UE ISEC

Corrigé TD Authentification

Jean Leneutre

CORRECTION EXERCICE 1

a- 1 Les messages 1 et 2 ne font intervenir que l'autorité de certification AC et le client C : il s'agit pour le client de récupérer la clef publique de S, PKs. Avant 1'envoi du message 1, C génère et stocke un nombre pseudo-aléatoire N_1 qui jouera le rôle de défi pour AC. AC interprète le message 1 comme une requête par C de la clef publique de S, et y répond dans le message 2. À la réception du message 2, C déchiffre la partie chiffrée du message avec PKac, et vérifie que la valeur du champ N_1 correspond à celle précédemment stockée. Cela permet à C d'authentifier AC, de récupérer PKs.
De même les messages 4 et 5 permettent à S de récupérer la clef publique de S, PKs.

a-2 Les trois différentes méthodes d'authentification utilisées dans ce protocole dont :

- messages 1 et 2 (resp. messages 4 et 5) : un chiffrement avec la clef privée (correspondant à une signature) d'un défi envoyé dans un message précédent
- message 3 : un chiffrement avec la clef privée (correspondant à une signature) d'une estampille (time stamp)
- message 3 et message 6 : le déchiffrement d'un défi chiffré avec la clef publique.

a-3 C est authentifié par S après le message 5 : en effet C doit attendre de récupérer la clef PKs pour traiter le message 3.

a-4 le nonce N_2 est chiffré avec PKc dans le message 6 pour assurer sa confidentialité entre C et S. Ce chiffrement n'est pas nécessaire pour assurer l'authentification de S par C.

Une alternative possible consiste à utiliser un autre nonce N_3 :

1. $C \rightarrow AC$: C, S, N_1
2. $AC \rightarrow C$: $AC, \{AC, C, N_1, PKs\}_{SKac}$
3. $C \rightarrow S$: $C, S, \{C, T, L, \{ N_2, N_3\}_{PKs}\}_{SKc}$
4. $S \rightarrow AC$: S, C, N_3
5. $AC \rightarrow S$: $AC, \{AC, S, N_3, PKc\}_{SKac}$
6. $S \rightarrow C$: S, C, N_3+1

Après un déroulement complet du protocole, N_2 est toujours utilisé par C et S comme une clef symétrique afin de sécuriser leurs communications.

a-5 Il s'agit d'un protocole de transport de clef, car la clef de session (N_2) est généré par un des participants (C), puis envoyer à l'autre participant (S). Pour obtenir un protocole de mise en accord sur la clef, les deux participants (C et S) doivent contribuer à la génération de la clef. Pour cela on peut ajouter un nouveau nonce N_4 dans le message 6 :

1. $C \rightarrow AC$: C, S, N_1
2. $AC \rightarrow C$: $AC, \{AC, C, N_1, PKs\}_{SKac}$
3. $C \rightarrow S$: $C, S, \{C, T, L, \{N_2\}_{PKs}\}_{SKc}$
4. $S \rightarrow AC$: S, C, N_3
5. $AC \rightarrow S$: $AC, \{AC, S, N_3, PKc\}_{SKac}$

6. S → C : S, C, $\{S, N_2+1, N_4\}_{PKc}$

À la fin du protocole, S et C calculent : $K = h(N_2 \| N_4)$.

a-6 Le protocole SPLICE/AS ne satisfait pas PFS car la connaissance de SKs et de SKc permet à un attaquant de déchiffrer tous les nonces N_2 distribués dans les sessions précédentes.
Pour obtenir un protocole de mise en accord sur la clef, on peut s'inspirer du protocole Diffie-Hellman. On suppose que C et S connaissent un générateur g commun, et que x_c (resp. x_s) est un secret généré à chaque session par C (resp. S) :

1. C → AC : C, S, N_1
2. AC → C : AC, $\{AC, C, N_1, PKs\}_{SKac}$
3. C → S : C, S, $\{C, T, L, \{g^{Xc}\}_{PKs}\}_{SKc}$
4. S → AC : S, C, N_3
5. AC → S : AC, $\{AC, S, N_3, PKc\}_{SKac}$
6. S → C : S, C, $\{g^{Xc}, g^{Xs}\}_{PKc}$

À la fin du protocole, C et S calculent $K = g^{XcXs}$.

b- X peut se faire passer pour S auprès de C de la manière suivante

1. C → X/AC : C, S, N_1
1'. X/C → AC : C, X, N_1
2. AC → C : AC, $\{AC, C, N_1, PKx\}_{SKac}$
3. C → X/S : C, S, $\{C, T, L, \{N_2\}_{PKx}\}_{SKc}$
4. X → AC : X, C, N_3
5. AC → X : AC, $\{AC, X, N_3, PKc\}_{SKac}$
6. X/S → C : S, C, $\{S, N_2+1\}_{PKc}$

c- Le problème provient du fait que dans le message 2, C n'est pas assuré que la clef publique est celle de S. Il suffit de rajouter l'identifiant de S dans le « certificat » du message 2 :

1. C → AC : C, S, N_1
2. AC → C : AC, $\{AC, C, N_1, \mathbf{S}, PKs\}_{SKac}$
3. C → S : C, S, $\{C, T, L, \{N_2\}_{PKs}\}_{SKc}$
4. S → AC : S, C, N_3
5. AC → S : AC, $\{AC, S, N_3, PKc\}_{SKac}$
6. S → C : S, C, $\{S, N_2+1\}_{PKc}$

Avec ce nouveau protocole, l'attaque précédente n'est plus valide : après avoir reçu le message 2, le client vérifie que dans le « certificat », $\{AC, C, N_1, X, PKx\}_{SKac}$, l'identifiant précédant la clef publique correspond bien à celui de S. Dans le cas de l'attaque précédente, le serveur obtient un identifiant différent et stoppe le déroulement du protocole.

d- Le problème du message 3 étant le même dans le message 5, il s'agit de l'attaque duale :

1. X → AC : X, S, N_1
2. AC → X : AC, $\{AC, X, N_1, S, PKS\}_{SKac}$

3. X/C → S : C, S, $\{C, T, L, \{N_2\}_{PKs}\}_{SKx}$
4. S → X/AC : S, C, N_3
4'. X/S → AC : S, X, N_3
5. AC → S : AC, $\{AC, S, N_3, PKx\}_{SKac}$
6. S → X/C : S, C, $\{S, N_2+1\}_{PKx}$

Comme précédemment, il suffit de rajouter l'identifiant de C dans le « certificat » du message 5 :

1. C → AC : C, S, N_1
2. AC → C : AC, $\{AC, C, N_1, S, PKs\}_{SKac}$
3. C → S : C, S, $\{C, T, L, \{N_2\}_{PKs}\}_{SKc}$
4. S → AC : S, C, N_3
5. AC → S : AC, $\{AC, S, N_3, \mathbf{C}, PKc\}_{SKac}$
6. S → C : S, C, $\{S, N_2+1\}_{PKc}$

e- L'attaquant intercepte le message 3, récupère le nonce chiffré pour construire un nouveau message 3' qu'il envoie au serveur sous sa propre identité. Ensuite X peut construire et envoyer un message 6 correct à C, à partir du message 6' qu'il a reçu du serveur.

3. C → X/S : C, S, $\{C, T, L, \{N_2\}_{PKs}\}_{SKc}$
3'. X → S : X, S, $\{X, T, L, \{N_2\}_{PKs}\}_{SKx}$
6'. S → X : S, X, $\{S, N_2+1\}_{PKx}$
6. X/S → C : S, C, $\{S, N_2+1\}_{PKc}$

f- Le problème vient du fait que le serveur est trompé sur l'origine du nonce chiffré qu'il récupère dans le message 3. Il suffit donc de modifier le message 3 :

3'. C → S : C, S, $\{C, T, L, \{\mathbf{C}, N_2\}_{PKs}\}_{SKc}$

g- On suppose que l'on chiffre par blocs de longueur k :

- le message M est divisé en blocs de longueur k, M1, M2, …,Mn (en complétant le cas échéant le dernier bloc) et le chiffre final est obtenu en concaténant les chiffres des blocs, par exemple, $\{M\}_{PK}=\{M1\}_{PK}\ldots\ \{Mn\}_{PK}$
- les identifiants sont codés sur une longueur p*k.

Sous ces hypothèses, l'attaquant récupère dans le message 3,
$\{C, N_2\}_{PKs}= \{C1\}_{PKs} \ldots\ \{Cp\}_{PKs}\{N_{21}\}_{PKs}\ldots\ \{N_{2m}\}_{PKs}$
Il remplace les premiers blocs (chiffrant C) par $\{X\}_{PKs}$ (qu'il peut calculer), et construit le message 3'

3. C → X/S : C, S, $\{C, T, L, \{C,N_2\}_{PKs}\}_{SKc}$
3'. X → S : X, S, $\{X, T, L, \{X,N_2\}_{PKs}\}_{SKx}$
6'. S → X : S, X, $\{S, N_2+1\}_{PKx}$
6. X/S → C : S, C, $\{S, N_2+1\}_{PKc}$