Protocole du groupe void

Tristan Claverie, Maxime Cadoret, Lucie Labadie

Le protocole défini est le suivant:

$A \to S : \{TrID, A, B, pub_A, mac(< TrID, A, B, pub_A >, K_{as})\}$
$S \to B : \{TrID, A, pub_A, mac(< TrID, A, pub_A >, K_{bs})\}$
$B \to S : \{A, senc(< TrID, pub_B >, K_{bs}), mac(< TrID, A, pub_B >, K_{bs})\}$
$S \to A : \{B, senc(< TrID, pub_B >, K_{as}), mac(< TrID, B, pub_B >, K_{as})\}$
$A \to B : \{< TrId, K >\}_{pub_B}$
$B \to A : \{h(< TrId, K >)\}_{pub_A}$

Connaissances initiales : On suppose que l'agent S (Serveur) a connaissance de K_{as} et K_{bs} et les partage respectivement avec A et B au début du protocole. Le serveur est considéré comme un agent de confiance.

On définit la fonction $mac(msg, K)$ comme étant $senc(h(msg), K)$. Sa fonction de coût suit donc les mêmes particularités:
$cost(mac(msg, K)) = cost(senc(h(msg), K))$
$cost(mac(N_a, K)) = 1 + 5 + 1 + 1 = 8$
$cost(mac(< N_a, A, N_b, B >, K)) = 1 + 5 + 1 + 1 + 1 + 1 + 1 = 11$

Valeurs générées : $TrID$, pub_A, $priv_A$, pub_B, $priv_B$, K sont des valeurs générées à chaque instance du protocole.

Description du protocole : Au début du protocole Axel génère un nouveau couple de clé pub_A et $priv_A$ ainsi qu'un $TrID$ non utilisés. Il envoie à S la paire $TrID$, son identité, l'identité de B, sa clé publique et le mac de ce message chiffré avec K_{as}.
Le serveur vérifie l'intégrité des données et la fraicheur du $TrID$. Si l'intégrité du message est douteuse ou le $TrID$ déjà utilisé, le serveur interrompt le protocole. Dans le cas contraire, le serveur envoie à Bernard la paire $TrID$, l'identité de A, la clé publique de A et le mac du message chiffré avec K_{bs}.
Bernard vérifie l'intégrité et la fraicheur du $TrID$. En cas de problème, il stoppe le protocole. B génère un nouveau couple de clé pub_B et $priv_B$ qui n'a jamais été utilisé. Il chiffre la paire $TrID$ et sa clé publique avec K_{bs}. Il envoie au serveur l'identité de A, le message chiffré et un mac du $TrID$, de l'identité de A et de pub_B chiffré avec K_{bs}.

Le serveur déchiffre la partie chiffrée et vérifie l'intégrité des éléments du messages avec le mac. Il vérifie aussi que la transaction correspondant au $TrID$ prend bien place entre A et B. S'il ne rencontre pas de problèmes, S chiffre la paire $TrID$ et clé publique de B avec K_{as}. Il envoie à A un message contenant l'identité de B, le message chiffré et un mac du $TrID$, de l'identité de B et de pub_B chiffré avec K_{as}.
A vérifie qu'il attend bien ce message pour cette transaction à l'aide du $TrID$ et vérifie l'intégrité des données déchiffrées à l'aide du mac. S'il ne rencontre pas de problème, il génère une clé K et envoie à B la paire $TrID$ et K, chiffrés avec pub_B.
B déchiffre le message et vérifie que le $TrID$ et son couple de clés appartiennent bien à la même transaction. Si c'est le cas, il hash la paire $TrID$ et K puis le chiffre avec pub_A. A déchiffre le message et vérifie que le hash correspond aux données envoyées.

Propriétés du protocole :
Authentification Lorsque Bernard reçoit le dernier message, il est sûr que celui-ci vient d'Axel. Lorsque Axel reçoit le dernier message, elle est sûre que celui-ci vient de Bernard.
Confidentialité Les deux agents Bernard et Axel sont les seuls à connaitre le clé publique de Bernard pub_B et la clé de chiffrement K.

Poids du protocole: 221
Message 1 : $1+1+1+1+1+5+1+1+1+1+1=15$
Message 2 : $1+1+1+1+5+1+1+1+1=13$
Message 3 : $1+1+50+1+1+1+1+5+1+1+1+1=65$
Message 4 : $1+1+50+1+1+1+1+5+1+1+1+1=65$
Message 5 : $1+50+1+1+1$
Message 6 : $1+5+1+1+1=9$
Total : $15+13+65+65+54+9=221$