Utilisation de Trusted Platform Modules pour la sécurité et l'anonymat des réseaux de véhicules

Gilles Guette (gilles.guette@irisa.fr)*
Olivier Heen (olivier.heen@irisa.fr)†

Résumé : Les réseaux de véhicules (VANET) font actuellement l'objet d'une attention accrue de la part des constructeurs et des chercheurs. Néanmoins, leur déploiement à grande échelle nécessite de résoudre de nombreux problèmes de sécurité et de respect de la vie privée. Dans cet article nous discutons de la pertinence d'une architecture de sécurité basée sur des TPM (Trusted Platform Module) embarqués dans les véhicules. Nous décrivons la gestion et le renouvellement du matériel cryptographique nécessaire à la sécurisation et à l'anonymisation des communications entre véhicules. Comparativement aux résultats du domaine, notre architecture ne nécessite aucun déploiement de stations de base le long des routes.
Une attention particulière est portée sur l'anonymat des communications afin d'éviter le traçage non autorisé d'un véhicule. Seule une autorité habilitée peut lever partiellement l'anonymat, comme c'est aujourd'hui le cas avec le système de plaques minéralogiques. Nous discutons de la résistance de notre système d'anonymat à la compromission d'un des acteurs. Enfin nous indiquons comment l'utilisation d'un élément de stockage portable, une mémoire USB par exemple, permet d'améliorer la qualité de l'anonymat en liant de manière opportuniste le TPM à un serveur de sécurité.

Mots Clés : Sécurité des communications, TPM, Réseau de véhicules, VANET, Anonymat.

1 Introduction

Ces dernières années, les réseaux de véhicules font l'objet d'une grande attention de la part des constructeurs et des chercheurs [CG07b, CG07a]. Une des raisons principales est le nombre grandissant d'applications conçues pour la sécurité des passagers telles que le freinage d'urgence, la détection de bouchon, la conduite coopérative, mais aussi pour leur confort, telles que les jeux, les messageries instantanées ou le partage de données entre véhicules (*i.e.* CarTorrent [LLC+07]). Les VANETs sont des réseaux ad hoc fortement dynamiques possédant un accès très restreint à une infrastructure réseau. De plus, si des stations de bases sont déployées de manière clairsemée le long de la route, cet accès est aussi de très courte durée et n'est pas continu, notamment à cause de la vitesse de déplacement des véhicules. Les applications embarquées ont besoin d'échanger des informations c'est pourquoi le problème de la sécurité des communications doit être résolu. L'absence d'une infrastructure réseau accessible de manière permanente signifie qu'une architecture décentralisée est nécessaire. De plus, étant donnée la nature critique des applications relatives à la sécurité des passagers, cette architecture de sécurité doit empêcher un attaquant de lancer avec succès une attaque visant à provoquer des collisions. Dans ce papier, nous

* Université de Rennes 1/IRISA, Campus Universitaire de Beaulieu, 35042 Rennes
† INRIA Rennes Bretagne Atlantique, Campus Universitaire de Beaulieu, 35042 Rennes

supposons une architecture basée sur des TPM [TCG05, GB08] embarqués dans les véhicules pour sécuriser les échanges entre les nœuds mobiles qui constituent le réseau (*i.e.* les véhicules). Les spécifications des TPM sont en permanente évolution et sujet à recherche au sein du *Trusted Computing Group Association* (TCGA) [TCGW].

Un TPM est une puce matérielle avec des capacités cryptographiques. Certaines de ces fonctions nécessitent que le TPM puisse parfois se connecter à un serveur externe. Notre idée principale est d'utiliser une mémoire USB[1] portable pour relier le TPM embarqué dans le véhicule et l'infrastructure filaire, cf. figure 1. La mémoire USB est utilisée pour conserver les données issues des capteurs du véhicule comme la consommation de carburant, et des données venant du TPM telles que les requêtes de certification de clés. Le conducteur apporte la mémoire USB d'un bâtiment possédant une connexion à Internet (maison, bureau) jusqu'au véhicule.

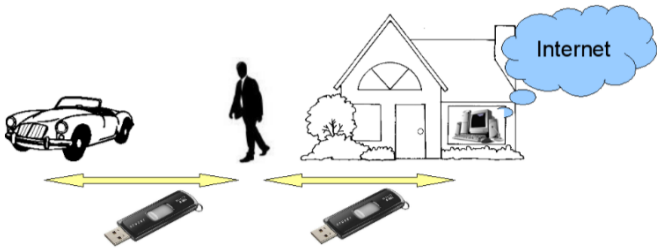


Fig. 1: Un élément de stockage portable, une mémoire USB, couvre les derniers mètres.

Dans la section 2, nous présentons les motivations de ce papier et nous rappelons l'environnement technique. Dans la section 3, nous présentons notre solution et nous décrivons les protocoles de communication entre les différents éléments constituant l'architecture proposée. Dans la section 4, nous discutons plusieurs aspects de notre solution comme sa sécurité, les applications incitatives pour l'utiliser et quelques considérations quantitatives.

2 Motivations

La solution proposée dans ce papier permet des communications **sécurisées** et **anonymes** et **certifiées** entre les véhicules qui constituent le réseau.

Communications sécurisées : il s'agit en particulier de compliquer l'injection de paquets malveillants et d'en minimiser l'impact.

Communications anonymes : il s'agit d'empêcher la surveillance et le suivit d'un véhicule particulier sur la route.

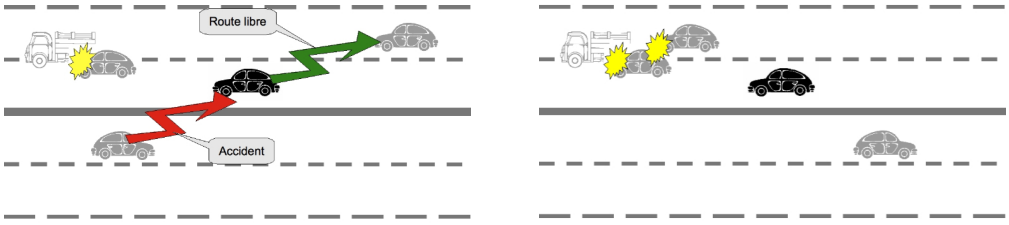
Communications certifiées : il s'agit de permettre le travail des autorités uniquement dans des circonstances particulières (une enquête après un accident, un délit de fuite).

Dans la section 2.1, nous présentons quelques travaux antérieurs sur la sécurité des réseaux de véhicules. Puis, dans la section 2.2, nous décrivons les éléments de base constituant un TPM ainsi que les différents types de clés utilisées.

[1] Par la suite nous utilisons le terme *mémoire USB* plutôt que l'habituel *clé USB* pour éviter toute confusion avec les *clés* cryptographiques.

2.1 La sécurité dans les réseaux de véhicules

Les réseaux de véhicules utilisent des communications sans fil, les données sont donc diffusées sur un medium partagé et non sûr. Il est alors très simple pour un nœud malicieux d'intercepter et de modifier des données, ou d'injecter de faux messages. Une injection de données peut provoquer des collisions dans un convoi de véhicules [BE04], comme le montre la figure 2. La nature ouverte des VANETs rend la sécurisation des communications difficile [ZMTV02, HvL04, PP05, ABD+06].



(a) Interception, injection (b) Collision

Fig. 2: Un exemple d'attaque par injection dans un VANET.

Une approche connue pour la sécurisation des communications dans les réseaux de véhicules est l'utilisation d'une infrastructure de gestion de clés dédiée au VANET [RH05, RPH06] (VPKI). Dans cette approche, l'hypothèse fondamentale est que des stations de base sont disposées le long des routes pour supporter cette infrastructure et notamment la distribution et la révocation des clés.

Les solutions basées sur une VPKI résolvent le problème de protection de la vie privée en utilisant un ensemble de *clés anonymes* et un algorithme permettant de changer périodiquement la clé utilisée pour éviter la possibilité de pouvoir tracer un véhicule particulier. Si la clé utilisée n'était pas changée périodiquement, un véhicule utiliserait toujours la même clé pour signer tous ses messages. Il deviendrait alors simple pour une personne observant le trafic réseau de corréler une clé donnée avec le trajet d'un véhicule. Les VPKI sont une approche prometteuse pour la sécurité des réseaux de véhicules, néanmoins, leur déploiement à large échelle est long, difficile et potentiellement très coûteux car il nécessite des tests grandeur nature pour assurer le bon fonctionnement en conditions réelles.

D'autres travaux traitent du problème de préservation de la vie privée [Döt05, GFL+07, FFBA07] dans les VANET avec l'aide d'une infrastructure (stations de base et autorité de certification) et l'utilisation de pseudonymes. [FFBA07] traite des challenges rencontrés lorsqu'on applique le principe d'anonymat aux communications dans les réseaux de véhicules et propose une architecture pour supporter le pseudonymat. Une étude de l'impact sur le routage géographique de l'utilisation et du changement de pseudonyme a été faite dans [SKL+06]. Il est à noter que toutes les solutions présentées dans cette section nécessitent que des stations de bases soient déployées avec une densité suffisante. Nous noter aussi que tous les travaux basés sur l'utilisation d'un ensemble de clés ou de pseudonymes certifiés ne discutent pas réellement la charge induite sur l'autorité de certification, ni du problème de passage à l'échelle induit par le nombre de certificats à produire pour toute la flotte de véhicules.

2.2 Présentation des *Trusted Platform Module*

Un exemple de puce matérielle conçue pour l'informatique de confiance est le *Trusted Platform Module* (TPM) [TCG05] qui peut être intégré dans n'importe quel périphérique. Les TPM sont maintenant intégrés dans de multiple PC, 200 millions de PC possédant un TPM ont été vendus en 2007. Un TPM est une puce matérielle nécessitant une infrastructure logicielle capable de protéger et stocker des données dans des emplacements sécurisés. Un TPM possède aussi des capacités cryptographiques telles qu'un moteur RSA, un moteur SHA-1 et un générateur de nombres aléatoires. La figure 3, issue de [TCG05], présente les principaux composants d'un TPM.

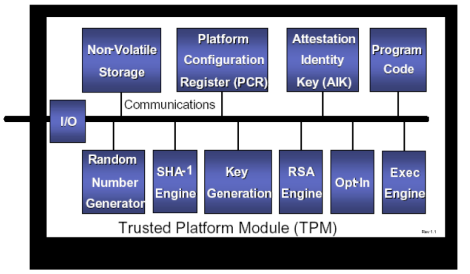


Fig. 3: Architecture d'un TPM

Plusieurs certificats servent à démontrer qu'un TPM fonctionne correctement et que les clés servant à attester son identité (*i.e.* des clés générées par le TPM et certifiées par un tiers de confiance) ont été générées par un TPM valide. Les plus importants sont :

- Chaque TPM possède une clé maîtresse unique appelée *Endorsement Key* (EK). Il s'agit d'une paire de clés RSA d'une taille minimum de 2048 bits. L'EK est générée par le constructeur du TPM et est certifiée.
- Le *Platform Credential*[2] qui est créé et signé par celui qui fournit la plate-forme dans laquelle est intégré le TPM. Il identifie la plate-forme.
- En général, le fournisseur de la plate-forme intégrant le TPM teste ou fait tester par un tiers de confiance le TPM. Si les tests sont passés avec succès un *Conformance Credential* est délivré pour ce TPM.
- Un TPM peut générer des *Attestation Identity Key* (AiK) qui servent dans le protocole d'attestation spécifique au TPM. Des certificats peuvent aussi être générés pour ces clés pour prouver qu'elles ont été créées par un TPM valide. Les spécifications des TPM décrivent un protocole faisant intervenir un tiers de confiance appelé *Privacy Certification Authority* (PCA) qui peut certifier les AiK créées par un TPM. L'avantage de cette méthode est que l'utilisation d'une AiK certifiée ne nécessite pas de dévoiler l'identité de la plate-forme ou du TPM (via l'EK ou le *Platform Credential*) lors du déroulement du protocole d'attestation. Cela diminue les possibilités de traçage par analyse du matériel cryptographique utilisé.

[2] Dans le jargon TPM un *credential* est un certificat

3 Une architecture fournissant sécurité et anonymat pour les communications inter-véhicules

La solution proposée est basée sur plusieurs éléments matériels :
- un TPM et de la mémoire dans le véhicule,
- une mémoire USB (MS) transportée par le conducteur,
- un serveur en ligne pour la PCA.

La solution met aussi en jeu plusieurs entités :
- le constructeur du véhicule qui effectue les opérations de démarrage du système,
- le conducteur ainsi que l'entité chargée de la maintenance du véhicule qui peuvent effectuer les opérations de mise à jour,
- l'autorité administrative qui gère le cycle de vie du véhicule (vente, revente, destruction,...),
- une PCA qui publie aussi les listes de révocation.

Avant d'aller plus avant dans la description de notre solution, nous passons en revue les principales contraintes de conception :
- fournir une meilleure sécurité pour les communications inter-véhicules,
- fournir l'anonymat des communications inter-véhicules et en particulier éviter l'utilisation pendant une longue durée du même identifiant,
- fournir à une entité administrative autorisée la possibilité de lever *a posteriori* l'anonymat d'un message particulier,
- respecter les mécanismes de sécurité natifs des TPM,
- donner la possibilité de passer dans un état de fonctionnement standard lorsqu'une des entités mise en jeu n'effectue pas les opérations nécessaires.

Dans la section 3.1, nous adaptons le protocole d'attestation natif des *AiK* [TCG05] pour la signature des messages envoyés par les véhicules. Dans la section 3.2, nous présentons un exemple typique de communication inter-véhicule [GB08]. Dans la section 3.3, nous décrivons les interactions entre le TPM, la mémoire USB (MS) et la PCA.

3.1 La signature des messages inter-véhicules

Les TPM offrent deux types d'authentification anonyme : l'utilisation d'une PCA ou l'utilisation du protocole DAA (*Direct Anonymous Attestation* [BCC04]). Dans notre proposition, nous nous basons sur l'utilisation d'une PCA, principalement car elle permet de faire lever l'anonymat par un tiers de confiance autorisé (voire la section 3.4) et pour des raisons de passages à l'échelle (voire la section 4.3). Il est à noter qu'une solution basée sur DAA n'est pas impossible, mais cela nécessiterait une étude complète et certainement une adaptation car il est indiqué dans le résumé de [BCC04] : *DAA can be seen as a group signature without the feature that a signature can be opened, i.e., the anonimity is not revocable.* Or, l'une de nos hypothèses est que l'anonymat soit révocable par une autorité et dans certaines conditions pour des raisons de responsabilité en cas d'accident.

Une première solution naïve qui peut être envisagée en utilisant un TPM est d'utiliser une *AiK* dédiée à la signature des messages envoyés par le véhicule. Deux problèmes s'opposent à cette approche. Tout d'abord, une *AiK* ne peut pas être utilisée pour signer

des données arbitraires, tel que le signalent les spécifications des TPM. Cela a pour but de ne pas affaiblir une *AiK* en fournissant beaucoup de données signées par cette clé et donc de matériel à cryptanalyser. Il ne faut pas oublier qu'une *AiK* possède un certain pouvoir d'attestation. Heureusement, il existe une astuce connue pour pallier ce problème [KS07]. L'*AiK* est autorisée à signer des données et des clés générées par le TPM. La solution consiste donc à demander au TPM de générer une clé appelée *Key Signing*, puis de certifier cette clé grâce à une *AiK*. Nous avons donc une *AiK* certifiée par véhicule, ce qui ne représente pas une grosse charge pour la PCA, un certificat est créé à la construction du véhicule.

Le second problème est qu'avec cette approche, un véhicule est traçable, car l'*AiK* utilisée pour signer toutes les *Key Signing* peut alors être utilisée comme un identifiant unique. Même si cet identifiant ne contient pas au premier abord d'élément permettant de révéler l'identité de son propriétaire, cela peut au minimum révéler la trajectoire et les déplacements d'un véhicule. De plus, il devient assez facile de corréler certaines informations comme le type de véhicule si l'attaquant est suffisamment proche pour voir quel véhicule démarre lorsqu'une *AiK* donnée commence à apparaître. Cette approche ne convient donc pas à nos prérequis d'anonymat.

Une amélioration de cette solution, toujours en utilisant l'astuce des *Key Signing*, est de stocker un très grand nombre d'*AiK* certifiées dans le véhicule lors de sa phase de construction. En supposant qu'un véhicule parcourt en moyenne 25000 kilomètres par an et que l'on change de clé tous les 500 mètres pour éviter d'être tracé, nous avons besoin de 50000 clés par an dans chaque véhicule. En France, environ 2 millions de véhicules neufs sont vendus chaque année. Cela signifie donc que la PCA (une autorité de l'état) devra créer 10^{11} certificats par an. Nous avons là un problème de passage à l'échelle. De plus, si cet ensemble de clés doit être renouvelé chaque année, le nombre de certificat à créer va augmenter jusqu'à atteindre 5.10^4 fois le nombre de véhicules en circulation (environ 20 millions en France) par an.

Devant ces problèmes, nous avons décidé d'utiliser une solution hybride, c'est-à-dire que nous pré-chargeons un certain nombre de clés dans le véhicule à sa construction, et nous utilisons un protocole exploitant une mémoire USB pour renouveler les clés certifiées de manière opportuniste et les charger dans le véhicule.

3.2 Les communications inter-véhicules

Un scénario de communications inter-véhicules utilisant la solution présentée dans ce papier est décrite sur la figure 4.

Dans le schéma proposé, comme les *AiK* ne peuvent signer que des données produites par le TPM, nous utilisons une *Key Signing* produite par le TPM et signée par une *AiK* [KS07], pour signer les messages envoyés par l'application embarquée dans le véhicule.

Lorsqu'un véhicule A veut envoyer un message msg_A à un véhicule B, l'application tournant sur le véhicule A date le message à l'aide de la date et de l'heure courante du véhicule (pas nécessairement à jour), puis envoie une requête `TPM_Sign`($msg_A.date$) à son TPM pour faire signer le message. Le TPM renvoie à l'application le message daté signé ainsi que les certificats nécessaires à sa vérification. L'application A envoie alors le message daté signé et le matériel cryptographique associé à B (l'envoi peut être une diffusion). À la réception, l'application tournant sur le véhicule B vérifie les certificats et les signatures. La seule clé publique devant être connue de B est la clé publique de la

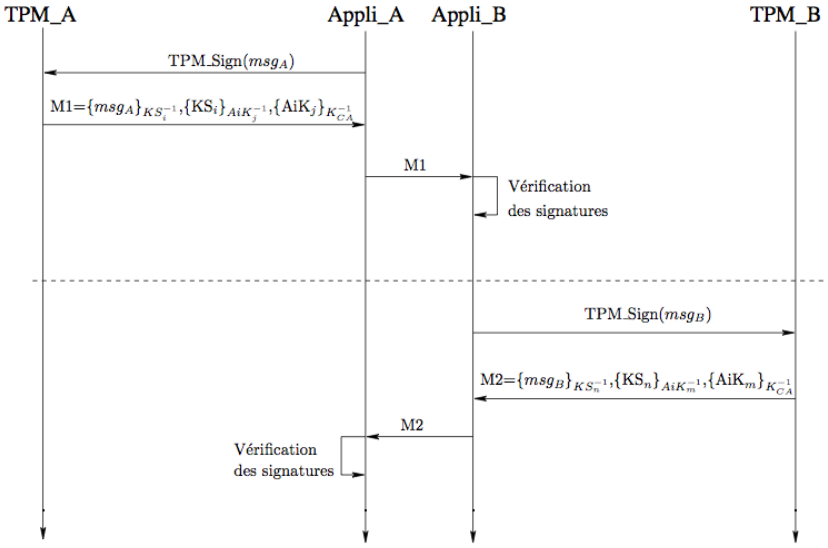


Fig. 4: Une communication inter-véhicules utilisant des messages signés (le TPM n'est utilisé que pour signer les messages).

PCA. Nous pouvons noter ici que le TPM du véhicule B n'est pas sollicité pour les étapes de vérifications. L'application peut également vérifier la date du message et la comparer à la date de sont véhicule (pas nécessairement à jour). Cette comparaison optionnelle peut permettre d'éliminer facilement des cas inutiles ou des rejeux triviaux, comme lorsque la date du message reçu est très antérieures à la date courante.

Pour fournir un anonymat de bonne qualité, il est nécessaire de changer périodiquement la *Key Signing* utilisée et donc par extension l'AiK utilisée, pour éviter les possibilités de tracer un véhicule. Il faudra donc veiller à ce que le TPM ait suffisamment d'AiK à disposition. Les aspects quantitatifs sont discutés en 4.3.

3.3 Interactions entre le TPM et la PCA

Lors d'une communication, le TPM utilise une de ses AiK choisie dans un ensemble fini d'AiK signées. C'est cette AiK qui certifiera la clé effectivement utilisée pour signer le message (pour rappel les spécifications du TPM interdisent de signer directement avec une AiK). Pour un meilleur anonymat l'AiK utilisée ne doit pas être traçable. Elle doit donc changer fréquemment. Néanmoins, l'ensemble d'AiK certifiées disponibles n'est pas infini, il faut donc le renouveler.

Les interactions avec la PCA décrits sur la figure 5 permettent justement d'obtenir de nouvelles AiK signées. Elles ont lieu :

- à la mise en service d'un nouveau véhicule, c'est alors le constructeur qui peut se charger d'insérer les AiK signées dans la mémoire associée au TPM.
- au fil du temps lorsque le TPM d'un véhicule déjà en service demande des AiK supplémentaires.

– lors de la cession d'un véhicule à un nouveau propriétaire ; dans ce cas particulier, la commande standard `TPM_OwnerClear` rend les précédentes AiK inutilisables et provoque la demande de nouvelles AiK.

Les demandes d'AiK supplémentaires et les réponses transitent *via* la mémoire USB du propriétaire du véhicule. Conformément à l'une des contraintes de conception, si le propriétaire n'utilise jamais sa mémoire USB alors le système se comporte au pire comme un système de communication standard : les AiK signées ne sont pas renouvelées, et la qualité de l'anonymat se dégrade.

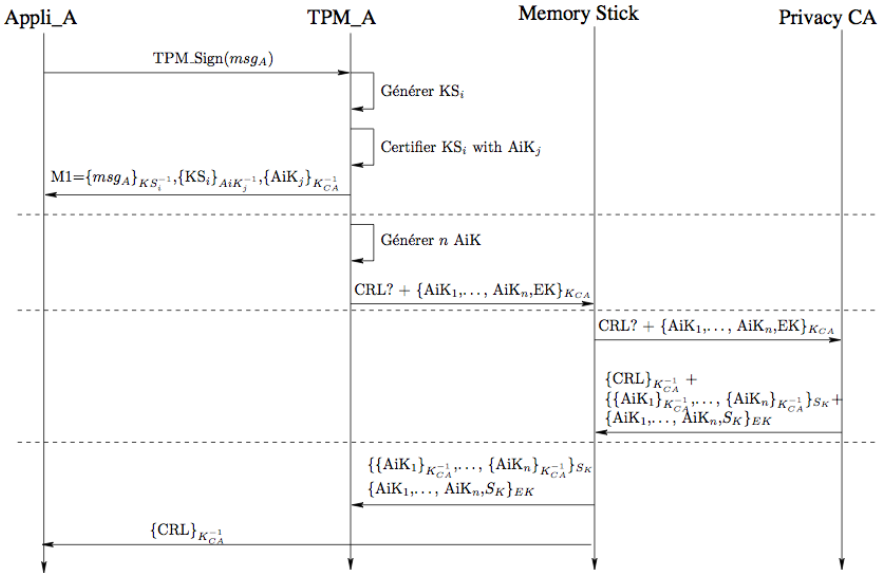


Fig. 5: Interactions entre les différents éléments de l'architecture. On peut remarquer que l'information échangée via la mémoire USB est soit publique (CRL) soit chiffrée ($AiK\ldots$)

Lorsqu'un véhicule se déplace, le TPM utilise ses AiK et *Key Signing* pour signer les messages à envoyer aux autres véhicules. Une fois qu'une *Key Signing* a été utilisée pendant un certain temps, une certaine distance ou a déjà signé un certain nombre de messages, le TPM change de *Key Signing*. Lorsque le nombre de *Key Signing* disponible (ou par extension d'AiK disponible) tombe en dessous d'un seuil fixé, le TPM crée une demande de certification contenant l'ensemble des nouvelles AiK à certifier concaténé au certificat de l'EK. Cette demande est chiffrée avec la clé publique de la PCA puis stockée dans la mémoire USB. Lorsque la mémoire USB sera connectée à l'Internet, la demande sera relayée à la PCA. La PCA déchiffrera la demande, vérifiera le certificat de l'EK et pourra alors certifier les AiK fournies tout en gardant une trace en associant ces AiK au certificat de l'EK. La PCA doit vérifier le certificat de l'EK afin d'être sûr que la demande vient d'un TPM valide et si nécessaire faire le lien entre une EK et un TPM pour les besoins d'une enquête de police.

La réponse de la PCA contenant les AiK certifiées est alors chiffrée avec une clé de session S_K choisie par la PCA. Cette clé est concaténée avec les AiK et l'ensemble et

chiffrée par l'EK puis renvoyée à la mémoire USB. Cela permet notamment de diminuer le volume de données chiffrées par l'EK et de correspondre à la logique TPM qui ne transmettra la clé de session S_K à l'application qu'en cas de réussite de la vérification des AiK. À la prochaine introduction de la mémoire USB dans le véhicule, les AiK certifiées seront délivrées au TPM. Chiffrer la clé de session avec l'EK permet de se prémunir contre l'utilisation frauduleuse d'une EK d'un autre véhicule dans le but d'obtenir des clés certifiées. Seul le TPM possédant la partie privée de l'EK pourra déchiffrer la clé de session générée par la PCA. De plus, comme il peut y avoir un nombre conséquent d'AiK à signer dans notre cas, l'utilisation une clé de session symétrique est particulièrement bien adaptée.

3.4 Levée partielle de l'anonymat

L'ajout d'anonymat dans les communications inter-véhicules ne doit pas gêner le travail des autorités dans les situations autorisées. Cet aspect est important pour le déploiement d'une solution réaliste. Nous pensons en particulier qu'une solution sans trappe a peu de chance d'être acceptée dans le contexte automobile où le conducteur engage sa responsabilité.

Nous attirons l'attention sur le fait que les systèmes actuels d'immatriculation offrent exactement une fonctionnalité de ce type : l'immatriculation fournit un anonymat pour tout un chacun et seule une entité autorisée peut remonter du numéro de l'immatriculation vers l'identité du conducteur.

Nous introduisons un paramètre supplémentaire noté α dans les certificats des AiK. Ce paramètre permet à une autorité connaissant le secret K_{auth}^{-1}, et à elle seule, de retrouver une information j concernant le véhicule. L'information en question est le numéro d'ordre du véhicule dans une base de données ou même son numéro d'immatriculation. Ainsi, le certificat de la $i^{\text{ème}}$ AiK soumise à la PCA aura la forme :

$$\{AiK_i, \alpha_i\}_{K_{CA}^{-1}} \text{ avec } \alpha_i = \{j, i\}_{K_{auth}}$$

Plusieurs points sont importants dans l'expression de α :

- Le paramètre i permet de diversifier les α_i afin qu'un observateur extérieur ne puisse pas savoir si deux α_i sont attribués à un même véhicule j ou non.
- La clé K_{auth} utilisée pour fabriquer les α n'est pas publique ; elle doit être connue uniquement de l'entité habilitée à fabriquer les α_i, en l'occurrence la PCA.
- La clé K_{auth}^{-1} nécessaire pour lever l'anonymat n'est pas publique ; elle doit être connue uniquement du tiers autorisé, qui n'est pas obligatoirement la PCA.

3.5 Révocation des clés

Deux types de clés peuvent être révoqués dans notre modèle, les AiK et les EK. Il serait plus difficile de révoquer les *Key Signing* car la PCA n'a pas connaissance de ces clés, de plus il n'est pas indispensable de révoquer une *Key Signing* si celle-ci venait à être corrompue. En effet, la *Key Signing* est certifiée par une AiK, il est donc plus facile de révoquer cette AiK connue de la PCA.

Lorsque la PCA a connaissance d'une clé corrompue, AiK ou EK, elle met à jour sa liste de révocation et incrémente son numéro de série. L'utilisation de listes de révocation basées sur une journalisation des changements est essentielle dans notre contexte pour

diminuer la taille des messages transmis. Dans notre architecture, nous utilisons donc un modèle de révocation tout à fait classique pour révoquer les AiK corrompues. Lors de sa demande de CRL, le véhicule transmet le dernier numéro de série connu à la mémoire USB, ce numéro sera transmis à la PCA qui répondra alors avec les différentes modifications apportées à la CRL depuis ce numéro de série.

Pour révoquer une EK, le système est un peu plus complexe. En effet, cela signifie que les fonctions premières du TPM ne sont plus remplies. La preuve de la compromission d'une EK peut être donnée après enquête, dans ce cas la PCA doit révoquer toutes les AiK associées à cette EK, placer aussi cette EK dans la liste de révocation et conserver l'information localement afin de ne plus certifier les AiK émanant de ce TPM.

4 Analyse

Dans cette section nous argumentons la sécurité de notre solution vis-à-vis d'un attaquant ayant tout pouvoir sur les communications (typiquement lecture, écriture, destruction). En particulier l'attaquant peut manipuler les communications radio inter-véhicules, les communications intra-véhicule entre le TPM et son application, les communications au travers de la mémoire USB et enfin les communications via Internet.

4.1 Modélisation du protocole

Nous utilisons les outils `AVISPA` [ABB+05] et `SPAN` [SPA] pour vérifier deux propriétés du protocole :

1. le secret de la clé de session entre la PCA et le TPM
2. le secret des AiK entre le TPM et la PCA via la mémoire USB.

Le secret des AiK pendant leur transport n'est pas indispensable mais illustre le fait qu'aucune hypothèse n'est nécessaire concernant la sécurité de la mémoire USB utilisée. Celle-ci peut être perdue, dupliquée, modifiée, sans dommage pour le protocole autre que la simple perte des données contenues.

N.B. : les AiK ne sont plus secret dès leur utilisation standard par le TPM et l'application gérant le TPM. Typiquement, ils sont transmis en clair avec les messages qu'ils ont contribué à signer.

Nous reproduisons en annexe la spécification "standard.hlpsl" utilisée pour la vérification des deux propriétés de secret. Dans les deux cas, `AVISPA` ne relève aucune attaque.

Par ailleurs, nous avons développé une spécification plus compacte (nommée "compact.hlpsl" et reproduite en annexe) où le canal entre un TPM et son application n'est plus accessible à l'attaquant. Les vérifications sur cette spécification n'ont montré aucune attaque sur le secret des informations échangées.

4.2 Analyse de la levée partielle de l'anonymat

La solution indiquée en 3.4 permet à une entité autorisée de lever l'anonymat d'un message particulier. En effet, un message fabriqué à l'aide d'un des AiK a la forme suivante :

$$\{msg\}_{KS}, \{KS\}_{AiK_i{}^{-1}}, \{AiK, \alpha_i\}_{AiK_i{}^{-1}}$$

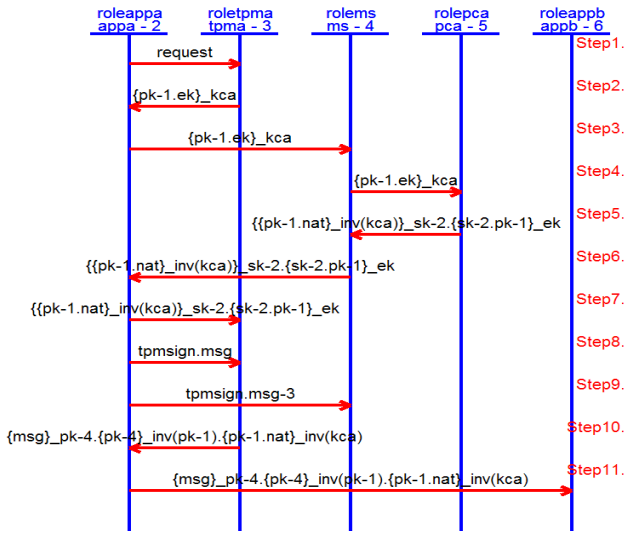


Fig. 6: Un diagramme d'exécution SPAN du protocole tel que spécifié dans "standard.hlpsl".

Toute entité en possession d'un tel message peut facilement retrouver α_i, mais seule une autorité autorisée (possédant K_{auth}^{-1}) pourra retrouver le paramètre j en déchiffrant $\alpha_i = \{j, i\}_{K_{auth}}$.

Une collaboration est alors nécessaire entre l'entité ayant retrouvé j et la PCA, afin de résoudre l'opération de jointure menant à l'identité du véhicule. Cette collaboration est essentielle pour rendre le mécanisme résistant à la compromission d'une des deux autorités. Il faudrait donc une corruption des deux autorités pour casser l'anonymat de manière non autorisée. Nous pensons que cette éventualité est peu probable.

4.3 Aspects quantitatifs

Il est clair que la qualité de l'anonymat croit avec le nombre d'AiK certifiées à disposition du TPM. Une approche directe consisterait à utiliser une AiK sur une courte période (ou sur une courte distance parcourue par le véhicule), et à la détruire immédiatement après. Si par exemple la même AiK est utilisée pour les communications pendant 500 mètres parcourus par le véhicule, si ce même véhicule parcourt 25000 kilomètres par an, alors il faut disposer de 50000 AiK certifiés par an. En d'autres termes, pour ce véhicule, le TPM devrait générer 50000 AiK que la PCA devrait certifier.

Une estimation montre rapidement que cette approche n'est pas valide en temps de certification. En effet, sur la base de 2 millions de nouveaux véhicules par an rien qu'en France, la PCA devrait pouvoir certifier 6.10^9 par mois, ce qui dépasse la cadence actuelle des autorités de certifications, de l'ordre de 2.10^6 (source IBM [BGA+07] cadence de signature estimée avec matériel dédié).

Il est donc nécessaire de réutiliser les AiK pour obtenir un compromis réaliste entre temps de certification et anonymat. Avec 2 millions de véhicules par an et une cadence de l'ordre de 2.10^6 AiK certifiés par an le compromis se situe à quelques milliers d'AiK

certifiés par nouveau véhicule.

Ces *AiK* peuvent être complétées par d'autres *AiK* lors des allers-retours de la mémoire USB, compensant au fil du temps le vieillissement de l'ensemble. N.B. : il ne s'agit pas ici d'alléger la cadence de certification, mais de lisser la charge au cours du temps.

Ainsi, en ajoutant de l'ordre de 100 *AiK* par véhicule et par mois, l'augmentation de la charge de la PCA est faible tout en permettant un renouvellement partiel des *AiK*.

Contrairement au temps de certification, l'encombrement des *AiK* n'est pas problématique : la taille d'un *AiK* est celle d'une paire de clés de 2048 bits. Les mémoires flash actuelles, dépassant le Go à faible coût, permettent donc d'en stocker plus que nécessaire. Des opérations de réinitialisation peuvent éventuellement avoir lieu à l'occasion d'un changement de propriétaire (les anciennes *AiK* devenant inutilisables suite au `TPM_OwnerClear`).

4.4 Applications incitatives

Une utilisation régulière de la mémoire USB permet de faire le lien entre le TPM et la PCA et donc de renouveler les *AiK* certifiées, en vue d'un meilleur anonymat. Afin d'inciter l'utilisateur à utiliser sa mémoire USB, il est possible de collecter des données supplémentaires utilisées par d'autres applications : consommation, distances parcourues, statistiques de conduite. De telles applications motivent l'utilisateur à brancher de temps à autres sa mémoire USB dans son ordinateur et c'est à cette occasion, de manière opportuniste, que des *AiK* certifiées pourront être récupérées.

De plus, des opérations de maintenance régulières du véhicule (révisions, contrôles techniques) peuvent être l'occasion d'une récupération d'*AiK* certifiées.

5 Conclusion

Nous avons montré une manière d'utiliser les composants TPM pour améliorer la sécurité et l'anonymat des communications dans les VANET. Pour autant, cette utilisation n'est pas immédiate et nécessite des dispositions particulières telles que la mise en place d'un moyen de communication opportuniste avec la PCA, l'utilisation de clés de signature éphémères pour les messages, l'ajout d'un paramètre permettant de lever l'anonymat dans les cas autorisés. Dans la solution décrite, le lien avec la PCA est illustré par l'utilisation d'une mémoire USB. Ce dispositif nécessite l'intervention du conducteur et peut donc être un frein au déploiement de la solution. Néanmoins, aucune hypothèse de sécurité n'est nécessaire concernant le lien utilisé. On peut donc envisager l'utilisation de tout autre lien comme par exemple un téléphone 3G dans la voiture qui se connecterait à la PCA via une liaison GSM. On pourrait aussi envisager un module de communication sans fil intégré dans la clé de contact et le module idoine connecté à un PC domestique. Ainsi, lorsque la clé est dans le véhicule elle communique avec le TPM et lorsqu'elle est proche du PC connecté à Internet elle communique avec la PCA.

Enfin, il est possible que le mécanisme DAA (*Direct Anonymous Attestation*) maintenant intégré dans les TPM puisse être adapté au contexte des VANET. Ce mécanisme est potentiellement plus léger que le mécanisme PCA puisqu'il ne nécessite pas d'autorité de certification. En revanche il peut compliquer d'autres opérations telles que la levée de l'anonymat. Une étude spécifique semble donc nécessaire.

Remerciements

Nous remercions les relecteurs de ce papier pour leurs remarques et suggestions d'amélioration.

Annexe

"standard.hlpsl"

```
role roletpma (APPA,APPB,TPMA,PCA,MS:agent,EK,Kca:public_key,SND,RCV:channel(dy)) played_by TPMA def=
local   State,Alpha:nat, AiK,SignKey:public_key, Msg,Break:message
init    State :=1
transition
1.      State  =1 /\ RCV(request)       =|>     State':=2 /\ AiK':=new() /\ SND({AiK'.EK}_Kca) /\ secret(AiK',sec_aik,{TPMA,PCA})
2.      State  =2 /\ RCV(Msg')          =|>     State':=3
3.      State  =3 /\ RCV(tpmsign.Break')=|>     State':=4 /\ SignKey':=new() /\ SND({Break'}_SignKey'.{SignKey'}_inv(AiK).{AiK.Alpha}_inv(Kca))
end role

role roleappa (APPA,APPB,TPMA,PCA,MS:agent,EK,Kca:public_key,SND,RCV:channel(dy)) played_by APPA def=
local   State,Alpha:nat, SignedBreak,Break,Msg:message, AiK:public_key
init    State :=1
transition

1.      State  =1 /\ RCV(start)         =|>     State':=2 /\ SND(request)
2.      State  =2 /\ RCV({AiK'.EK}_Kca) =|>     State':=3 /\ SND({AiK'.EK}_Kca)
3.      State  =3 /\ RCV(Msg')          =|>     State':=4 /\ SND(Msg')
4.      State  =4                       =|>     State':=5 /\ Break':=new() /\ SND(tpmsign.Break')
5.      State  =5 /\ RCV(SignedBreak')  =|>     State':=6 /\ SND(SignedBreak')
end role

role rolems (APPA,APPB,TPMA,PCA,MS:agent,EK,Kca:public_key,SND,RCV:channel(dy)) played_by MS def=
local   State :nat, Msg:message
init    State :=1
transition
1.      State  =1 /\ RCV(Msg')          =|>     State':=1 /\ SND(Msg')
end role

role rolepca (APPA,APPB,TPMA,PCA,MS:agent,EK,Kca:public_key,SND,RCV:channel(dy)) played_by PCA def=
local   Alpha,State:nat, Session:symmetric_key, AiK:public_key
init    State :=1
transition
1.      State  =1 /\ RCV({AiK'.EK'}_Kca)=|>     State':=2 /\ Session':=new() /\
        SND({{AiK'.Alpha}_inv(Kca)}_Session'.{Session'.AiK'}_EK') /\
        secret(Session',sec_session,{PCA,TPMA})
end role

role roleappb (APPA,APPB,TPMA,PCA,MS:agent,EK,Kca:public_key,SND,RCV:channel(dy)) played_by APPB def=
local   Alpha,State:nat, Msg,SignedBreak:message, AiK,SignKey:public_key
init    State :=1
transition
1.      State  =1 /\ RCV(SignedBreak')  =|>     State':=2
end role

role environment() def=
local   SND, RCV: channel(dy)
const   appa,appb,tpma,ms,pca:agent, request,tpmsign : message, ek, kca : public_key
intruder_knowledge = {appa,appb,tpma,ms,pca,ek,kca}
composition
        roleappa(appa,appb,tpma,pca,ms,ek,kca,SND,RCV)
/\      roletpma(appa,appb,tpma,pca,ms,ek,kca,SND,RCV)
/\      rolems  (appa,appb,tpma,pca,ms,ek,kca,SND,RCV)
/\      rolepca (appa,appb,tpma,pca,ms,ek,kca,SND,RCV)
/\      roleappb(appa,appb,tpma,pca,ms,ek,kca,SND,RCV)
end role

goal
        secrecy_of sec_session,sec_aik
end goal

environment()
```

"compact.hlpsl"

```
role rolecara (CARA,CARB,PCA,MS:agent,EK,Kca:public_key,SND,RCV:channel(dy)) played_by CARA def=
local   State,Alpha:nat, AiK,SignKey:public_key, SignedBreak,Msg,Break:message
init    State :=1
transition
1.      State  =1 /\ RCV(start)         =|>     State':=2 /\ AiK':=new() /\ secret(AiK',sec_aik,{CARA,PCA}) /\ SND({AiK'.EK}_Kca)
2.      State  =2 /\ RCV(Msg')          =|>     State':=3
3.      State  =3 /\ RCV(tpmsign.Break')=|>     State':=4 /\ SignKey':=new()
                  /\ SND({Break'}_SignKey'.{SignKey'}_inv(AiK).{AiK.Alpha}_inv(Kca))
                  /\ secret(AiK,sec_aik,{CARA,PCA})
4.      State  =4                       =|>     State':=5 /\ Break':=new() /\ SND(tpmsign.Break')
5.      State  =5 /\ RCV(SignedBreak')  =|>     State':=6 /\ SND(SignedBreak')
```

```
end role

role rolems (CARA,CARB,PCA,MS:agent,EK,Kca:public_key,SND,RCV:channel(dy)) played_by MS def=
local   State :nat, Msg:message
init    State :=1
transition
1.      State  =1 /\ RCV(Msg')          =|>     State':=1 /\ SND(Msg')
end role

role rolepca (CARA,CARB,PCA,MS:agent,EK,Kca:public_key,SND,RCV:channel(dy)) played_by PCA def=
local   Alpha,State:nat, Session:symmetric_key, AiK:public_key
init    State :=1
transition
1.      State  =1 /\ RCV({AiK'.EK'}_Kca)=|>     State':=2 /\ Session':=new() /\ secret(Session',sec_session,{PCA,CARA})
                  /\ SND({{AiK'.Alpha}_inv(Kca)}_Session'.{Session'.AiK'}_EK')
end role

role rolecarb (CARA,CARB,PCA,MS:agent,EK,Kca:public_key,SND,RCV:channel(dy)) played_by CARB def=
local   Alpha,State:nat, Msg,SignedBreak:message, AiK,SignKey:public_key
init    State :=1
transition
1.      State  =1 /\ RCV(SignedBreak') =|>      State':=2
end role

role environment() def=
local   SND, RCV: channel(dy)
const   cara,carb,ms,pca:agent, request,tpmsign:message, ek,kca:public_key
intruder_knowledge = {cara,carb,ms,pca,ek,kca}
composition
        rolecara(cara,carb,pca,ms,ek,kca,SND,RCV) /\ rolems  (cara,carb,pca,ms,ek,kca,SND,RCV)
/\      rolepca (cara,carb,pca,ms,ek,kca,SND,RCV) /\ rolecarb(cara,carb,pca,ms,ek,kca,SND,RCV)
end role

goal
        secrecy_of sec_session,sec_aik
end goal

environment()
```

Références

[ABB+05] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, P.-C. Héam, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. von Oheimb, M. Rusinowitch, J. Santos Santiago, M. Turuani, L. Viganò, and L. Vigneron. The AVISPA Tool for the automated validation of internet security protocols and applications. In K. Etessami and S. Rajamani, editors, *17th International Conference on Computer Aided Verification, CAV'2005*, volume 3576 of *Lecture Notes in Computer Science*, pages 281–285, Edinburgh, Scotland, 2005. Springer.

[ABD+06] A. Aijaz, B. Bochow, F. Dötzer, A. Festag, M. Gerlach, R. Kroh, and T. Leinmüller. Attacks on Inter-Vehicle Communication Systems - An Analysis. In *3rd International Workshop on Intelligent Transportation*, 2006.

[BCC04] E. Brickell, J. Camenisch, and L. Chen. Direct Anonymous Attestation. In *ACM Conference on Computer and Communication Security (CCS)*, pages 132–145, 2004.

[BE04] J. Blum and A. Eskandarian. The Threat of Intelligent Collisions. *IT Professional*, 6(1) :24–29, January-February 2004.

[BGA+07] P. Bari, M Gasparovic, H. Almeida, G. Detro, D. Druker, M. Gnirss, JF. Jiguet, and M. Raicher. *Security on z/VM.* IBM Redbooks, 2007.

[CG07a] M. Conti and S. Giordano. Multihop ad hoc networking : The reality. *IEEE Cmmunications Magazine*, 45(4) :88–95, 2007.

[CG07b] M. Conti and S. Giordano. Multihop ad hoc networking : The theory. *IEEE Cmmunications Magazine*, 45(4) :78–86, 2007.

[Döt05] F. Dötzer. Privacy Issues in Vehicular Ad Hoc Network. In *Workshop on Privacy Enhancing Technologies*, pages 197–209, 2005.

[FFBA07] E. Fonseca, A. Festag, R. Baldessari, and R. Aguiar. Support of Anonymity in VANETs - Putting Pseudonymity into Practice. In *IEEE Wireless Communications and Networking Conference*, 2007.

[GB08] G. Guette and C. Bryce. Using TPMs to secure Vehicular Ad hoc Networks (VANETs). In *Workshop on Intormation Theory and Practices*, pages 106–116, 2008.

[GFL+07] M. Gerlach, A. Festag, T. Leinmüller, G. Goldacker, and C. Harsch. Security Architecture for Vehicular Communication. In *Workshop on Intelligent Transportation*, 2007.

[HvL04] JP. Hubaux, S. Čapkun, and J. Luo. The Security and Privacy of Smart Vehicles. *IEEE Security and Privacy*, 2(3) :49–55, May-June 2004.

[KS07] N. Kuntze and A. U. Schmidt. Trusted Ticket Systems and Applications. In *IFIP sec : New Approaches for Security, Privacy and Trust in Complex Environments*, pages 49–60, 2007.

[LLC+07] K. C. Lee, SH. Lee, R. Cheung, U. Lee, and M. Gerla. First Experience with CarTorrent in a Real Vehicular Ad Hoc Network Testbed. In *Mobile Networking for Vehicular Environments*, pages 109–114, 2007.

[PP05] B. Parno and A. Perrig. Challenges in Securing Vehicular Networks. In *Fourth Workshop on Hot Topics in Networks*, 2005.

[RH05] M. Raya and JP. Hubaux. The Security of Vehicular Ad Hoc Networks. In *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks*, pages 11–21, 2005.

[RPH06] M. Raya, P. Papadimitratos, and JP. Hubaux. Securing Vehicular Communications. *IEEE Wireless Communications Magazine, Special Issue on Inter-Vehicular Communications*, 13(5) :8–15, 2006.

[SKL+06] E. Schoch, F. Kargl, T. Leinmüller, S. Schlott, and P. Papadimitratos. Impact of pseudonym changes on geographic routing in vanets. In *European Workshop on Security in Ad-hoc and Sensor Networks*, pages 43–57, 2006.

[SPA] A Security Protocol Animator for AVISPA (SPAN). `http ://www.irisa.fr/lande/genet/span/`.

[TCG05] Trusted Computing Group. TPM Main Specification. Main Specification Version 1.2 rev. 85, Trusted Computing Group, February 2005.

[TCGW] Trusted Computing Group Website. https ://www.trustedcomputinggroup.org/home.

[ZMTV02] M. El Zarki, S. Mehrotra, G. Tsudik, and N. Venkatasubramanian. Security Issues in a Future Vehicular Network. In *European Wireless*, 2002.