TP : vérification de protocoles cryptographiques

On utilise Avispa, outil à installer dans votre répertoire (voir chamillo).

Il prend en entrée (premier paramètre) un fichier écrit dans le langage HLPSL. Pour ce TP, vous n'écrirez pas vous-même des protocoles en langage HLPSL, on vous fournit des descriptions dans les fichiers du répertoire Sources du TP.

Par exemple, pour lancer Avispa sur le fichier NSPK_1.hlpsl :

```
avispa NSPK_1.hlpsl --output=. --ofmc
```

ou bien créer un répertoire différent pour output si vous ne souhaitez pas que les fichiers résultats soient mis dans le répertoire courant. L'option `ofmc` définit quel est l'outil de vérification (« model-checker ») qu'on utilise pour cette session.

Vous pouvez exécuter `avispa --help` pour avoir plus d'options. Noter qu'Avispa commence par traduire le format HLPSL (High Level Protocol Specification Language) en une forme intermédiaire (.if pour Intermediate Format) qui apparaîtra dans votre répertoire output. Il exécute ensuite OFMC pour vérifier si les propriétés exprimées dans la partie « `goal` » du fichier HLPSL sont satisfaites et vous donne un compte-rendu (sur la sortie standard).

Pour chaque question décrire les résultats fournis par l'outil Avispa. Si une attaque est trouvée on la dessinera en précisant les actions et le calcul fait par l'attaquant.

Exercice 1 :

Objectif : comprendre la modélisation, les attaques sur secret et sur authentification, le rôle des différentes sessions.

Soit le protocole suivant (Needham-Schroeder)

A pour Alice – Bob pour Bob. ka et kb sont respectivement les clés publiques de A et B. On notera inv(ka) la clé privée de A. Na et Nb sont des nonces (nombres aléatoires uniques).

1. A -> B: {Na . A}_kb
2. B -> A: {Na . Nb}_ka

3. A -> B: {Nb}_kb

L'opérateur . est la concaténation de messages.

Question 1 (sur papier) :

- Ce protocole se veut garantir l'authentification de A auprès de B et réciproquement. Justifier informellement comment fonctionne cette double authentification.
- Si Na (respectivement Nb) n'est pas secret expliquer pourquoi l'authentification n'est plus garantie.

Question 2 :

On considère dans un premier temps une session unique entre A et B.

session(A, B, ka, kb)

En d'autres termes, il n'y a qu'Alice et Bob, pas d'intrus présent, et nos deux interlocuteurs n'exécutent le protocole qu'une seule fois. Ceci est exprimé dans la partie « environment » du fichier HLPSL (il n'y a qu'Alice et Bob dans cet environnement).

On s'intéresse aux 4 propriétés suivantes (qu'on exprimer dans la partie « goal » à la fin du fichier HLPSL) :

- le nonce Na généré par A reste secret entre A et B : secrecy_of secret_na
- le nonce Nb généré par B reste secret entre A et B : secrecy_of secret_nb
- le nonce Na sert à authentifier B auprès de A (c'est bien B qui répond à A) : authentication_on bob_alice_na
- le nonce Nb sert à authentifier A auprès de B (c'est bien A qui répond à B) : authentication_on alice_bob_nb

Lancer avispa sur le fichier NSPK_1.hlpsl. Interpréter les résultats fournis

Question 3 :

On considère maintenant 3 sessions possibles en parallèle :

session(A, B, ka, kb) /\ session(A, I, ka, ki) /\ session(I, B, ki, kb)

Une session entre A et B, une session entre A et I et une session entre I et B, I étant l'intrus. Noter que A parle à B, mais aussi, volontairement à I en utilisant la clé de I, et de même I avec B. Ce n'est donc pas exactement une attaque MitM où A et B ne seraient pas conscients de la présence d'un intrus furtif.

On s'intéresse uniquement aux 2 propriétés de secret. Est-ce que le secret entre 2 acteurs du trio reste bien gardé du troisième membre du trio ?
Réfléchissez d'abord en essayant des scénarios sur papier. Ensuite, pour confirmer :
lancez avispa sur le fichier NSPK_2.hlpsl. Interpréter les résultats fournis

Question 4 :

On garde les mêmes sessions que la question précédente mais on s'intéresse aux propriétés d'authentification.

Lancer avispa sur le fichier NSPK_3.hlpsl. Interpréter les résultats fournis

Question 5 :

Proposer une correction du protocole qui permet d'assurer les propriétés de secret et d'authentification.

- Copier le fichier NSPK_3.hlpsl en NSPK_4.hlpsl
- Modifier le protocole en début de fichier (version en commentaire) : imaginez une façon d'empêcher l'intrus de tromper Alice et Bob
- Encoder cette modification dans les rôles bob et alice
- Vérifier avec Avispa si votre modification aboutit à un protocole sûr qui respecte les propriétés de secret et l'authentification mutuelle.

Exercice 2 :

On s'intéresse à une variante de la version corrigée de Nedham-Schroeder dans laquelle on utilise un xor bit à bit. Cette variante peut être vue comme plus « légère » car on chiffre un message plus petit.

1. A -> B: {Na, A}_kb
2. B -> A: {Nb, Na xor B}_ka
3. A -> B: {Nb}_kb

Question 1 :
L'opération xor entre deux bits est définie de la manière suivante :
0 xor 0 = 0 0 xor 1 = 1 1 xor 0 = 1 1 xor 1 = 0
C'est en fait l'addition modulo 2, mais on peut aussi la voir comme le OU exclusif entre des booléens codés par 0 pour Faux et 1 pour Vrai (d'où son nom : eXclusive OR).
Pour deux suites de bits (comme le sont les nonces et les identités), c'est-à-dire deux nombres binaires, le xor se fait bit à bit, comme une addition sans retenue (puisque modulo 2 pour chaque bit).

Montrez les propriétés suivantes (où b représente un seul bit) :

- 0 xor b = b
- b xor b = 0

Question 2 :
Le fichier XorNSPK_1.hlpsl contient la description du protocole ci-dessus avec la propriété de secret de Na.
Ici `Na` est déclaré de type `message` et peut donc accepter n'importe quelle valeur.
Vérifier le protocole à l'aide d'Avispa

Question 3 :
Le fichier XorNSPK_2.hlpsl étend le fichier précédent avec la propriété de secret de Nb.
Vérifier le protocole à l'aide d'Avispa et expliquer les résultats obtenus.

Question 4 :
Copier le fichier XorNSPK_2.hlpsl puis modifier la déclaration du type de Na : remplacer message par `text` (qui est le vrai type des nonces, Avispa vérifie le bon typage des messages) dans les processus de A et B.
Vérifier le protocole à l'aide d'Avispa et expliquer les résultats obtenus.

Exercice 3 :
On considère le protocole suivant :
1. A -> B: A,{Na}_kb
2. B -> A: B,{Na . Nb}_ka
3. A -> B: {zero. Msg}_(Na.Nb)
4. B -> A: {one . Msg}_(Na.Nb)

Question 1 :
Ce protocole vise à échanger un message de manière sécurisé après échange d'une clé de session. Expliquer précisément comment fonctionne ce protocole.

Question 2 :
Ouvrir le fichier A.hlpsl. La propriété testée est le secret de la clé de session générée. Vérifier le protocole à l'aide de Avispa et expliquer le résultat obtenu.

Question 3 :
Le fichier B.hlpsl contient une version légèrement modifiée du pas 2. On a remplacé :

2. B -> A: B,{Na.Nb}_Ka

par

2. B -> A: B,{Nb}_Ka

Vérifier ce protocole à l'aide d'Avispa et expliquer le résultat obtenu.

Question 4 :
Le fichier C.hlpsl contient le même protocole que **B.hlpsl** mais à présent on teste aussi l'authentification mutuelle entre A et B.

Analyser le protocole avec Avispa. Que se passe-t-il ?

Question 5
Proposer une correction de ce protocole

Exercice 4 : Protocole Secure RPC

Soit le protocole suivant :

1 A --> B : A, {Na}_|Kab|

2 B --> A : {Succ(Na), Nb}_|Kab|

3 A --> B : {Succ(Nb)}_|Kab|

4 B --> A : {Kpab,Nbp}_|Kab|

% A ajouter question 3

%5 A --> B : {Nap}_|Kpab|

%6 B --> A : {Succ(Nap)}_|Kpab|

Ce protocole permet d'échanger une clé de session Kpab et le nombre nbp qui pourra être utilisé dans la suite du protocole (partie handshake d'un protocole de transport). On veut donc lier la clé, A et B pour la suite du protocole.

En plus de l'authentification classique mutuelle entre A et B pour l'échange de clé on veut par la suite établir une authentification entre A et B sur la clé partagée.

Question 1 :

Récupérer et étudier le fichier **AS RPC 1 W.hlpsl**. Vérifier les propriétés de secret et d'authentification mutuelle sur le partage de la clé.

Question 2 :

On s'intéresse maintenant à l'authentification de B auprès de A sur la clé Kpab. L'authentification faible revient à vérifier qu'avant un request il y a un witness correspondant . L'authentification forte vérifie par contre que pour chaque request il existe un witness qui lui correspond de manière unique. Quelles sont les propriétés vérifiées ?

On pourra commenter/décommenter les propriétés dans la partie goal.

Question 3 :

Ajouter et traduire dans le fichier HLPSL les étapes suivantes à la fin du protocole:

A -> B : {Nap}_|Kpab|
B -> A : {Succ(Nap)}_|Kpab|

Vérifier l'authentification forte utilisant Kpab (of responder to initiator) pour ce nouveau protocole. Expliquer le résultat obtenu