TP Conception de protocoles de sécurité :
Comprendre les enjeux et l'intérêt des outils de validation par la pratique[‡]

Saïd Ider, Maryline Laurent, Maktoum Gaba, Alan Van Trigt
SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris
91120 Palaiseau, France

2026

Abstract

Dans un monde de plus en plus connecté et sujet à des cyber attaques, la sécurité des communications est devenue un enjeu majeur et les protocoles de sécurité en sont une brique essentielle. La toute première étape menant à l'utilisation sur Internet d'un protocole de sécurité sûr consiste à concevoir un tel protocole et à s'assurer qu'il satisfait aux propriétés de sécurité attendues. Cette étape est décisive, car un protocole mal conçu aboutit forcément au déploiement sur Internet d'un protocole vulnérable et à des attaques de types usurpations d'identité, Man in The Middle...

Dans ce TP, vous découvrirez de manière pratique les enjeux et les difficultés à concevoir un protocole cryptographique (ou protocole de sécurité) pour protéger des échanges numériques, en assurant les propriétés de confidentialité des messages, d'authentification de l'origine des messages, et d'authentification mutuelle des entités. Il vous permettra de consolider de manière autonome vos connaissances en cryptographie appliquée, de comprendre en particulier l'importance des nonces et leur intégration combinée à des mécanismes de chiffrement pour construire un protocole et de savoir détecter à un premier niveau les failles d'un protocole de sécurité. Vous utiliserez l'outil SPAN & AVISPA ("a Security Protocol ANimator for AVISPA") de vérification formelle de protocoles de sécurité. Vous serez en mesure de visualiser grâce à l'interface graphique de SPAN les failles d'un protocole de sécurité exploitables par un attaquant.

Ce TP s'adresse à des étudiants de niveau licence ou mastère, ayant des notions en informatique et des notions basiques en sécurité. Il nécessite le téléchargement d'une machine virtuelle (VM pour Virtual Machine) de 740MB. Il est calibré pour être réalisé en 3 heures.

1 Objectifs pédagogiques

Les objectifs sont les suivants :

- Identifier les méthodes de bases d'attaque et de défense dans les protocoles cryptographiques,
- Savoir distinguer la cryptographie symétrique et la cryptographie asymétrique et comprendre leur apport dans la construction d'un protocole de sécurité,
- Comprendre l'importance des nonces et des mécanismes de chiffrement dans la conception des protocoles de sécurité, ainsi que leur intégration combinée dans un protocole,
- Savoir détecter à un premier niveau les failles d'un protocole de sécurité,
- Appréhender les enjeux cruciaux lors de la conception d'un protocole de sécurité,
- Comprendre l'intérêt et les limites des outils de vérification formelle tels que SPAN & AVISPA.

2 Organisation du TP et votre mission

Le TP comprend les étapes suivantes :

- Les risques liés à la transmission d'un secret en clair sur un canal de communication,
- Le chiffrement symétrique pour une transmission confidentielle du secret,

[*]TP réalisé dans le cadre du projet TCE (Train-Cyber-Expert) du PEPR Cybersécurité avec le soutien financier de France2030

- Le chiffrement asymétrique pour une transmission confidentielle du secret,
- Le chiffrement asymétrique pour authentifier l'origine d'un message, c'est-à-dire s'assurer qu'un message provienne de l'émetteur déclaré,
- L'ajout des nonces aux messages pour empêcher les attaques par rejeu et assurer grâce au chiffrement asymétrique l'authentification unidirectionnelle (d'un seul interlocuteur) et l'authentification mutuelle (des deux interlocuteurs),
- Analyse du protocole Needham-Schroeder (NSPK) qui permet une authentification mutuelle et repose sur l'échange confidentiel de nonces,
- Analyse de l'attaque Man in the Middle sur le protocole NSPK et de sa solution (Lowe),
- Compilation des méthodes vues précédemment dans un protocole formel pour assurer une authentification mutuelle.

Votre mission est d'aider les protagonistes du TP Alice et Bob à concevoir un protocole cryptographique le plus sûr possible, c'est-à-dire qui satisfait une ou plusieurs propriétés de sécurité classiques : la confidentialité d'un secret et l'authentification mutuelle.

3 Quelques mots sur l'outil SPAN & AVISPA utilisé dans le TP et leurs limitations

SPAN & AVISPA intègre plusieurs outils de vérification formelle pour vérifier qu'un protocole cryptographique satisfait les propriétés de sécurité attendues. En l'occurrence, nous ne ferons usage dans le TP que de l'outil ATSE. La bonne pratique, si un jour vous devez effectuer par vous même cette vérification, c'est d'abord de modéliser votre protocole cryptographique dans un langage formel. Le TP utilise le langage hlpsl ("High-Level Protocol Specification Language"). Dans la spécification formelle obtenue, vous devez mentionner les propriétés de sécurité que vous souhaitez vérifier, par exemple, la confidentialité du secret transmis. L'outil vérifie ensuite si le protocole tel que modélisé satisfait les propriétés indiquées. S'il détecte une attaque possible par un attaquant dénommé "Intruder" dans l'outil, l'outil SPAN vous permettra de visualiser l'attaque. L'attaquant dans l'outil a les pleins pouvoirs sur le canal de communication. Il peut être posté en écoute sur le canal, détourner des messages, les modifier, les rejouer... Il s'agit d'un attaquant de type Dolev-Yao.

Attention ! Ce n'est pas parce que l'outil ne détecte aucune attaque que votre protocole satisfait forcément les propriétés indiquées. Cela signifie juste que l'outil n'a pas détecté d'attaques.

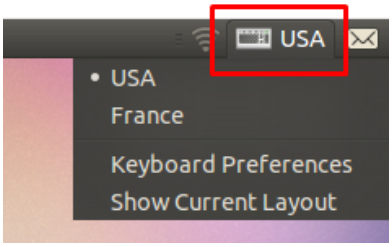
Pour une documentation plus complète sur l'utilisation de SPAN, vous êtes invités à consulter [1]. Une documentation plus approfondie sur les concepts est disponible ici [2].

4 Mise en place du TP

Ce TP utilise une machine virtuelle (VM) au format OVA.
Étapes si vous ne disposez pas d'outil de virtualisation :

1. Installer la version de VirtualBox la plus récente et compatible avec votre ordinateur `https://www.virtualbox.org/wiki/Downloads`
2. Lancer VirtualBox et cliquez sur Importer.
3. Sélectionner la VM du TP
4. Un écran vous permettant de modifier la configuration de la VM apparaît, cliquer sur Suivant puis démarrer.
5. Si l'installation est réussie (ne pas tenir compte du message d'erreur), vous devriez voir l'écran de connexion et être capable d'accéder au bureau qui comprend entre autres un exécutable SPAN et un dossier TP.

Pour changer la disposition du clavier : Tout en haut à droite de la VM vous pouvez choisir la disposition clavier, par défaut en QWERTY (USA) vers AZERTY (Fra). Voir l'image ci-après.



5 Notations

Les notations utilisées dans le TP pour décrire les protocoles de sécurité sont indiquées dans le tableau 1. La notation AVISPA fait référence aux notations utilisées dans les fichiers hspsl fournis dans la VM. La notation formelle fait référence aux notations utilisées dans l'énoncé de TP.

Table 1: Tableau récapitulatif des notions et notations

Notion	**Notation formelle**	**Notation AVISPA**
Identité du participant A, Alice	A ou a	A ou a
Identité du participant B, Bob	B ou b	B ou b
Identité de l'attaquant (ou intrus)	I ou i	I ou i
Secret	S	S, S' ou $S1$
Envoi d'un message M depuis A vers B	$A \rightarrow B : M$	$SND(M)$
Réception d'un message M provenant de A par B	$A \rightarrow B : M$	$RCV(M)$
Concaténation d'un mot W au message M	$M.W$	$M.W$
Chiffrement de S avec la clé K	$\{S\}_K$	$\{S\}\_K$
Chiffrement de S avec la clé publique de A	$\{S\}_{Ka}$	$\{S\}\_Ka$
Chiffrement de S avec la clé privée de A	$\{S\}_{Ka^{-1}}$	$\{S\}\_inv(Ka)$
Nonce généré par A	Na	Na ou Na'
Nonce généré par B	Nb	Nb ou Nb'

6 Quelle idée clairvoyante d'envoyer un secret en clair !

Alice et Bob souhaitent s'échanger des secrets sur Internet et leur choix se porte sur un canal de communication qui permet d'échanger des messages de manière hyper simple, rapide et gratuite ! Il est si simple, rapide et avancé qu'il s'appelle NSA (pour "Network Super Advanced"). On ne sait pas vraiment qui (ou quelle agence gouvernementale) se cache derrière ce canal, et c'est bien ça le problème.

Alice et Bob souhaitant se vanter d'avoir pu tester en avant première ce tout nouveau canal, se disent que c'est l'occasion rêvée pour s'échanger des petits secrets sans que personne ne le sache !

Le protocole mis en oeuvre dans ce canal NSA est le suivant :

$$A \rightarrow B : S.A$$

ce qui signifie que A envoie à B un secret S concaténé à son identité A.
Envoyer l'identité de l'émetteur A dans le message permet au destinataire de savoir qui lui a envoyé le message et de pouvoir répondre à la bonne personne.

Question 1) Quel est le risque de l'envoi d'un message en clair dans un canal de communication ? [1 bonne réponse]

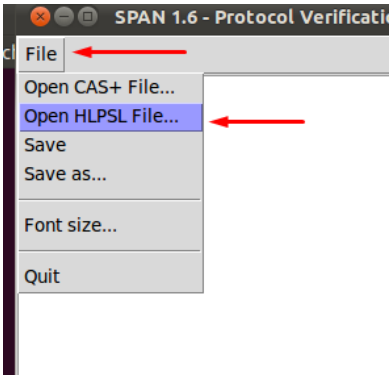
(a) Des personnes à qui le message n'est pas destiné pourront lire son contenu.

(b) Le message risque d'arriver trop rapidement car aucun traitement surperflu n'a été effectué.

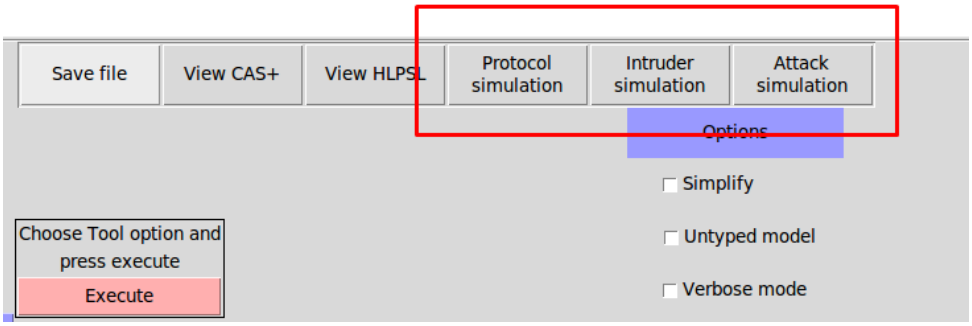
(c) Aucun, il est bien connu que tout système prétendu INFAILLIBLE n'a jamais subi de failles...

Dans la machine virtuelle (VM), accédez au bureau et double-cliquez sur le dossier "TP". Vous pouvez y voir les 10 fichiers hlpsl qui seront utilisés lors du TP et que vous analyserez avec le logiciel SPAN.

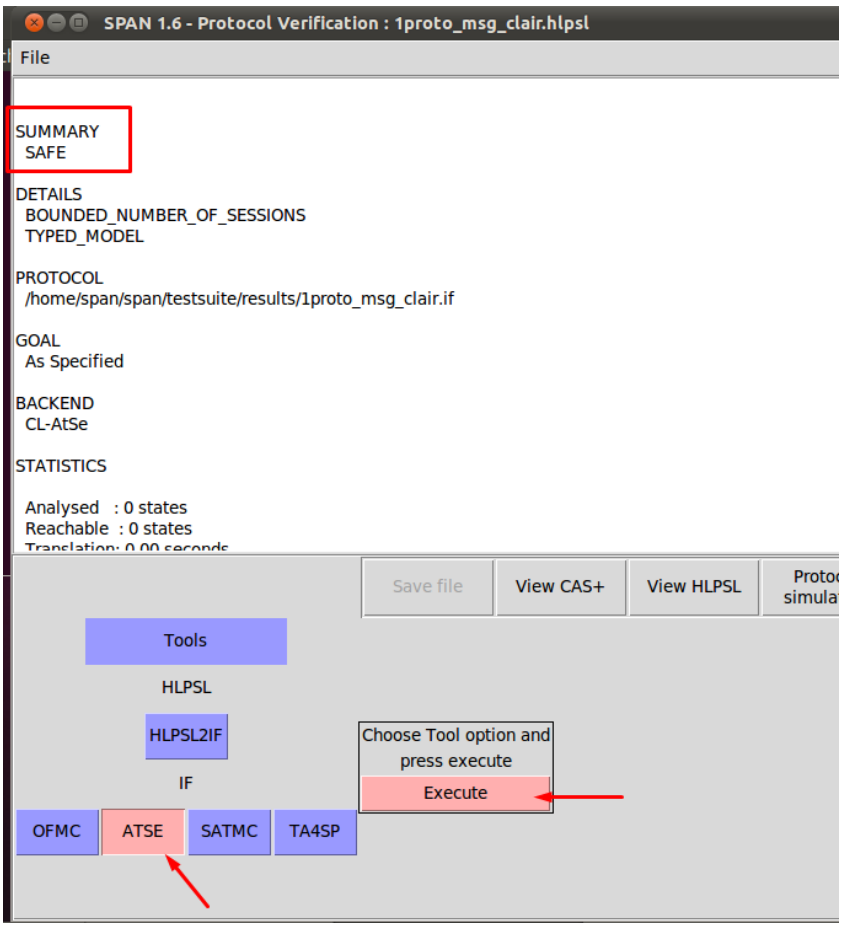
Lancez SPAN en double cliquant dessus depuis le Bureau. Dans File→Open HLPSL File...→Desktop→TP et double cliquez sur 1*proto_msg_clair.hlpsl* pour ouvrir le fichier depuis SPAN.



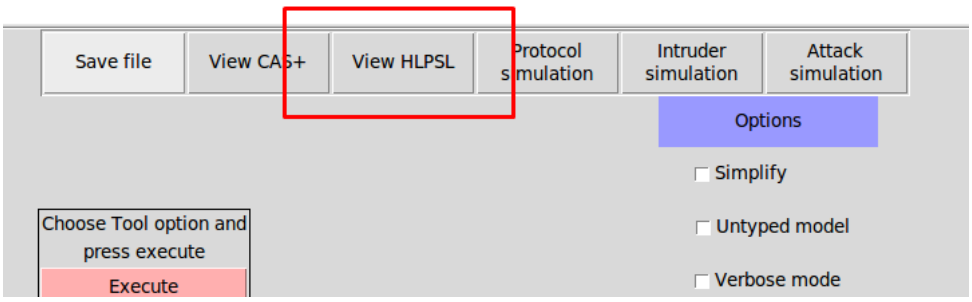
En cliquant sur Protocol Simulation, Intruder Simulation ou Attack Simulation, vous pouvez voir les trames du déroulé du protocole respectivement entre les participants légitimes, les participants légitimes et l'intrus, et le déroulé de l'attaque détectée par SPAN.



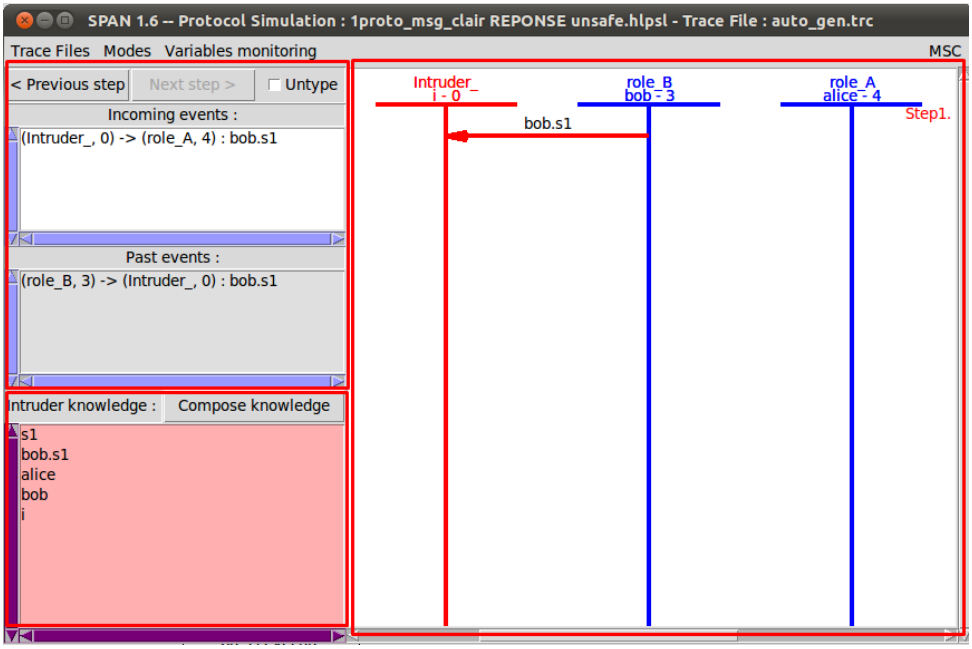
En analysant le protocole "msg clair" avec SPAN, cliquez en bas à gauche sur ATSE puis EXECUTE, vous devriez obtenir dans le cadre en haut le résultat "SAFE" dans SUMMARY qui correspond au résultat de la vérification... Cela paraît contre intuitif... La raison est simple et vous la retrouverez dans la suite : il nous faut introduire les propriétés que le protocole est sensé satisfaire et définir le secret à protéger dans le fichier hlpsl.



Retournez sur le Bureau, puis dans TP, double cliquez sur le fichier 1*proto_msg_clair.hlpsl* pour l'ouvrir avec un éditeur de texte. Dans le fichier, décommentez en supprimant le "%" devant les lignes 29 "/\secret(S,sec_1,A,B)" pour définir dans le protocole ce qui relève du secret et à la fin du fichier, lignes 59, 60 et 61 pour spécifier le secret à protéger et ainsi vérifier les propriétés de confidentialité. Le numéro de la ligne courante est indiquée en bas à droite de l'éditeur de texte. Sauvegardez et retournez dans SPAN.
Dans SPAN, cliquez sur View HLPSL pour actualiser les modifications.



Cliquez sur ATSE puis EXECUTE pour analyser le protocole. Le résultat attendu dans SUMMARY est UNSAFE. Cliquez sur Attack Simulation, pour voir apparaître la trame de l'attaque identifiée par SPAN dans une nouvelle fenêtre. N'hésitez pas à agrandir la fenêtre pour bien visualiser tous les échanges entre tous les acteurs.



Vous y verrez sur la gauche les actions faisables et déjà effectuées dans le protocole. Vous pouvez cliquer sur une ligne de la section "Incoming events" ou sur le bouton "< Previous step". A droite de la fenêtre, vous verrez la trame de l'attaque identifiée par SPAN. Cette section est complétée en bas à gauche par la section Intruder Knowledge qui indique tous les éléments à la connaissance de l'intrus à l'instant t (ici le secret $s1$, le message $bob.s1$ envoyé, les identités de A et B *alice* et *bob*, ainsi que sa propre identité i).

Question 2) Décrivez l'attaque identifiée par SPAN. [1 bonne réponse]

(a) L'intrus se fait passer pour Bob auprès de Alice et peut lire ainsi le secret $s1$ envoyé par Alice.

(b) L'intrus effectue une attaque par rejeu.

(c) Malgré l'attaque, l'intrus n'a pas accès au contenu du message, il ne peut donc pas lire le secret.

(d) L'intrus intercepte le secret $s1$ et peut directement lire son contenu.

Pour échanger des informations secrètes, que seules les personnes légitimes peuvent interpréter, il faut chiffrer les messages et ainsi garantir leur **confidentialité**.

7 Le chiffrement symétrique, c'est merveilleux (en théorie)

Après ce dur rappel à la réalité, Alice ne se sent plus du tout au pays des merveilles... Elle ne s'imaginait pas qu'il puisse y avoir des personnes malveillantes qui viennent agir sur son super canal de communication "Network Super Advanced".
Après avoir repris ses esprits, elle se souvient qu'il existe une famille d'algorithmes de chiffrement bien connue, réputée INFAILLIBLE (en théorie) : le chiffrement symétrique ! Elle met beaucoup d'espoir sur cette technique là.

Alice explique à Bob le principe simple du chiffrement symétrique qui en plus a l'avantage d'être peu coûteux en calculs : "Pour chiffrer un message secret, il suffit de générer aléatoirement une clé cryptographique qui va servir à la fois à chiffrer et à déchiffrer le message. Ainsi, il nous suffit de communiquer avec les personnes qui détiennent la clé symétrique et seules ces personnes pourront déchiffrer les messages !".

Résumé du protocole : Les participants détiennent des clés symétriques avant que le canal de communication ne soit utilisé. L'émetteur envoie son identité et le secret, chiffrés à l'aide de la clé symétrique au destinataire. Après réception, le destinataire déchiffre le tout avec la même clé symétrique pour lire le contenu. En voici le résultat en notation scientifique :

$$B \rightarrow A : \{B.S\}_K$$

Depuis SPAN, ouvrez le fichier $2proto\_symetrique.hlpsl$: dans File→Open HLPSL File...→Desktop→TP et double cliquez sur $2proto\_symetrique.hlpsl$.

Le protocole fait intervenir une clé symétrique qui servira à Bob à chiffrer le message et à Alice à déchiffrer le message.
Cliquez sur ATSE puis EXECUTE pour analyser le protocole. Le résultat attendu dans SUMMARY est SAFE.

Malgré les avantages théoriques indéniables du chiffrement symétrique, vous expliquez à Alice que très peu de protocoles de sécurité modernes s'appuient exclusivement sur la cryptographie symétrique, principalement à cause de la mise en pratique.

Question 3) Parmi les affirmations suivantes, lesquelles sont vraies pour le protocole défini jusque là avec du chiffrement symétrique ? [2 bonnes réponses]

(a) Si un participant souhaite être ajouté au réseau, les autres participants peuvent lui communiquer leurs clés symétriques directement sur le canal.

(b) Si un participant perd ses clés, il n'est plus possible de communiquer avec lui sans révéler publiquement les clés symétriques sur le canal.

(c) Il est impossible d'ajouter de nouveaux participants sans utiliser d'autres canaux pour partager les clés symétriques de manière confidentielle.

(d) Si un participant perd ses clés symétriques, il peut les retrouver en utilisant des moyens mnémotechniques.

(e) Si un participant perd ses clés symétriques, il lui suffit de déchiffrer les messages envoyés par force brute.

Une bonne pratique quant à l'usage des clés de chiffrement symétrique est d'éviter d'utiliser plusieurs fois la même clé avec différents participants pour limiter les risques que la clé soit compromise, et limiter les dégâts. En particulier, si une clé est volée à l'insu des deux participants, l'intrus pourra déchiffrer les messages qui ont déjà été échangés (messages passés), tout comme les messages à venir (messages futurs), ce qui conduit à la perte de confidentialité des communications. Il est alors intéressant de générer des clés uniques pour chaque interlocuteur, ce qui peut vite devenir coûteux en terme de ressources mémoire.

Question 4) Dans un système avec n participants, si une clé symétrique unique doit être sauvegardée pour chaque participant par tous les participants, combien de clés faut-il alors stocker ? [2 bonnes réponses]

(a) Chaque participant stocke uniquement sa clé symétrique.

(b) Chaque participant doit stocker $(n-1)$ clés.

(c) Parmi tous les participants, le total de clés sauvegardées s'élève à $n \times (n-1)$.

(d) Parmi tous les participants, le total de clés sauvegardées s'élève à n.

Depuis 1963 pour limiter les tensions en pleine Guerre Froide, les président des États-Unis et de l'URSS disposaient d'une ligne téléphonique directe secrète entre Moscou et Washington : le fameux téléphone rouge. À l'époque, le chiffrement utilisé était symétrique et les clés étaient à usage unique pendant les conversations. Chaque appel consommait des clés qu'il fallait renouveler, synchroniser et partager via des valises diplomatiques transportées entre les deux nations.
Le canal de communication principal, chiffré symétriquement, était donc la ligne téléphonique rouge mais pour fonctionner et garantir sa sécurité, il fallait utiliser un canal auxiliaire pour l'échange de clés. Ce canal auxiliaire prenait la forme du transfert physique de valises diplomatique top secrètes.
On remarque qu'aucune clé de chiffrement n'était échangée sur le canal de communication principal (la ligne téléphonique rouge). L'écoute sur la ligne ne permettait donc pas de récupérer les clés et donc de porter atteinte à la confidentialité des communications. Cependant, le risque était déplacé sur le canal auxiliaire, ce qui complexifie l'attaque puisqu'il faut non seulement maitriser le canal de communication principal (en réalisant une écoute), mais maitriser aussi le canal auxiliaire (en déjouant la vigilance des porteurs de valises).

Question 5) Que se passe t-il si une clé de chiffrement symétrique est volée ? [1 bonne réponse]

(a) Les messages chiffrés avec cette clé et envoyés sur le réseau sont compromis. La propriété de confidentialité n'est plus garantie tant que l'ensemble des clés compromises ne sont pas mises à jour avec de nouvelles clés symétriques chez tous les participants concernés.

(b) Les participants seront capables d'identifier systématiquement un intrus qui utilise la clé volée d'une personne légitime.

(c) Les messages chiffrés avec les autres clés symétriques sont également compromis et leur confidentialité ne sera plus assurée tant que l'ensemble des clés symétriques du réseau ne seront pas mises à jour.

Pour palier le problème de partage de clés, une première idée est d'envoyer en clair les clés sur le canal de communication, ce qui revient à supposer que le "premier message est sécurisé", ce qui est ÉVIDEMMENT un énorme risque !

Ne transmettez JAMAIS une clé en clair sur un réseau de communication ! Assurez-vous toujours de la confidentialité de son transfert !

Enfin, en cas de mise à jour de clés, de perte de clés, ou de l'ajout de nouveaux participants, il faut mettre à jour l'ensemble des clés du système, ce qui se révèle très coûteux en communication et en charges administratives.

Les protocoles de sécurité s'appuyant exclusivement sur de la cryptographie symétrique ne sont donc pas à privilégier, non pas pour des raisons de sécurité, mais pour des raisons de déploiement et de passage à l'échelle.

8 Le chiffrement asymétrique, le couteau suisse de la crypto

Bob est déçu des échecs successifs rencontrés dans la conception d'un protocole de sécurité robuste et il en veut à Alice. Bob décide tout de même de passer l'éponge, et se souvient d'un cours de cybersécurité sur le chiffrement asymétrique. Il s'agit d'une approche multifonction largement utilisée qui corrige de nombreux défauts pratiques du chiffrement symétrique !

Bob explique à Alice : un couple de clés est généré localement. Une de ces clés est désignée comme **clé publique** et sera visible par l'ensemble des participants sur le réseau. La seconde clé quant à elle sera la **clé privée**, elle ne sera connue que de son propriétaire et ne sera jamais révélée à personne. La clé privée est la seule clé qui peut déchiffrer un message chiffré avec la clé publique et inversement.

Principe de la cryptographie à clés publiques : Bob chiffre son identité et le secret avec la clé PUBLIQUE d'Alice, Alice déchiffre ensuite le message (envoyé par Bob) avec sa propre clé PRIVÉE. Comme Alice est la seule à connaître sa clé privée, la confidentialité du secret inclu dans le message sera assurée, et il y a moins de risque de divulgation d'une clé privée, comparativement à une clé symétrique partagée !

Question 6) Lesquelles de ces affirmations concernant le chiffrement symétrique comparé au chiffrement asymétrique sont vraies ?

(a) Il y a moins de clés à sauvegarder car chaque participant a juste besoin de sauvegarder ses clés asymétriques et à demander la clé publique des autres participants au besoin.

(b) Le nombre total de clés à sauvegarder est doublé car il y la clé publique ET la clé privée à sauvegarder.

(c) Si une clé privée est corrompue, la confidentialité des messages n'est pas menacée car, sans la clé publique, on ne peut pas déchiffrer les messages.

(d) Tout comme le chiffrement symétrique, le premier message doit être en clair pour initialiser les clés du chiffrement si on souhaite utiliser uniquement le canal de communication principal, et donc la confidentialité n'est que partielle.

Résumé du protocole considéré : Bob envoie un secret et son identité chiffrés avec une clé asymétrique à Alice.
Dans SPAN, ouvrez le fichier $3proto\_asymetrique.hlpsl$. Dans File→Open HLPSL File...→Desktop→TP et double cliquez sur $3proto\_asymetrique.hlpsl$.
Retournez sur le Bureau, puis dans TP, double cliquez sur le fichier pour l'ouvrir avec un éditeur de texte. Remplacez le mot CLE dans $RCV(\{B.S'\}\_CLE)$ ligne 9 et $SND(\{B.S\}\_CLE)$ ligne 26 par l'une des clés suivantes :

- Clé publique de A : Ka
- Clé publique de B : Kb
- Clé privée de A : $inv(Ka)$
- Clé privée de B : $inv(Kb)$

Attention : pour fonctionner, il faut que la clé soit identique aux deux endroits.

Sauvegardez et retournez dans SPAN.
Dans SPAN cliquez sur View HLPSL pour actualiser les modifications. Cliquez sur ATSE puis EXECUTE pour analyser le protocole. Analysez les trames des différentes simulations Protocol simulation et Attack simulation proposées par SPAN.

Question 7) En vous aidant des résultats d'analyse d'attaque d'AVISPA et de votre analyse de la situation, lesquelles de ces affirmations sont vraies ?

(a) Les messages chiffrés avec les clés publiques sont analysées comme "SAFE" car l'intrus ne peut pas deviner les clés privées de A ou B.

(b) Les messages chiffrés avec les clés privées sont UNSAFE car l'intrus peut les déchiffrer avec les clés publiques de A ou B.

(c) Même si le message chiffré avec la clé privée d'Alice apparaît comme SAFE, le scénario n'est pas possible car Bob n'a pas connaissance de cette clé.

(d) Pour garantir que le protocole soit SAFE, il faut toujours chiffrer les messages avec la clé privée du destinataire.

Maintenant, vous maitrisez les subtilités du chiffrement asymétrique et l'utilisation de ses clés. Mais ce n'est pas le seul intérêt du chiffrement asymétrique, il permet également de garantir une seconde propriété de sécurité fondamentale : l'**authentification**.

9 L'authentification de l'origine des messages avec le couteau suisse asymétrique

La confidentialité des messages est désormais garantie et ce dès le premier message entre deux participants. Seul le destinataire légitime est donc en capacité de lire le contenu du message chiffré. Cependant, n'importe qui peut envoyer un message chiffré et inclure l'identité qu'il souhaite pour usurper un participant... Bob explique qu'il ne veut pas envoyer des messages à une Alice quelconque mais seulement à SA Alice ! Alice, le feu aux joues, lui répond "B...Baka ! ><". Youpi ! Les voilà réconciliés !

Il nous faut donc un moyen fiable de prouver l'identité des participants auprès des autres - il s'agit de l'**authentification** qui est notre seconde propriété de sécurité essentielle à vérifier, et en particulier l'**authentification de l'origine des messages** qui vise à s'assurer qu'un message provient de l'interlocuteur déclaré.

Lancez le protocole $4proto\_auth\_UNSAFE.hlpsl$ dans SPAN. Dans File→Open HLPSL File...→Desktop→TP et double cliquez sur $4proto\_auth\_UNSAFE.hlpsl$ pour ouvrir le fichier depuis SPAN.
Il s'agit du protocole précédent qui chiffre le message pour Alice, auquel on a rajouté dans la section "goal" (ligne 64) le besoin d'authentification ainsi que des instructions dans les rôles (lignes 14 et 32) pour qu'AVISPA renvoie "SAFE" si et seulement si les propriétés de confidentialité **et** d'authentification de l'origine du message sont toutes les deux satisfaites.

Dans SPAN cliquez sur ATSE puis EXECUTE pour analyser le protocole. Le résultat attendu dans SUMMARY est UNSAFE. Cliquez sur Attack Simulation pour afficher le scénario de l'attaque identifiée par SPAN. L'attaque identifiée par AVISPA s'appelle une attaque par usurpation.

Question 8) A l'aide de la trame d'attaque retournée par AVISPA, que pouvez-vous dire de cette attaque par usurpation ? [2 bonnes réponses]

(a) Le destinataire peut vérifier l'identité de l'émetteur en déchiffrant le message.

(b) Le résultat est UNSAFE car l'attaquant peut lire le secret.

(c) L'attaque est possible car n'importe qui peut générer un message chiffré avec une clé publique en incluant n'importe quelle identité.

(d) L'attaquant a déchiffré le secret pour générer son attaque et a réussi à ouvrir la connexion avec Alice.

(e) L'attaque n'aurait pas été possible avec un chiffrement symétrique.

Pour empêcher cette attaque par usurpation, il faut s'assurer que le message provienne bien de l'expéditeur déclaré : Bob. L'expéditeur peut par exemple chiffrer son message avec sa clé privée, de sorte qu'Alice soit sûre que le message provient bien de Bob (le seul à connaître sa clé privée)[1].

Question 9) Quelles modifications peut-on apporter aux lignes 9 et 27 du protocole pour vérifier que Bob est bien l'émetteur du message, et ainsi corriger la vulnérabilité ? [1 bonne réponse]

(a) $RCV(\{B.S'\}\_Ka) \rightarrow RCV(\{B.\{S'\}\_Kb\}\_Ka)$ (ligne 9)
$SND(\{B.S\}\_Ka) \rightarrow SND(\{B.\{S\}\_Kb\}\_Ka)$ (ligne 27)

(b) $RCV(\{B.S'\}\_Ka) \rightarrow RCV(\{B.\{S'\}\_inv(Kb)\}\_Ka)$ (ligne 9)
$SND(\{B.S\}\_Ka) \rightarrow SND(\{B.\{S\}\_inv(Kb)\}\_Ka)$ (ligne 27)

(c) $RCV(\{B.S'\}\_Ka) \rightarrow RCV(\{B.\{S'\}\_Ka\}\_Ka)$ (ligne 9)
$SND(\{B.S\}\_Ka) \rightarrow SND(\{B.\{S\}\_Ka\}\_Ka)$ (ligne 27)

(d) $RCV(\{B.S'\}\_Ka) \rightarrow RCV(\{B.\{S'\}\_inv(Ka)\}\_Ka)$ (ligne 9)
$SND(\{B.S\}\_Ka) \rightarrow SND(\{B.\{S\}\_inv(Ka)\}\_Ka)$ (ligne 27)

Pour vous aider : Retournez sur le Bureau, puis dans TP et double cliquez sur le fichier $4proto\_auth\_UNSAFE.hlpsl$ pour l'ouvrir avec un éditeur de texte. Modifiez le fichier avec la réponse souhaitée puis sauvegardez. Dans SPAN, cliquez sur View HLPSL pour actualiser les modifications. Cliquez sur ATSE puis EXECUTE pour analyser le protocole et vérifiez que vous obtenez bien le résultat SAFE.

[1] Attention toutefois ! Pour qu'Alice soit sûre que c'est bien Bob qui a émis le message, elle doit s'assurer que le déchiffrement du message avec la clé publique de Bob s'est correctement passé. Pour cela, elle doit s'assurer que le contenu du message est cohérent, par exemple, il contient l'identité de bob.

Le protocole 5*proto_auth_rejeu.hlpsl* ajoute une session au protocole, c'est-à-dire que deux sessions parallèles permettent à Alice et Bob de communiquer. Cet ajout de session correspond à un contexte plus réaliste sur un réseau car Alice et Bob sont amenés à communiquer plusieurs fois entre eux.
Lancez le protocole 5*proto_auth_rejeu.hlpsl* dans SPAN. Dans File→Open HLPSL File...→Desktop→TP et double cliquez sur 5*proto_auth_rejeu.hlpsl*.
Dans SPAN cliquez sur ATSE puis EXECUTE pour analyser le protocole. SPAN renvoie UNSAFE et décrit **une attaque par rejeu**.
Cliquez sur Attack Simulation pour afficher le scénario de l'attaque par rejeu identifiée par SPAN.
En analysant l'attaque, on constate que l'attaquant a intercepté un message légitime de Bob et a renvoyé exactement le même message plus tard afin d'induire en erreur Alice et de lui faire croire que les messages proviennent de Bob alors qu'ils proviennent de lui. C'est donc une usurpation d'identité réalisée grâce à un rejeu de messages. Si ce protocole était déployé dans un vrai réseau, avec Alice dans le rôle d'un service ou d'un serveur, et une authentification basée sur l'envoi d'un mot de passe comme secret, alors un attaquant pourrait ouvrir une session en se faisant passer pour un employé Bob auprès du serveur, et ce sans que Bob ne s'en rende compte.

Question 10) En vous aidant des informations disponibles générées dans Attack Simulation par SPAN, l'attaquant peut-il lire le secret ? [1 bonne réponse]

(a) Oui, car on a spécifié le besoin de confidentialité du secret.

(b) Non, car il n'a accès qu'au message chiffré.

(c) Oui, sinon l'attaquant n'aurait pas pu s'authentifier.

(d) Non, car l'identité réelle du participant qui a généré le message n'est pas modifiée et donc le destinataire ne renvoie pas le secret à l'attaquant.

La vulnérabilité de ce protocole est simple : un message identique aura un chiffré identique ou tout du moins, un message chiffré restera valide du point de vue cryptographique pour le destinataire. Tout le monde peut donc intercepter le message et l'utiliser à son compte autant de fois qu'il le veut.

Un autre exemple plus concret est le suivant : imaginez un service nécessitant de s'authentifier auprès d'un serveur (Alice). Si cette authentification consiste à envoyer simplement vos login / mot de passe, ou bien vos login / mot de passe chiffrés avec la clé publique du serveur, ces informations ne diffèrent pas d'une session d'authentification à une autre et peuvent alors être identifiés puis rejouées de façon illégitime auprès du serveur pour usurper votre identité.

10 Échange de nonces - Comment faire en sorte que chaque session soit unique pour déjouer les attaques par rejeu ?

Après avoir analysé le protocole et l'attaque par rejeu, Alice pense qu'il faudrait un moyen de rendre les messages uniques et donc non réutilisables.
C'est là que les **NONCES** entrent en jeu.

Les **nonces** sont par définition des nombres aléatoires **UNIQUES**. Quand un protocole fait usage de nonces, il est sous-entendu que chaque génération de nombre aléatoire donne lieu à un tirage différent. C'est une hypothèse sur laquelle repose le protocole de sécurité et qui doit ensuite être respectée dans les implémentations du protocole.
L'utilisation des nonces dans un protocole permet donc de **rendre chaque message chiffré - pour peu que le message inclut le nonce - unique** dans son contenu et d'apparence totalement aléatoire, et ce même si le message initial est identique.
Les nonces sont couramment utilisés dans les protocoles de sécurité pour **garantir la fraicheur des messages**, c'est-à-dire garantir aux interlocuteurs que les messages font partie de leur interaction directe, qu'ils ne sont pas rejoués, injectés dans une communication...
Les nonces sont souvent utilisés **pour entrelacer les messages entre eux**. Par exemple si le message 1 $M1$ comprend le nonce $N1$, alors le message 2 $M2$ en retour va contenir ce même nonce $N1$. Cela permettra à l'émetteur de $M1$ d'avoir la garantie que $M2$ est fourni en réponse à $M1$ et donc la fraicheur de $M2$. On peut imaginer plusieurs messages $M1$, $M2$, $M3$ successifs qui utilisent deux nonces, $M1$ et $M2$ transportent le nonce $N1$ et $M2$ et $M3$ le nonce $N2$. Cette notion d'entrelacement est très importante pour concevoir des protocoles résistants aux rejeux.

Il existe deux version du protocole *proto_nonces_placement.hlpsl* : une version analysée par SPAN comme SAFE (6*proto_nonces_placement_SAFE.hlpsl*) et l'autre UNSAFE (7*proto_nonces_placement_UNSAFE.hlpsl*). Lancez successivement les deux protocoles *proto_nonces_placement.hlpsl* dans SPAN. Dans File→Open HLPSL File...→Desktop→TP et double cliquez sur le *proto_nonces_placement.hlpsl* souhaité.
Dans SPAN cliquez sur ATSE puis EXECUTE pour analyser le protocole. Le résultat attendu dans SUMMARY correspond au nom du fichier. Cliquez sur Attack Simulation pour afficher la trame de l'attaque identifiée par SPAN ou sur Protocol Simulation pour analyser le déroulé du protocole.

Question 11) Quelles affirmations sur les protocoles *proto_nonces_placement* SAFE et UNSAFE sont vraies ? [4 bonnes réponses]

(a) L'un des protocole chiffre les identités et ajoute le nonce au message, le second chiffre le nonce.

(b) L'attaque sur le protocole UNSAFE est possible car les nonces ne sont pas chiffrés.

(c) L'attaque sur le protocole UNSAFE est possible car les identités ne sont pas protégées.

(d) Le protocole SAFE est sûr car Bob et Alice sont en capacité de s'authentifier mutuellement, c'est-à-dire de vérifier l'identité de l'autre de façon cryptographique.

(e) Pour corriger le protocole UNSAFE, il faut retirer le chiffrement des identités d'Alice et Bob.

(f) Le protocole SAFE permet à Alice d'authentifier Bob du fait qu'elle a la preuve que Bob connait sa clé privée. Bob a du déchifrer le premier message pour avoir connaissance du nonce, résultat qu'il a communiqué à Alice dans le 2ème message.

Après avoir vu dans le protocole SAFE, l'**authentification unidirectionnelle** où Bob s'authentifie vis à vis d'Alice avec l'anti-rejeu assuré par un nonce, passons à l'authentification mutuelle où Alice et Bob s'authentifient l'un envers l'autre, toujours avec de l'anti-rejeu et où un secret est transmis de façon confidentielle.

Alice propose d'utiliser le fameux **protocole Needham-Schroeder**. Le protocole Needham-Schroeder à clés publiques (NSPK) a été conçu en 1978. Il répond à ce besoin d'authentification mutuelle et a été largement utilisé. Il s'agit de l'un des premiers protocoles d'authentification mutuelle entre deux parties sur un canal non sécurisé.

On distingue les propriétés suivantes. L'**authentification unidirectionnelle ou mutuelle** nécessite plusieurs échanges de messages (deux messages dans le fichier 6*proto_nonces_placement_SAFE.hlpsl*) contrairement à l'**authentification de l'origine des messages** consiste à chiffrer un ou plusieurs éléments du message avec la clé privée de l'émetteur. L'**authentification unidirectionnelle**, comme c'est le cas dans 6*proto_nonces_placement_SAFE.hlpsl*, permet à un seul interlocuteur, ici Alice, d'authentifier Bob. L'**authentification mutuelle** permet aux deux interlocuteurs de s'authentifier l'un l'autre. C'est l'objectif du protocole NSPK.

Fonctionnement du protocole Needham-Schroeder à clés publiques (NSPK) : Alice envoie un nonce chiffré avec la clé publique de Bob. Bob déchiffre le nonce envoyé par Alice (avec sa clé privée), y ajoute son propre nonce et renvoie à Alice le tout chiffré avec la clé publique d'Alice. Alice déchiffre ensuite le message envoyé par Bob (avec sa clé privée), récupère le nonce de Bob et retourne à Bob ce nonce chiffré avec la clé publique de Bob. En voici une description formelle :

$$A \rightarrow B : \{Na.A\}_{Kb}$$
$$B \rightarrow A : \{Na.Nb\}_{Ka}$$
$$A \rightarrow B : \{Nb\}_{Kb}$$

Après ces échanges de nonces, les deux parties se sont authentifiées l'une envers l'autre et ont la garantie que leur session est unique et ne pourra pas être rejouée.

Lancez le protocole 8*proto_NSPK.hlpsl* dans SPAN. Dans File→Open HLPSL File...→Desktop→TP et double cliquez sur 8*proto_NSPK.hlpsl*.
Cliquez sur ATSE puis EXECUTE pour analyser le protocole. SPAN analyse le protocole comme étant SAFE. Retournez sur le Bureau, puis dans TP, double cliquez sur le fichier pour l'ouvrir avec un éditeur de texte. Dans le fichier, aux lignes 87 à 90, vous pouvez rajouter, une par une, une session parallèle à celle entre Alice et Bob en décommentant la ligne souhaitée en supprimant le "%" en début de ligne. N'activez qu'une nouvelle session à la fois.
A chaque manipulation, sauvegardez. Dans SPAN cliquez sur View HLPSL pour actualiser les modifications. Cliquez sur ATSE puis EXECUTE pour analyser le protocole.

Question 12) Dans quels cas le résultat retourné par SPAN du protocole NSPK est SAFE ?

(a) Une session entre Alice et Bob.

(b) Deux sessions en parallèle entre Alice et Bob.

(c) Deux sessions en parallèle dont une entre Alice et Bob et l'autre entre Bob et Alice.

(d) Deux sessions en parallèle dont une entre Alice et Bob et l'autre entre l'intrus et Bob.

(e) Deux sessions en parallèle dont une entre Alice et Bob et l'autre entre Alice et l'intrus.

11 17 ans de sentiment de sécurité - l'attaque Man in The Middle découverte par G. Lowe sur le protocole NSPK et sa solution

C'est en 1995, 17 ans après la création du protocole Needham-Schroeder NSPK, qu'une attaque a été identifiée par G. Lowe ! NSPK était alors largement utilisé et la découverte de la faille fît grand bruit. Il s'agit d'une faille logique du protocole qui a été identifiée grâce à des méthodes formelles et dont personne n'avait soupçonné l'existence jusqu'alors. Cela démontre tout l'intérêt d'utiliser des outils de vérification formelle lors de la conception de protocoles de sécurité.

L'attaque utilisée pour contourner le protocole NSPK proposé en 1978 est un type d'attaque très connu en cybersécurité nommé "Man in the Middle".

> Une **attaque Man in the Middle (MiTM)** (ou « homme du milieu ») a pour objectif d'intercepter les communications entre deux parties, sans que celles-ci puissent se douter que le canal de communication a été détourné.

Lancez le protocole 9*proto_NSPK_MITM.hlpsl* dans SPAN. Dans File→Open HLPSL File...→Desktop→TP et double cliquez sur 9*proto_NSPK_MITM.hlpsl*.
Dans SPAN cliquez sur ATSE puis EXECUTE pour analyser le protocole. Le résultat attendu dans SUMMARY est UNSAFE. Cliquez sur Attack Simulation pour afficher la trame de l'attaque identifiée par SPAN. L'attaque générée par SPAN est une attaque Man in the Middle (MitM).

Question 13) Quelles sont les affirmations vraies concernant l'attaque Man in the Middle sur le protocole NSPK telle que présentée par SPAN. [4 bonnes réponses]

(a) Alice envoie son nonce et son identité chiffrés avec la clé publique de l'Intrus à l'Intrus. L'Intrus déchiffre le contenu et l'envoi chiffré avec la clé publique de Bob à Bob. Bob ajoute son nonce au nonce de Alice et l'envoie chiffré avec la clé publique de Alice à l'Intrus. L'Intrus transmet le message de Bob contenant les nonces à Alice. Alice déchiffre le message, accepte le nonce et renvoie donc le nonce de Bob à l'Intrus chiffré avec la clé publique de l'intrus.

(b) Alice envoie son nonce et son identité chiffrés avec la clé publique de Bob à l'Intrus. L'Intrus déchiffre le contenu et l'envoie chiffré avec la clé publique de Bob à Bob. Bob ajoute son nonce au nonce de Alice et l'envoie chiffré avec la clé publique de l'Intrus à l'Intrus. L'Intrus transmet le message de Bob contenant les nonces à Alice. Alice déchiffre le message, accepte le nonce et renvoie donc le nonce de Bob à l'Intrus chiffré avec la clé publique de l'intrus.

(c) L'Intrus usurpe l'identité d'Alice auprès de Bob et rejoue certains messages chiffrés, et ce pour compromettre l'authentification mutuelle entre Alice et Bob.

(d) Alice et l'Intrus sont des partenaires de crime ! Sans l'aide d'Alice, l'Intrus ne pourrait pas obtenir le nonce de Bob. C'est un peu des Bonnie and Clyde du réseau sauf que Bonnie ne sait pas dans quoi on l'a embarquée...

(e) L'Intrus est capable de déchiffrer les messages chiffrés avec les clés publiques d'Alice et Bob entre Alice et Bob.

(f) Les messages doivent être chiffrés avec la clé publique de l'intermédiaire et non du destinataire, ce qui ne permet pas de garantir les propriétés de confidentialité si le message doit transiter par plusieurs intermédiaires.

(g) Pour que l'attaque MiTM soit possible, il est nécessaire qu'Alice initie de son fait une communication avec l'Intrus.

Le protocole 10$proto_NSPK_SAFE_lowe.hlpsl$ est la spécification en hlpsl de la correction du protocole NSPK proposée par Lowe en 1995.
Lancez le protocole 10$proto_NSPK_SAFE_lowe.hlpsl$ dans SPAN. Dans File→Open HLPSL File...→Desktop→TP et double cliquez sur 10$proto_NSPK_SAFE_lowe.hlpsl$.
Dans SPAN cliquez sur ATSE puis EXECUTE pour analyser le protocole. Le résultat attendu dans SUMMARY est SAFE. Cliquez sur Protocol Simulation puis sur toutes les actions qui apparaissent dans la section "Incoming events" pour afficher et analysez la trame de la solution de Lowe.

Question 14) Comment la solution de Lowe permet de corriger la vulnérabilité du protocole NSPK ? [2 bonnes réponses]

(a) En ajoutant l'identité de Bob dans l'envoi de nonces, ce qui empêche l'intrus d'utiliser le message de Bob vers Alice pour s'authentifier.

(b) En supprimant la session entre Alice et l'Intrus qui compromettait le protocole.

(c) En permettant à Alice de vérifier qui a généré le nonce et ainsi de rejeter la connexion.

(d) En corrigeant les erreurs de chiffrement entre A et B, ce qui permet de garantir la confidentialité des échanges.

(e) En ajoutant des nonces dans les échanges, ce qui empêche l'Intrus de tous les déchiffrer.

Alice et Bob comprennent enfin que la popularité d'un protocole ne garantit pas qu'il soit sûr du point de vue de la sécurité de sa conception, et encore moins de la sécurité de son implémentation. Ils comprennent aussi qu'il est important que les spécifications d'un protocole de sécurité soient rendues publiques. Cela permet de s'appuyer sur les compétences et la vigilance de la communauté cyber pour détecter d'éventuelles vulnérabilités d'ordre logique ou logiciel. A savoir, un protocole de sécurité propriétaire confidentiel ne sera jamais aussi sûr qu'un protocole rendu publique. Tous deux comprennent aussi l'importance de recourir à des outils de vérification formelle de protocoles tels que SPAN & AVISPA pour identifier de potentielles vulnérabilités lors de la conception.

12 Protocole "final" : à vous de jouer !

Après cette longue aventure, Alice et Bob ont appris que la sécurité d'un protocole ne s'improvise pas.
Ils souhaitent compiler tout ce qu'ils ont appris avec vous dans un seul et même protocole formel avant de l'implémenter.

Question 15) Quel protocole décrit ci-dessous en notation formelle nous permet de garantir l'authentification mutuelle et la confidentialité et l'authenticité de deux secrets Sa et Sb émis par Alice et Bob ? [2 bonnes réponses]

(a) $A \to B : \{A.B.\{Na.Sa\}_{Ka^{-1}}\}_{Kb}$
$B \to A : \{A.B.\{Na.Nb.Sb\}_{Kb^{-1}}\}_{Ka}$
$A \to B : \{A.B.\{Nb\}_{Ka^{-1}}\}_{Kb}$

(b) $A \to B : \{A.Na.Sa\}_{Kb}$
$B \to A : \{Na.Nb.Sb\}_{Ka}$
$A \to B : \{Nb\}_{Kb}$

(c) $A \to B : \{A.B.\{Na.Sa\}_{Kb^{-1}}\}_{Kb}$
$B \to A : \{A.B.\{Na.Nb.Sb\}_{Ka^{-1}}\}_{Ka}$
$A \to B : \{A.B.\{Nb\}_{Kb^{-1}}\}_{Kb}$

(d) $A \to B : \{\{Na.Sa\}_{Ka^{-1}}\}_{Kb}$
$B \to A : \{\{Na.Nb.Sb\}_{Kb^{-1}}\}_{Ka}$
$A \to B : \{\{Nb\}_{Ka^{-1}}\}_{Kb}$

(e) $A \to B : \{A\{Na.Sa\}_{Ka^{-1}}\}_{Kb}$
$B \to A : \{B\{Na.Nb.Sb\}_{Kb^{-1}}\}_{Ka}$
$A \to B : \{\{Nb\}_{Ka^{-1}}\}_{Kb}$

(f) $A \to B : \{A.B.\{Na.Sa\}_{Ka}\}_{Kb}$
$B \to A : \{A.B.\{Na.Nb.Sb\}_{Kb}\}_{Ka}$
$A \to B : \{A.B.\{Nb\}_{Ka}\}_{Kb}$

13 Conclusion

Pour concevoir un protocole cryptographique, il est nécessaire de lister dans un premier temps les propriétés de sécurité que vous attendez de ce protocole, puis de vérifier que le protocole conçu vérifie bien ces propriétés. Avec l'aide d'Alice et Bob (sans oublier l'attaquant), vous avez appris à utiliser la cryptographie symétrique et asymétrique, ainsi que les nonces, pour déjouer les rejeux de messages. Vous avez également été en mesure d'identifier les limites et les failles de ces éléments pour garantir la confidentialité des secrets, l'authentification de l'origine des messages et l'authentification unidirectionnelle ou mutuelle. Nous avons vu à quel point il est difficile de définir un protocole sûr et à quel point il est important d'utiliser des outils de vérification formelle pour identifier les failles des protocoles qui pourraient être exploitées.

Mais rappelez-vous que vous n'avez fait qu'effleurer la surface de la sécurité des protocoles ! Il faut être vigilant et rigoureux depuis la conception jusqu'à l'implémentation et au déploiement du protocole sur un réseau pour ne pas introduire de vulnérabilités, par exemple un buffer overflow, une mauvaise configuration avec des algorithmes cryptographiques faibles, une clé publique qui ne correspond pas à la bonne entité... En effet, il suffit d'une faille à n'importe laquelle de ces étapes pour que la sécurité des communications soit compromise.

References

[1] A Short SPAN+AVISPA Tutorial, Thomas Genet, INRIA, `https://inria.hal.science/hal-01213074`

[2] SPAN: a Security Protocol ANimator for AVISPA USER Manual, Thomas Genet, INRIA, `https://people.irisa.fr/Thomas.Genet/Crypt/manual.pdf#:~:text=Avispa%20%5BABB%2B05%2C%20avi%5D%20is%20now%20a%20commonly%20used,cation%20techniques%20on%20the%20same%20protocol%20speci%20cation`

[3] Cryptographie et sécurité, Franck Pommereau, cours donné aux Master 1 MIAGE et informatique CNS SA/SR, Université Paris-Saclay/UEVE, 2023.