Etude de protocoles d'authentification (partie 3)

Dans ce TD, on va étudier les systèmes d'authentification basés sur un serveur de confiance qui permettent à deux participants *A* (Alice) et *B* (Bob) de s'authentifier et de dériver une clé de session commune alors qu'ils ne possèdent à priori aucun secret en commun. Soit *S* le serveur d'authentification qui est connu d'Alice et Bob et qui peut être considéré comme un tiers de confiance. Dans ce modèle, chaque participant possède une clé secrète qui est seulement connue de lui ainsi que du serveur d'authentification (il n'y a pas de clé secrète générée pour chaque paire de participants). Ainsi K_{AS} est la clé symétrique connue seulement de *A* et *S* et K_{BS} la clé symétrique connue seulement de *B* et *S*.

Protocole 1
Dans le protocole suivant, on dénote par N_A et N_B les nonces (c'est-à-dire des chaînes aléatoires spécifiques à la session) générés respectivement par Alice et Bob. Le but du protocole est d'assurer une authentification mutuelle entre Alice et Bob et de générer une clé de session commune K_{AB} qui pourra être ensuite utilisée pour sécuriser les communications futures. On suppose aussi qu'Alice est celle qui initie le protocole.

1. $A \rightarrow S : A, B, N_A$
Alice annonce au serveur qu'elle souhaite communiquer avec Bob et qu'elle a généré le nonce N_A pour cette session.
2. $S \rightarrow A : \{N_A, K_{AB}, B, \{K_{AB}, A\}K_{BS}\}K_{AS}$
Le serveur envoie à Alice un message chiffré avec sa clé secrète K_{AS} qui contient la clé de session K_{AB} qu'il a générée ainsi que le nonce N_A et l'identité de *A* ainsi qu'un message chiffré avec la clé de K_{BS}.
3. $A \rightarrow B : \{K_{AB}, A\}K_{BS}$
Alice fait suivre à Bob le message contenant la clé de session qui est chiffré avec la clé qu'il partage avec le serveur. Bob déchiffre ce message pour récupérer la clé de session.
4. $B \rightarrow A : \{N_B\}K_{AB}$
Bob envoie à Alice un nonce qui est chiffré avec la clé de session K_{AB}.
5. $A \rightarrow B : \{N_B - 1\}K_{AB}$
Alice décrémente la valeur du nonce, chiffre la valeur résultante avec sa clé de session et l'envoie à Bob. Bob vérifie ensuite la valeur reçue et authentifie Alice si cette valeur est correcte.

Questions :
-À quoi cela sert-il de mettre le nonce N_A et l'identité *B* dans le message ?
-Pourquoi est-ce que c'est OK pour la sécurité du protocole que le serveur génère la clé de session K_{AB}?
-Supposons qu'un attaquant est réussi à s'introduire dans la mémoire de l'ordinateur de *A* et a volé un ancien message $\{K_{AB}, A\}K_{BS}$ ainsi que la clé K_{AB} associée à ce message. Décrivez comme l'attaquant peut utiliser cette information pour faire une attaque par rejeu.
-Comment peut-on se prémunir contre cette attaque par rejeu ?

Protocole 2

Ce protocole se base sur un algorithme de chiffrement et de signature à clé publique. Plus précisément, K_{PA} et K_{SA} sont respectivement la clé publique de chiffrement et la clé secrète de déchiffrement d'Alice (de même pour Bob avec K_{PB} et K_{SB}) alors que K_{SS} est la clé secrète de signature pour le serveur et K_{PS} la clé publique de vérification de la signature.

1. $A \rightarrow S : A, B$
Alice demande la clé publique de Bob au serveur.
2. $S \rightarrow A : \{K_{PB}, B\}K_{SS}$
Le serveur répond à Alice avec la clé publique de Bob en même temps que son identité, le tout étant signé par la clé du serveur.
3. $A \rightarrow B : \{N_A, A\}K_{PB}$
Alice choisit un nonce, le chiffre avec la clé publique de Bob avant de lui envoyer.
4. $B \rightarrow S : B, A$
Bob demande la clé publique d'Alice au serveur.
5. $S \rightarrow B : \{K_{PA}, A\}K_{SS}$
Le serveur répond à Bob avec la clé publique d'Alice en même temps que son identité, le tout étant signé par la clé du serveur.
6. $B \rightarrow A : \{N_A, N_B\}K_{PA}$
Bob choisit un nonce, le chiffre avec la clé publique de Alice en même temps que le nonce que celle-ci a généré afin de prouver qu'il a pu déchiffrer le message envoyé à l'étape 3.
7. $A \rightarrow B : \{N_B\}K_{PB}$
Alice envoie N_B à Bob pour prouver qu'elle a pu déchiffrer le message envoyé à l'étape précédente.

Questions :

-Expliquez pourquoi ici on pourrait dire que le serveur d'authentification joue l'équivalent du rôle d'une autorité de certification qui peut sur demande produire des certificats contenant l'identité et la clé publique d'un participant ?
-Une fois que le protocole est terminé, comment est-ce qu'Alice et Bob peuvent générer une clé de session partagée à partir des deux nonces N_A et N_B ?
-Proposez une attaque de type homme-du-milieu sur ce protocole où un imposteur *I* qui a réussi à convaincre *A* d'initier une session avec lui peut utiliser cela pour réussir à convaincre *B* qu'il discute avec *A* dans une autre session.
-Proposez un changement au protocole permettant de contrer cette attaque.

Protocole 3

Dans ce protocole, *M* représente un identifiant spécifique à la session.

1. $A \rightarrow B : M, A, B, \{N_A, M, A, B\}K_{AS}$
2. $B \rightarrow S : M, A, B, \{N_A, M, A, B\}K_{AS}, \{N_B, M, A, B\}K_{BS}$
3. $S \rightarrow B : M, \{N_A, K_{AB}\}K_{AS}, \{N_B, K_{AB}\}K_{BS}$
4. $B \rightarrow A : M, \{N_A, K_{AB}\}K_{AS}$

Questions :

-Traduisez le protocole tel qu'il est écrit en notation de protocole de sécurité dans des phrases en langue naturelle expliquant son déroulement (un peu comme cela était fait pour les deux premiers protocoles).

-Expliquez en quoi l'ordre des interactions entre *A*, *B* et *S* est différent pour ce protocole que pour les deux premiers protocoles.

-Décrivez une attaque par rejeu contre le protocole.

-Essayez de trouver une attaque par laquelle l'adversaire réussit à se débrouiller pour que *A* et *B* finissent par avoir une clé différente à la fin du protocole sans que ceux-ci ne s'en rendent compte.