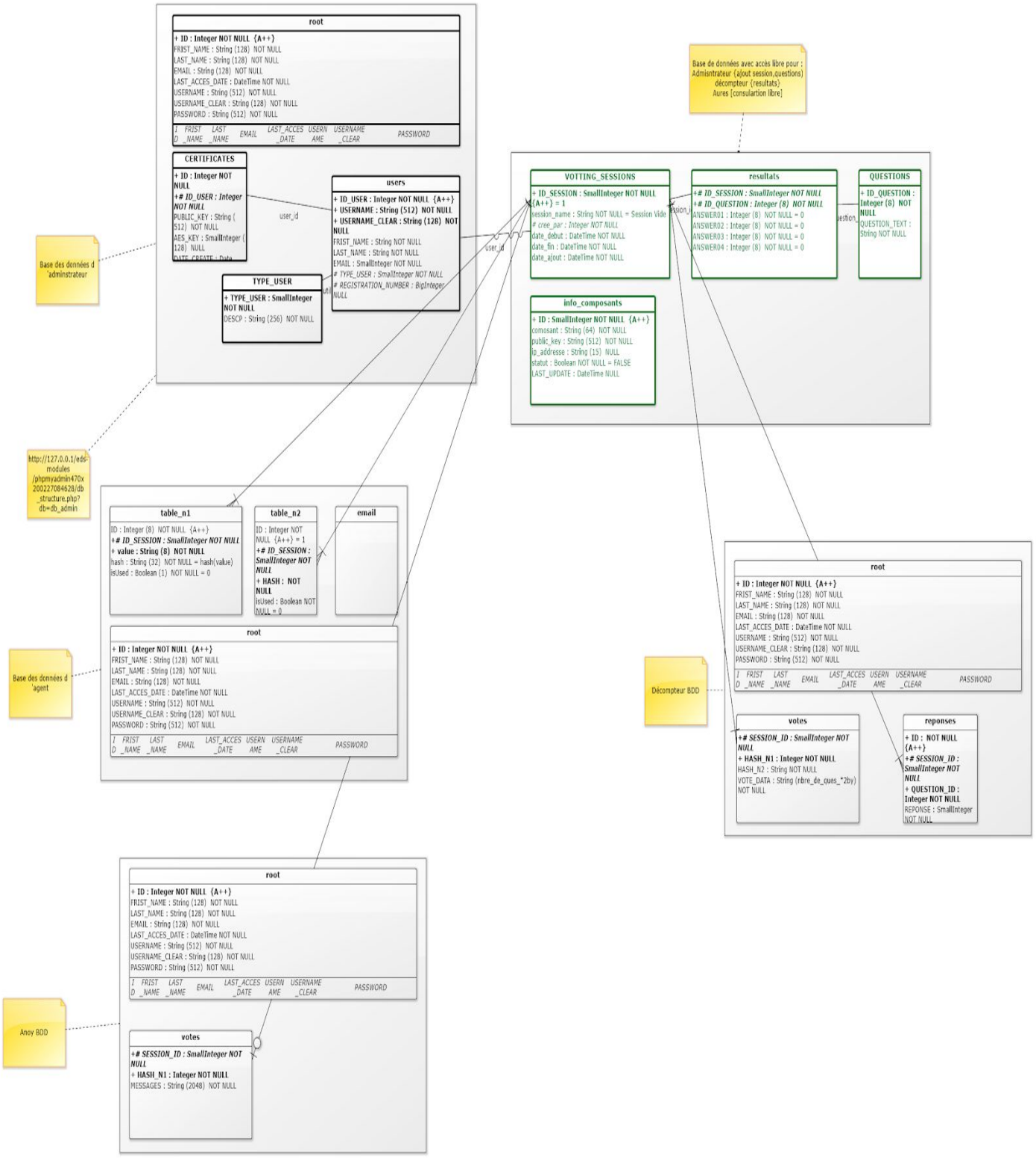
Décompteur

Il fait les tâches suivantes :

1) Décrypter les votes crypter.
2) Vérification des signatures.
3) Vérification de N2.
4) Mise à jours les résultats.
5) Comptera les voix une fois la session de vote finalisée.

Conception de la base des données

Chaque composant a son propre base des données, mais il a y une base des données accessible par des 4 composant pour les mise à jours .
la figure suivante montre la conception des bases des données et les relations entre eux.



Scénario de vote

Dans ce chapitre on va expliquer les étapes de :

1) Création de la session et la préparation : Comment un professeur fait une demande pour créer une session de vote, et quelle composant vont intervenir pour contacter les étudiants.
2) Le vote effectué par un étudiant : Comment un étudiant valide son vote.
3) Le décompte des voix : après la fin de session de vote, comment le décompteur compte les voix et assure l'intégrité des voix.

Préparation de session

Un prof connecte au adminstrateur, et demande de création de la session; à ce moment là l'administrateur va contacter l'agent en même temps il ajoute la session au base de données.

L'agent commence la création de deux valeurs N1 et N2, il envoi ces deux valeurs N1 et N2 pour chaque étudiants. la valeur N1 est le code d'authentification, par contre N2 est la valeur pour assurer l'intégrité de vote.

Après l'envoi des ces deux valeur l'agent garde la valeur N1 en clair et calcule le hash de N2 et supprime la valeur N2 en claire.

Quand l'envoi est terminé, il envoi l'acquittement de fin au administrateur. L'administrateur envoi l'acquittement au professeur et déclare que la session est démarrée. Dans cette étapes :

1) génération des valeurs N1 et les hashes N2.
2) envoi les N1 et N2 au étudiants.

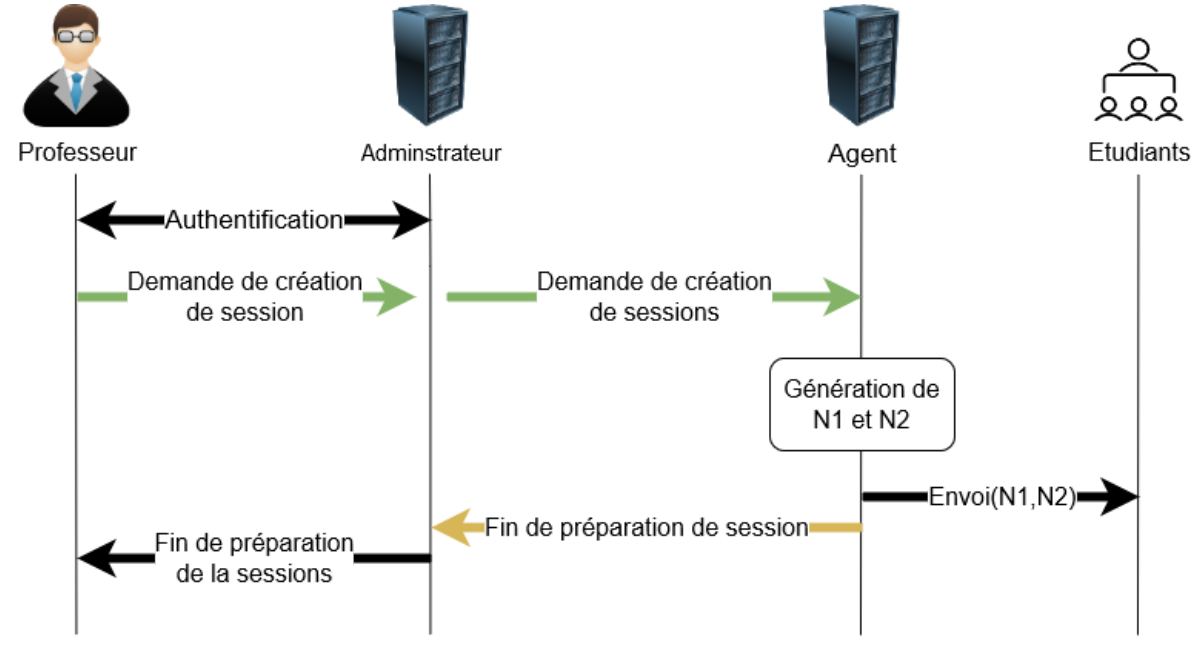


Figure : Préparation de vote

Le Vote

La session de vote commence, et les étudiants commence de voter.

L'étudiants connecter au Adminstrateur, avec sa carte à puce, et fait le défi. L'administrateur envoi au étudiant les sessions en cours.

Il commence le vote, et il fait ces choix. a ce moment là il crée son propre vote de la forme suivante :

message de vote = [valeur de N2, Choix de Vote]

il masque le message de vote et il demande au adminstrateur de le signer.

Pour que l'administrateur accepte de le singer, l'étudiant envoi la valeur N1. L'Administrateur demande au agent si la Valeur N1 existe, pour assurer que cette personne a le droite de voter. Ensuite l'administrateur envoi la signature de message de vote masquer, ensuite l'étudiant la démasque.

L'étudiant ajout au message de vote la signature de l'administrateur et le crypte avec la clé public de décompteur.

message crypter = Crypter ([message de voté, la signature], clé publique de décompteur)

L'étudiant envoi le *message crypté* au Anonymiseur, avec la valeur N1. Pour assurer que cette personne a le droit de vote; l'anonymisation demande l'existence de N1 au Agent, si elle existe, l'agent supprimer la valeur N1 de la base de données. Dans cette étape Anonymiseur sauvegarde le *message crypté* et envoi un accusé au étudiant.

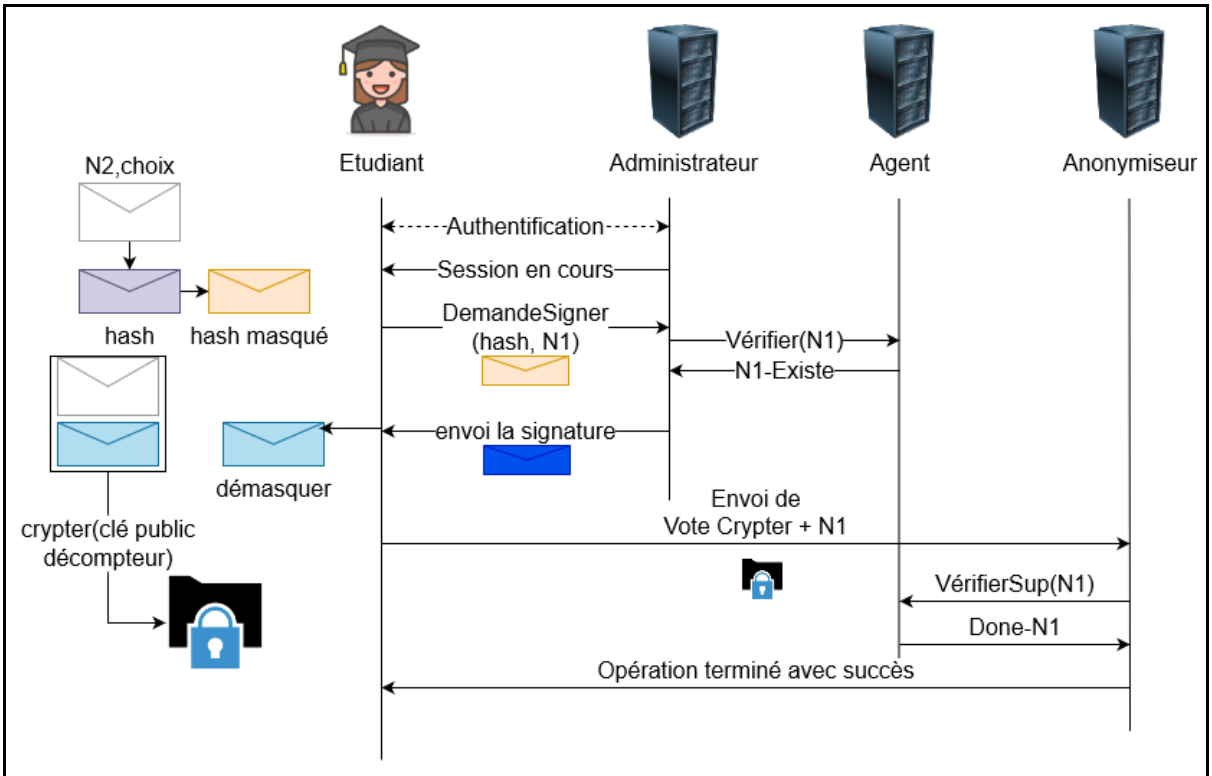


Figure : Opération de Vote d'un étudiant

Comptage des voix

Après la fin de délai de session, l'Administrateur ne singer plus, et le décompte démarre.
L'anonymiseur envois tous les messages cryptés au décompteur.
Le décompteur décrypter tous les messages cryptés reçus, puis il commence la vérification de chaque vote en deux étapes :

1) avec la clé publique de l'administrateur, il vérifier l'authenticité de vote.
2) avec la valeur N2 dans le vote, il demande au Agent de calculer le hasher de N2 et vérifier son existence, si il existe il sera supprimer.

Après ces deux vérification le vote sera compté et les données sont à jour.

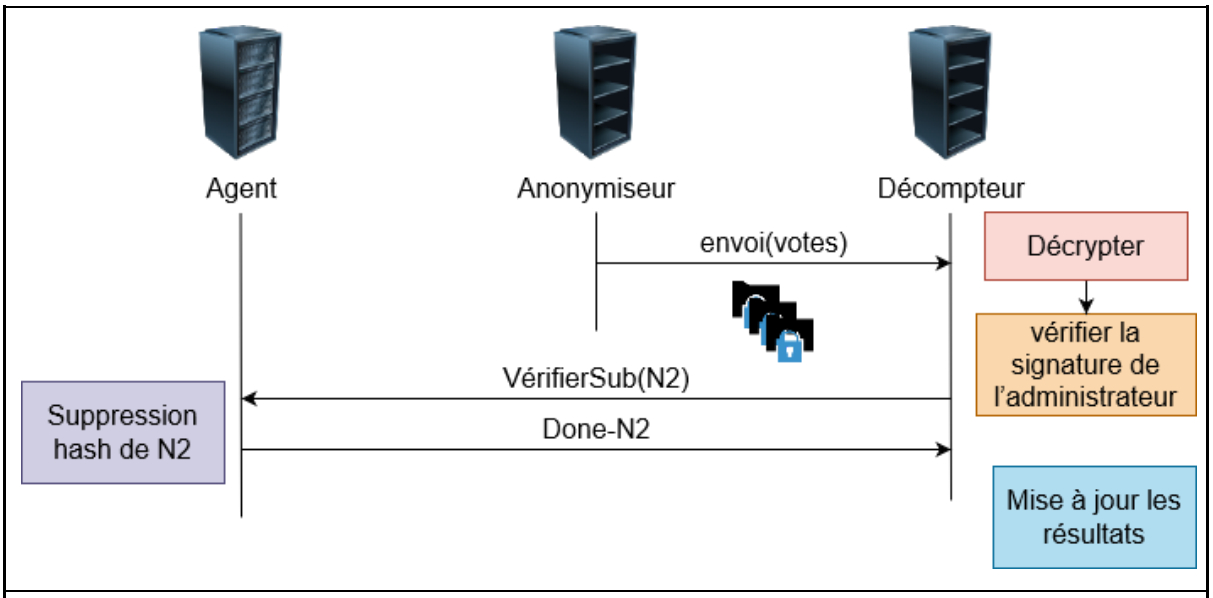


Figure : Comptage des voix

Interface :

Partie Administrateur

Démarrage de serveur d'administrateur/ interface graphique de l'adminsration

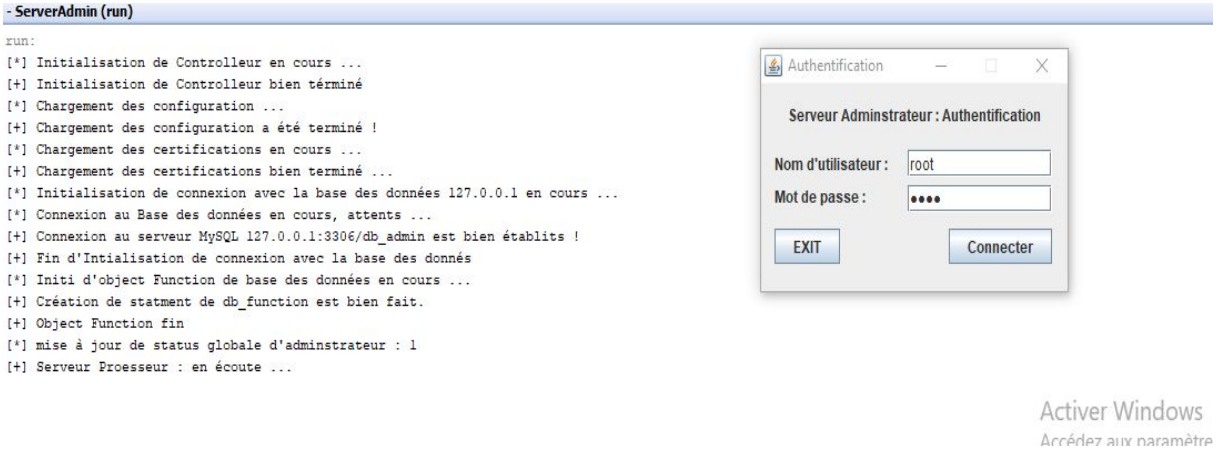
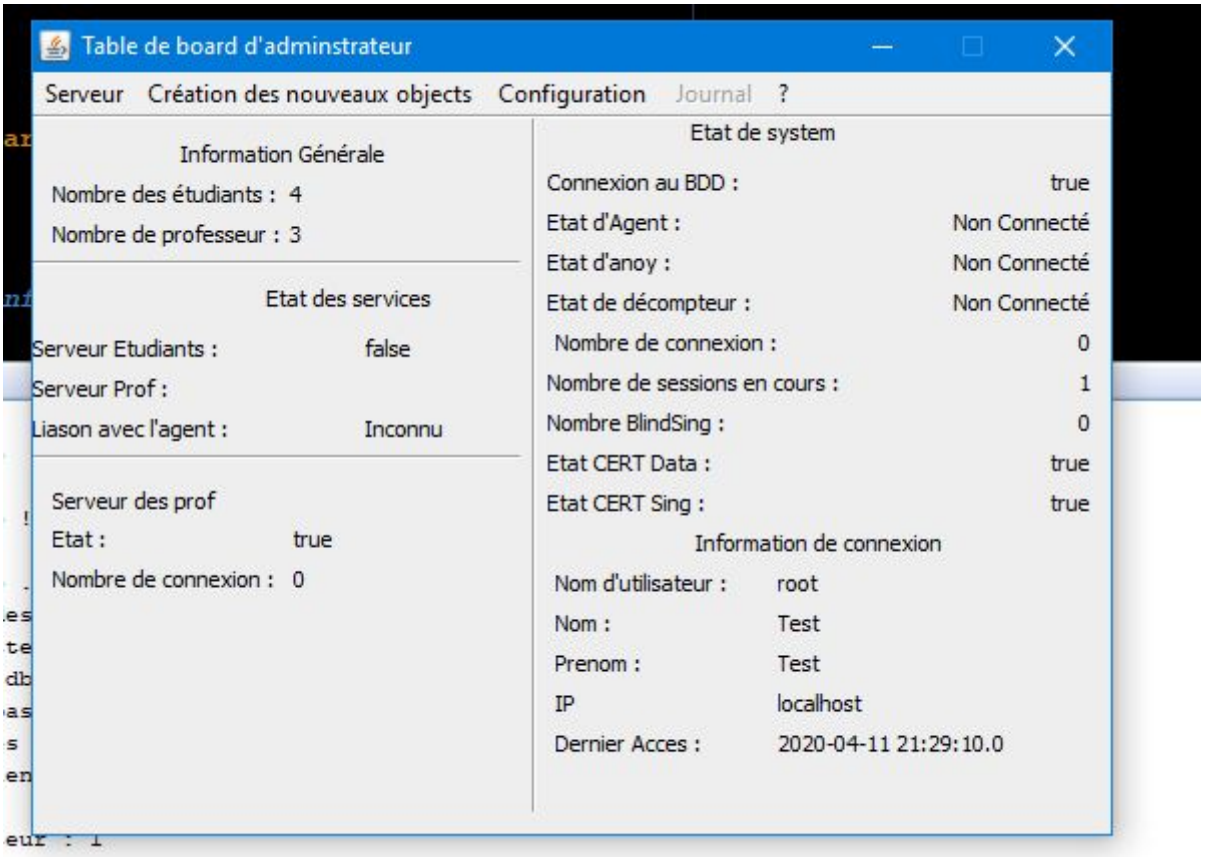
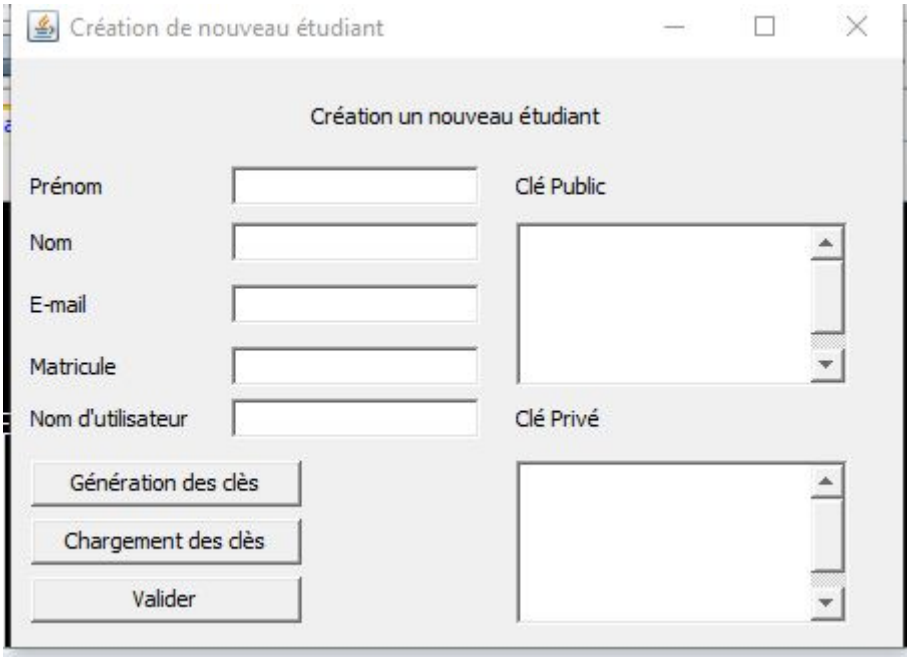


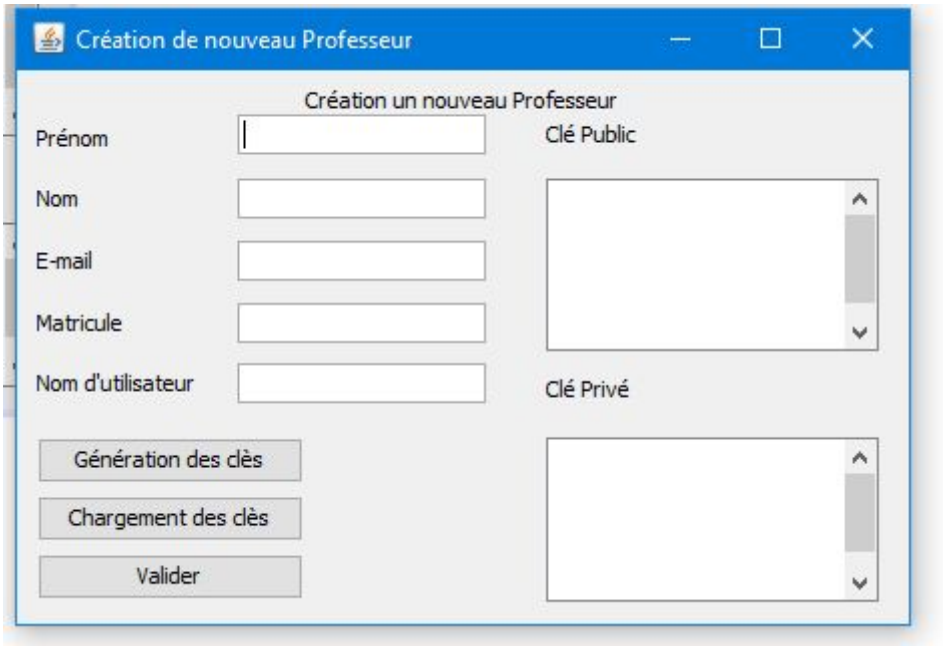
Tableau de bord d'administrateur



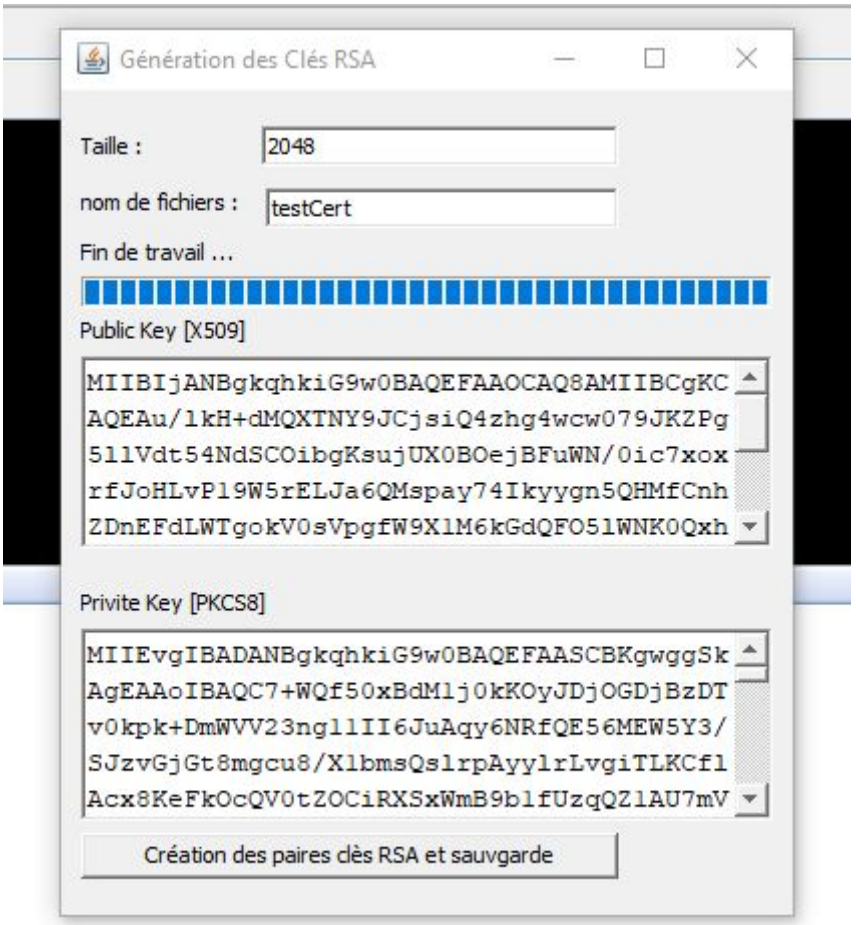
Interface pour la création d’un nouveau etudiant.



Interface pour la création d’un nouveau professeur



Génération de cryptosystème RSA

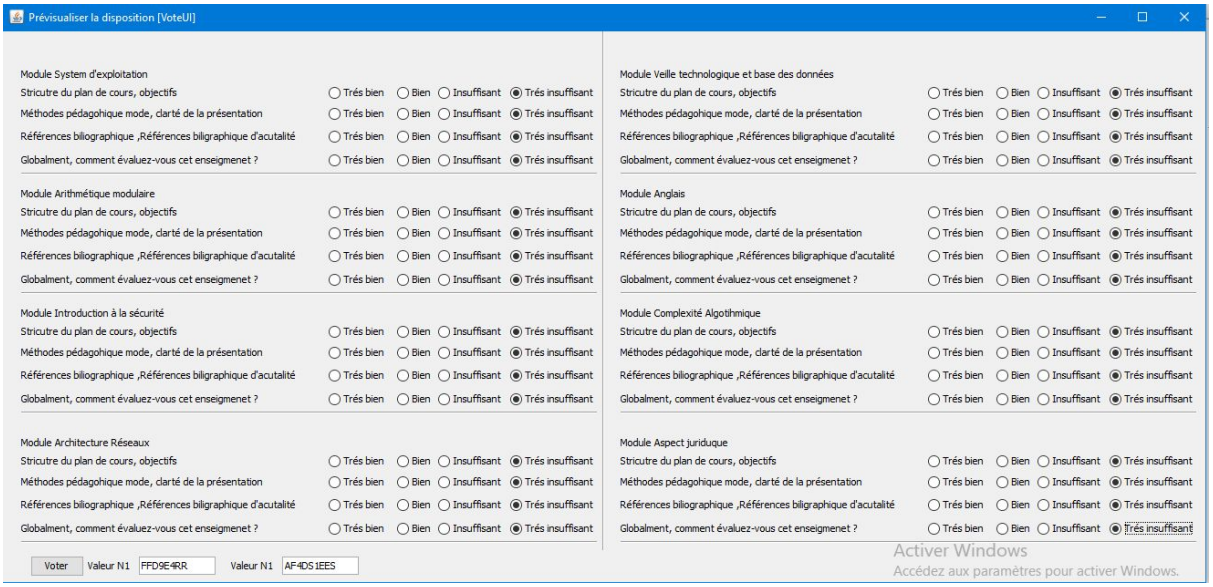


Exemple d'exécution du mécanisme de la signature aveugle :

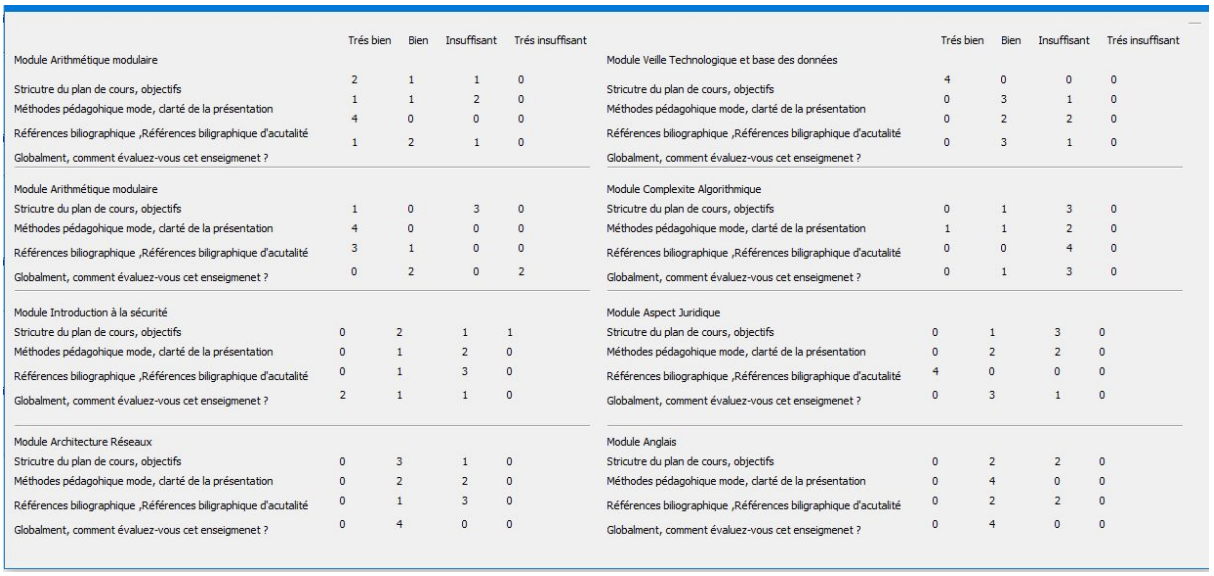
```
ALICE ----------------------------------------------
[*] Alice genère le crypto system RSA...
BOB ------------------------------------------------
[*] Le message en claire : usthb-ssi
le hasher de message en  SHA-256 (hex):
255b8e5187f68bd35deade3c37eb3ec317f3ce7cf6b022944662b60513032601
[+] la valeur numérique de hasher :
168973407265311962397546549051785989075528772602462373195986540479091176340 49
[+] le facteur R de masquage : 44793875943622201755262 8
[+] Le message masquer:
151870803324485965826680912657412341812496513586934371116265487471499093045437742995 0
635562202691297480243311833119056484697043009036663908809648448
[+] le message masquer en Hex :
1e67e24c8b984738c7986e905489c935e94c2665c8a1aea2f8745597858ffc74599b3cd82d2f801bbe5ea19c
7a7c56c301c725d66f73209b5c4bd112940
[*] Message envoyé à Alice ...
ALICE ----------------------------------------------
[*] Message reçu par Alice
151870803324485965826680912657412341812496513586934371116265487471499093045437742995 0
```

```
635562202691297480243311833119056484697043009036663908809648448
[+] Le message signer par Alice avec sa clé privé :
246628613934656394863830275240053848926495879761158531267918553372854128826052729430363989345656983731408051677597893471843240966447728703632643544906849 2
[+] Le message signer par Alice avec sa clé privé [Hex] :
2f16f49dfec6fe231babeec9828b197b4720df77057579e819c213441a3eb4db8cb13058a7f691f2a500617edbfee5be8f802408f0f747c0a9ec6c2c1221dbcc
BOB ----------------------------------------
[+] le messager signer reçu par Alice (masqué):
246628613934656394863830275240053848926495879761158531267918553372854128826052729430363989345656983731408051677597893471843240966447728703632643544906849 2
[+] le messager signer reçu par Alice (masqué) [Hex]:
2f16f49dfec6fe231babeec9828b197b4720df77057579e819c213441a3eb4db8cb13058a7f691f2a500617edbfee5be8f802408f0f747c0a9ec6c2c1221dbcc
[+] La signature après le démasquage :
348213740027123912268407281179281670228196652763358747066920568625100141180394358044486520444140340510172297953674917121397935443151023358914850071484830 3
[+] Vérification de la signature ...
Signature  :
348213740027123912268407281179281670228196652763358747066920568625100141180394358044486520444140340510172297953674917121397935443151023358914850071484830 3
Message decrypte with(e)     :  sign^e [N] =  :
168973407265311962397546549051785989075528772602462373195986540479091176340 49
Message decrypte with(e)     :  sign^e [N] [HEX]=  :
255b8e5187f68bd35deade3c37eb3ec317f3ce7cf6b022944662b60513032601
le hasher de message en  SHA-256 (hex):
255b8e5187f68bd35deade3c37eb3ec317f3ce7cf6b022944662b60513032601
Verification of signature completed successfully
Program executed in 478 milliseconds
```

Interface de vote pour étudiants



affichage résultat



	Trés bien	Bien	Insuffisant	Trés insuffisant
Module Arithmétique modulaire				
Stricutre du plan de cours, objectifs	2	1	1	0
Méthodes pédagohique mode, clarté de la présentation	1	1	2	0
Références biliographique ,Références biligraphique d'acutalité	4	0	0	0
Globalment, comment évaluez-vous cet enseigmenet ?	1	2	1	0

	Trés bien	Bien	Insuffisant	Trés insuffisant
Module Veille Technologique et base des données				
Stricutre du plan de cours, objectifs	4	0	0	0
Méthodes pédagohique mode, clarté de la présentation	0	3	1	0
Références biliographique ,Références biligraphique d'acutalité	0	2	2	0
Globalment, comment évaluez-vous cet enseigmenet ?	0	3	1	0

	Trés bien	Bien	Insuffisant	Trés insuffisant
Module Arithmétique modulaire				
Stricutre du plan de cours, objectifs	1	0	3	0
Méthodes pédagohique mode, clarté de la présentation	4	0	0	0
Références biliographique ,Références biligraphique d'acutalité	3	1	0	0
Globalment, comment évaluez-vous cet enseigmenet ?	0	2	0	2

	Trés bien	Bien	Insuffisant	Trés insuffisant
Module Complexite Algorithmique				
Stricutre du plan de cours, objectifs	0	1	3	0
Méthodes pédagohique mode, clarté de la présentation	1	1	2	0
Références biliographique ,Références biligraphique d'acutalité	0	0	4	0
Globalment, comment évaluez-vous cet enseigmenet ?	0	1	3	0

	Trés bien	Bien	Insuffisant	Trés insuffisant
Module Introduction à la sécurité				
Stricutre du plan de cours, objectifs	0	2	1	1
Méthodes pédagohique mode, clarté de la présentation	0	1	2	0
Références biliographique ,Références biligraphique d'acutalité	0	1	3	0
Globalment, comment évaluez-vous cet enseigmenet ?	2	1	1	0

	Trés bien	Bien	Insuffisant	Trés insuffisant
Module Aspect Juridique				
Stricutre du plan de cours, objectifs	0	1	3	0
Méthodes pédagohique mode, clarté de la présentation	0	2	2	0
Références biliographique ,Références biligraphique d'acutalité	4	0	0	0
Globalment, comment évaluez-vous cet enseigmenet ?	0	3	1	0

	Trés bien	Bien	Insuffisant	Trés insuffisant
Module Architecture Réseaux				
Stricutre du plan de cours, objectifs	0	3	1	0
Méthodes pédagohique mode, clarté de la présentation	0	2	2	0
Références biliographique ,Références biligraphique d'acutalité	0	1	3	0
Globalment, comment évaluez-vous cet enseigmenet ?	0	4	0	0

	Trés bien	Bien	Insuffisant	Trés insuffisant
Module Anglais				
Stricutre du plan de cours, objectifs	0	2	2	0
Méthodes pédagohique mode, clarté de la présentation	0	4	0	0
Références biliographique ,Références biligraphique d'acutalité	0	2	2	0
Globalment, comment évaluez-vous cet enseigmenet ?	0	4	0	0

Connection (Authentification de tous les utilisateurs