Rapport du projet « Encapsulation cryptographique dans un protocole de communication chiffré et signé »

# Principe

### Enregistrement auprès de l’autorité de Certification

**2) Génération des clés**

**Clé privée Alice**

**Clé publique Alice**



login: Alice

IP :

Clé pub :

**Clé de cession Alice**



**Clé publique AC**

**Autorité de Certification**

1. **Diffusion de la clé publique**

**(Clé privée Alice) chiffrée**

1. **Délivrance de la clé privée**

**Registre de certificats**

1. **Alice fait une demande de certificat**

**Alice**

1. Alice fait une demande de certificat auprès de l’Autorité de Certification (AC). Elle envoie sa demande accompagnée d’un jeton (soit une clé de cession, chiffrée avec la clé publique de l’AC de longueur 128 bits) pour permettre à l’AC de lui répondre.
2. L’Autorité de Certification génère une paire de clé (clé publique, clé privée). Ces clés sont des clés RSA de longueur 1024 bits, générées avec la méthode RSA\_generate\_key.
3. L’AC délivre la clé privée au demandeur, soit Alice. La clé privée est chiffré à l’aide de la clé de cession d’Alice et de l’algorithme Blowfish (méthode :  BF\_cfb64\_encrypt). La clé privée chiffrée est de taille 1024 bits.
4. L’AC publie le certificat dans son registre de certificats. Dans celui-ci, il y a le login, l’IP et la clé publique du demandeur de certificat.

### Alice envoie un message à Bob

### http://lesactualitesdudroit.20minutes-blogs.fr/media/00/02/165028087.jpg

1. **Création du message à envoyer à Bob**
2. **Déchiffrement et vérification clé publique Bob**

login: Alice

IP :

Clé pub :



1. **Renvoi clé publique Bob signée par le CA**

**Message (CertificatBob, signature CA )**

1. **Alice demande clé publique Bob**

**2) Récupération clé publique Bob**

**Clé privée Alice**

**Clé publique Alice**

**Clé privée AC**

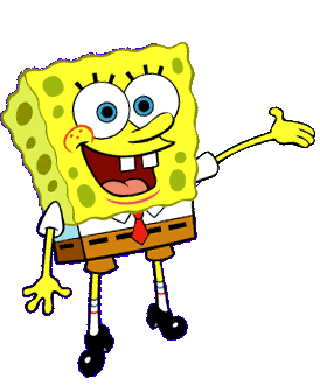
**Clé publique AC**

**Autorité de Certification**

**Registre de certificats**

**Alice**

1. **Envoi du message EnvXML**



**Clé de session Alice**

**Clé privée Bob**

**Clé publique Bob**

**Bob**

1. Alice contacte l’Autorité de Certification(AC) afin de récupérer les informations nécessaires pour communiquer avec Bob, via la méthode retrieveRemoteUser. Il s’agit de la clé publique de Bob ainsi que de son adresse IP. Nous avons en effet choisit d’inclure l’IP afin de faciliter la communication.
2. L’AC récupère le certificat de Bob dans son registre de certificats.
3. L’AC crée un message chiffré contenant le certificat de Bob (sa clé publique et son IP) et la signature de l’AC. Cette dernière est en fait le hash de la clé publique de Bob chiffrée avec la clé privée de l’AC. Il l’envoie à Alice.

**ATTENTION :** Les étapes 1 à 3 ne sont effectuées que pour la première communication, ensuite la clé publique de la personne avec qui on communique est stockée dans la classe Messenger qui se charge de la communication d’un utilisateur local (LocalUser) et d’un utilisateur distant (RemoteUser). Pas besoin donc d’aller demander cette clé à chaque fois.

1. Alice doit vérifier que le message vient bien de l’AC et contient bien la clé publique de Bob. Pour cela, elle déchiffre la signature de l’AC avec la clé publique de l’AC et obtient ainsi le hash de la clé publique de Bob. Elle compare ce hash avec la clé publique de Bob qu’elle hash. Ils doivent être égaux.
2. Alice chiffre son message avec une clé de session. Elle chiffre ensuite la clé de session avec la clé publique de Bob, elle a donc un jeton. Elle rédige ensuite le MsgXML, soit :

<MsgXml>

<Token> jeton </Token>

<MsgCrypt> message chiffrée avec la clé de cession</MsgCrypt>

</MsgXml>

Elle hash ensuite de MsgXML et chiffre ce hash avec sa clé privée, c’est ce qu’on appelle la signature. Elle rédige ensuite EnvXML, soit :

<EnvXml>

<From> emetteur </From>

<To> destinataire </To>

<Sign> signature </Sign>

<MsgXml>

<Token> jeton </Token>

<MsgCrypt> message chiffrée avec la clé de cession </MsgCrypt>

</MsgXml>

</EnvXml>

1. Alice envoie EnvXML à Bob via notre méthode sendTo.

### Bob reçoit le message d’Alice

1. **Création du message à envoyer à Bob**
2. **Déchiffrement et vérification clé publique Bob**

login: Alice

IP :

Clé pub :



**3) Récupération clé publique Bob**

**Clé privée Alice**

**Clé publique Alice**

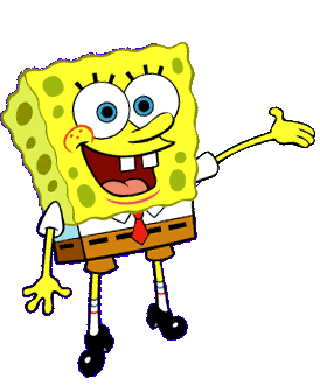
**Registre de certificats**

**Alice**

1. **Reçoit le message EnvXML**

**Autorité de Certification**

1. **Bob demande clé publique Alice**



1. **Analyse du message EnvXML pour lire le message envoyé par Alice**

**Message (CertificatAlice, signature CA )**

**Clé privée AC**

**Clé publique AC**

1. **Renvoi clé publique Alice signée par le CA**

**Clé privée Bob**

**Clé publique Bob**

**Bob**

1. Bob recoit EnvXML. Il récupère le nom de l’émetteur et veux sa clé publique pour déchiffrer la signature d’EnvXML.
2. Bob contacte l’Autorité de Certification(AC) afin de récupérer le certificat d’Alice pour voir si le message vient bien d’Alice et pour ainsi le déchiffrer.
3. L’AC récupère le certificat d’Alice dans son registre de certificats.
4. L’AC crée un message chiffré contenant le certificat d’Alice (sa clé publique et son IP) et la signature de l’AC. Cette dernière est en fait le hash de la clé publique d’Alice chiffrée avec la clé privée de l’AC. Il l’envoie à Bob.

**ATTENTION :** Les étapes 2 à 4 ne sont effectuées que pour la première communication, ensuite la clé publique de la personne avec qui on communique est stockée dans la classe Messenger qui se charge de la communication d’un utilisateur local (LocalUser) et d’un utilisateur distant (RemoteUser). Pas besoin donc d’aller demander cette clé à chaque fois.

1. Bob déchiffre la signature d’EnvXML à l’aide de la clé publique d’Alice et récupère ainsi un hash de MsgXML. Il hashe le MsgXML d’EnvXML et le compare au hash obtenu en déchiffrant la signature. Uniquement s’ils sont identiques, le processus continue. Bob déchiffre ensuite le jeton de MsgXML avec sa propre clé privée, il obtient ainsi la clé de session. A l’aide de cette dernière, il déchiffre le message chiffré et obtient ainsi le message que lui a envoyé Alice.

# Protocole de communication

# Optimisations proposées

Notre programme pourrait encore être amélioré, nous avons pensé à différentes possibilities tells que :

-Fixer un temps d’expiration aux certificat, c’est-à-dire que passé ce délai d’expiration, l’Autorité de Certification élimine le certificat expiré de son registre de certificat. L’utilisateur devra donc faire une nouvelle demande de certificat pour une prochaine communication.

- Vérification que le demandeur de clé privé a bien recu la réponse de l’AC contenant ladite clé privée produite